66AK2G1x
Multicore DSP+Arm® KeyStone II System-on-Chip (SoC)

Texas Instruments Family of Products

Technical Reference Manual

Literature Number: SPRUHY8I
January 2016–Revised March 2019

Contents

List of Figures

List of Tables

Revision History

Changes from H Revision (May 2018) to I Revision **Page**

- Added Table Note for USB in Table Device Memory Map 215
- Updated ARMSS Endian Support Details in Section ARMSS Endian Configuration Registers and BOOTCFG_DEVSTAT Register 336
- Updated PLL Controller Base Address in Register Section of Section Clock Management 569
- Updated ARMSS Endian Support Details in Section Features, Section AXI2VBUS_MASTER Overview, and AXI2VBUS_CPU0_END Register 597
- Fixed Typo in GPIO Pin Numbers Controlled by GPIO Registers 2676
- Updated MMCHS_HCTL[2] HSPE and MMCHS_CAPA[21] HSS Bits in Section MMC/SD 3304
- Added Clarification in Section QSPI Overview 3965

Read This First

About This Manual

This Technical Reference Manual (TRM) details the integration, the environment, the functional description, and the programming models for each peripheral and subsystem in the device.

Related Documentation From Texas Instruments

For a complete listing of related documentation and development-support tools for the device, visit the Texas Instruments website at www.ti.com.

Trademarks

Neon is a trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.
Arm, Cortex are registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere.
EtherCAT is a trademark of Beckhoff Automation GmbH.
Ethernet Powerlink is a trademark of Bernecker + Rainer Industrie Elektronik GmbH.
QSPI is a trademark of Cadence Design Systems, Inc.
MIPI is a registered trademark of MIPI Alliance, Inc.
MediaLB is a registered trademark of Microchip Technology Inc.
EtherNet/IP is a trademark of ODVA, Inc.
PCIe is a registered trademark of PCI-SIG.
PROFINET, PROFIBUS are registered trademarks of PROFIBUS and PROFINET International.
SERCOS is a trademark of SERCOS INTERNATIONAL E.V.
OneNAND is a trademark of Samsung Electronics Co.
All other trademarks are the property of their respective owners.

Introduction

This chapter introduces the features, subsystems, and architecture of 66AK2G1x System on Chip (SoC).

Topic **Page**

1.1 Device Overview

The 66AK2G1x SoC is a high performance, highly integrated device based on TI KeyStone II Multicore SoC architecture. It incorporates the performance optimized Arm® Cortex®-A15 and a C66x DSP core, built to meet the processing and system level integration needs of industrial communications and control, automotive and performance audio applications.

The device is targeted for a variety of applications which include, but are not limited to:

- Automotive Audio Amplifiers
- Home Audio
- Professional Audio
- Computer Navigational Control (Motion Control)
- Smart Grid (Intelligent Electric Device)
- Industrial Programmable Logic Control (PLC)
- Test and Measurement

NOTE: The supported set of features and peripherals is device part number dependent. For more information, see the device Data Manual.

The device is composed of the following main subsystems, among others:

- One Arm subsystem (ARMSS), based on a single-core Cortex-A15 processor
- One Digital Signal Processor (DSP) C66x subsystem
- Two dual-core Programmable Real-time Unit and Industrial Communication Subsystems (PRU-ICSS)
- One Display subsystem (DSS)
- One Network subsystem (NSS), consisting of Ethernet MAC (EMAC), Navigator subsystem (NAVSS), and Security Accelerator (SA)

The device provides a rich set of connectivity peripherals:

- Two USB 2.0 High Speed subsystems with integrated PHY
- One single-lane PCI Express Gen2 subsystem
- Peripherals for general connectivity, including I2C (x3), SPI (x4), UART (x3), and GPIO (x2)
- Media and storage interfaces, including one GPMC, one Quad-SPI, and MMC/SD (x2) interfaces
- Audio peripherals, including McASP (x3), one McBSP interface, and one Asynchronous Audio Sample Rate Converter (ASRC)
- Automotive interfaces, including CAN (x2), and one MediaLB® interface
- Industrial and control interfaces, including ePWM (x6), eCAP (x2), and eQEP (x3) interfaces

The device includes a dedicated power manager micro controller (PMMC), facilitating power and sleep control (PSC) management, power state transitions, and power related system configurations.

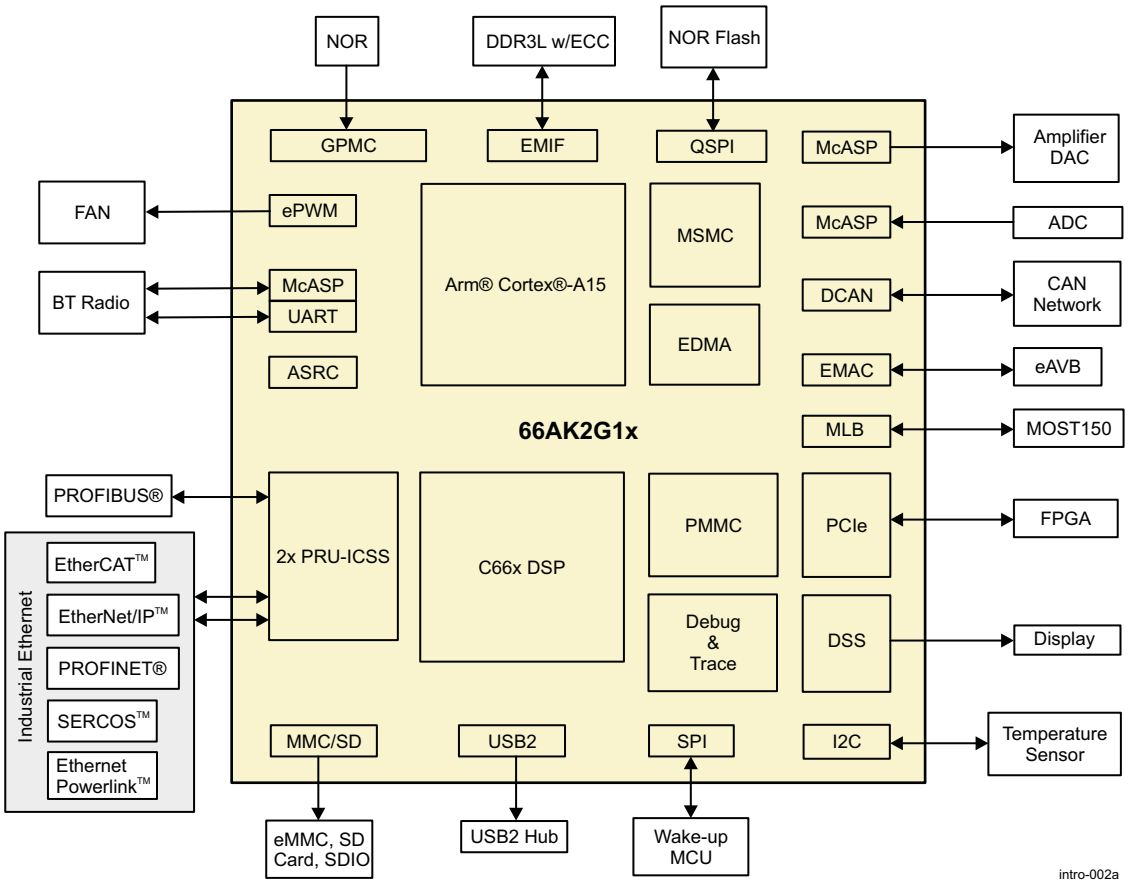
The device also integrates the following components, among others:

- Multicore Shared Memory Controller (MSMC)
- DDR External Memory Interface (EMIF)
- Message Manager (MSGMGR)
- Seven 64-bit Timer modules
- Semaphore
- Enhanced DMA controller (EDMA)
- Debug and Trace subsystem

1.2 Device Environment

Figure 1-1 is an example for a non-exhaustive connectivity diagram of the device.

Figure 1-1. Device Environment Diagram

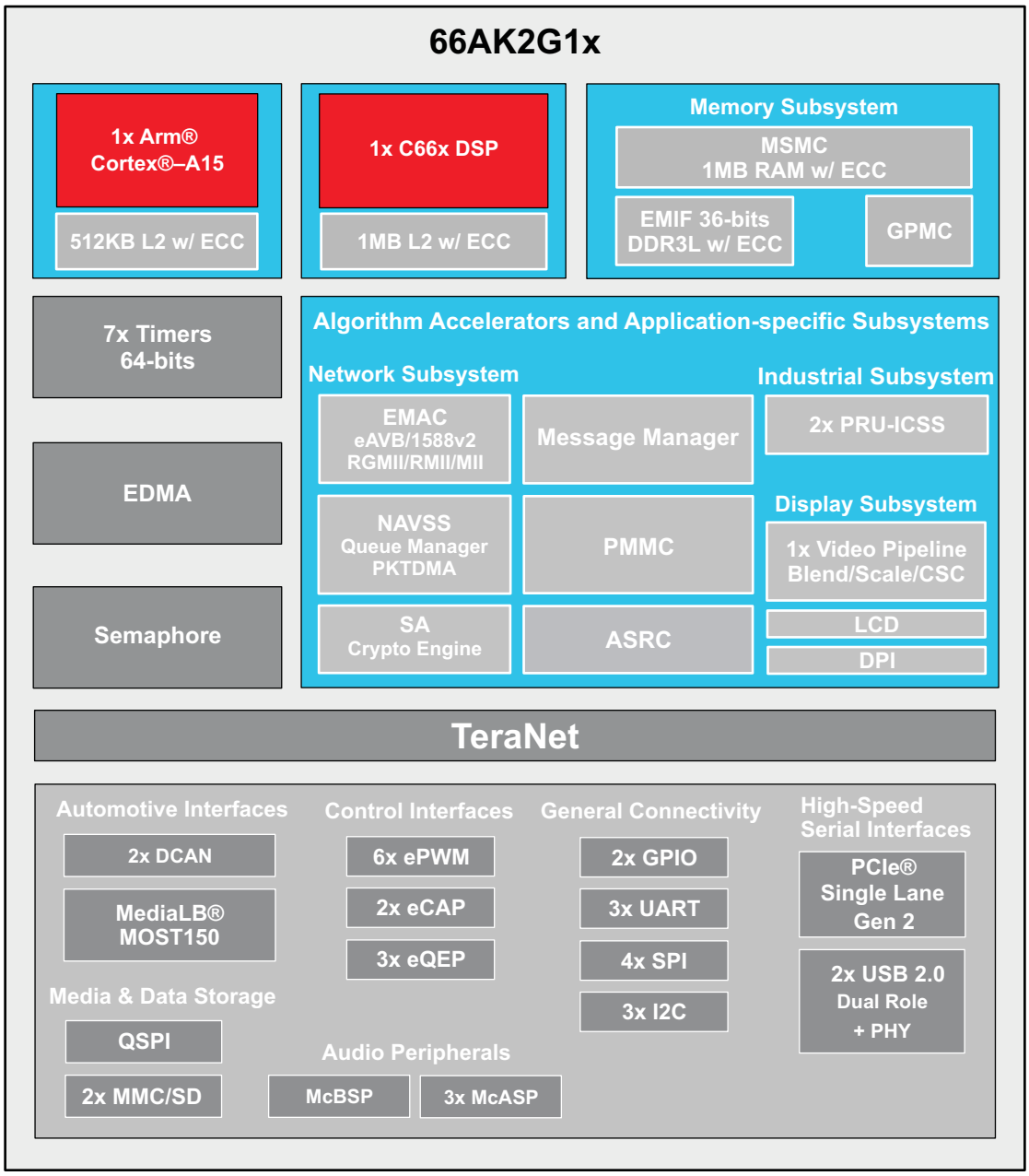


NOTE: The supported set of features and peripherals is device part number dependent. For more information, see the device Data Manual.

1.3 Device Description

Figure 1-2 is a block diagram of the device.

Figure 1-2. Device Block Diagram



intro_001a

NOTE: The supported set of features and peripherals is device part number dependent. For more information, see the device Data Manual.

1.3.1 Arm Subsystem

The Arm Cortex-A15 subsystem provides the following main features, among others:

- Arm Cortex-A15 processor, with one central processing unit (CPU)

- Full implementation of Armv7-A architecture instruction set
- Integrated Arm® Neon™ and VFP (Vector Floating Point) unit
- 32-KB instruction and 32-KB data level 1 (L1) cache
- 512-KB level 2 (L2) cache
- Error Correction Code (ECC) protection for L1 data cache and L2 cache, parity protection for L1 instruction cache
- Support for up to four integrated general-purpose timers, in adition to one device-level watchdog timer
- Local advanced power management
- Dedicated digital phase-locked loop (ARMPLL)
- Integrated Generic Interrupt Controller (GIC)
- Debug and trace features

1.3.2 DSP Subsystem

The C66x DSP subsystem provides the following main features, among others:

- Fixed/Floating-point C66x CPU based on a superset of the C64x+ and C67x+ ISA
- 32-KB L1D and 32-KB L1P cache or addressable SRAM
- 1024-KB local L2 cache or addressable SRAM
- Extended Memory Controller (XMC)
- Address extension unit to 36-bit address
- Memory protections for multiple segments, and for internal L1/L2 RAM
- Error Detection for L1P
- Error Detection and Correction for L1D, and for L2
- Integrated interrupt controller
- Support for one integrated general-purpose timer, in adition to one device-level watchdog timer
- Debug and trace features

1.3.3 Algorithm Accelerators and Application-specific Subsystems

1.3.3.1 Asynchronous Audio Sample Rate Converter (ASRC)

ASRC provides the following main features, among others:

- Hardware accelerated audio sample rate converter
- Support interpolation and decimation for asynchronous clock zone crossing for audio streams
- Capable of -140dB SNR operation
- Supports up to 16 concurrent audio channels
- Supports 4 input clock zones, and 4 output clock zones
- Up to 216 kHz sample rate input/output, sample ratios from 16:1 to 1:16
- Support direct interfacing with McASP for input or output via EDMA, without software data packing
- Two modes of operation: group mode and stream mode
- Separate DMA events for input and output for each channel and groups of channels

1.3.3.2 Display Subsystem (DSS)

DSS provides the following main features, among others:

- One video pipeline with in-loop scaling, color space conversion, and background color overlay
- Video pipeline pixel formats: BITMAP, RGB16, RGB24, RGB32, RGB48, ARGB16, ARGB32, ARGB48, YUV420, YUV422, RGB565-A8
- In-loop scaling capabilities:

  - Supports minimum of 1/4 downscaler, and up to x16 upscaler
  - Support for 5-tap (up to 1280 pixels ARGB32) or 3-tap (up to 2560 pixels ARGB32) filters
- Supported display interfaces:
  - MIPI® DPI 2.0 parallel interface
  - BT.656 4:2:2
  - BT.1120 4:2:2
  - Remote Frame Buffer Interface for MIPI DBI 2.0 support
- Display panel support features

1.3.3.3 Programmable Real-time Unit and Industrial Communication Subsystem (PRU-ICSS)

There are two PRU-ICSS in the device, each providing the following main features, among others:

- Two PRUs, each with multiplier and optional accumulator, with 16-KB program memory and 8-KB data memory
- 64-KB general purpose memory with ECC
- ECC support for all internal memories
- Two MII_RT ports configurable to connect to each PRU to support multiple industrial communication protocols
- One Industrial Ethernet Peripheral (IEP) to manage/generate Industrial Ethernet functions
- One MDIO
- One UART 16550, with a dedicated 192 MHz to support 12 Mbps PROFIBUS
- One Industrial Ethernet 64-bit timer with 9 capture and 16 compare events, alogn with slow and fast compensation
- One integrated Enhanced Capture Module (ECAP)
- Power management support

Among the interfaces supported by the PRU-ICSS are real-time industrial protocols used in master and slave mode, such as:

- EtherCAT™
- PROFINET®
- EtherNet/IP™
- PROFIBUS®
- Ethernet Powerlink™
- SERCOS™

1.3.3.4 Message Manager

Message Manager provides the following main features, among others:

- Hardware acceleration for pushing and popping messages to/from logical queues in a multicore environment
- Supports up to 512 messages, and 64 queues
- Support for (self-contained) mode with zero software initialization
- Flexible message allocation with ability to store the same message multiple times in different queues or multiple times in the same queue

1.3.3.5 Power Management Micro Controller (PMMC)

PMMC provides the following main features, among others:

- PSC management, power state transitions, and power related system configurations
- 40-KB unified memory, and 8-KB data memory for the PMMC processor, with double detection and single error correction for both memories

- Two Timer modules and one Window Watchdog Timer

1.3.3.6 Network Subsystem (NSS)

NSS consists of the following main submodules:

- Ethernet MAC subsystem (EMAC), providing the following main features, among others:
  - One-port Gigabit Ethernet: RMII/RGMII
  - 10/100/1000 Mbps full duplex support
  - 10/100 Mbps half duplex support
  - Ethernet Audio/Video Bridging (eAVB) support
  - IEEE 1588 v2 (with Annex D/E/F) Support
  - SER protection (SECDED) support
- Navigator subsystem (NAVSS), including PKTDMA controller, and Queue Manager with SER protection (SECDED) support
- Security Accelerator, providing the following encryption/decryption functions, among others:
  - Crypto function library for software acceleration (AES, 3DES, SHA1, MD5, SHA2 - 224/256)
  - Block data encryption supported via hardware cores
  - Data encryption modes supported via MCE (programmable Mode Control Engine)
  - Public Key accelerator (PKA) with Elliptic Curve Cryptography (ECC)
  - Keyed HMAC operation via hardware core
  - True Random number generator (TRNG)

1.3.4 Memory Controllers and Memory

- Multicore Shared Memory Controller (MSMC), providing the following main features, among others:
  - Common managed path to the DDR EMIF for multiple DSP cores, Arm cores, and other system masters
  - 1MB MSMC SL2 RAM (in a single bank) shared between the Arm and DSP cores, with ECC
  - 256-bit slave interfaces for DSP and Arm cores
  - Two 256-bit VBUSM slave interfaces for DMA traffic (one for MSMC SRAM, another for DDR EMIF)
  - Memory protection unit (MPU) for both MSMC SRAM and DDR memory
- DDR External Memory Interface (EMIF), providing the following main features, among others:
  - 36-bit DDR3L interface (32-bit data + 4-bit ECC)
  - DDR3L at 800 MT/s maximum speed for wirebond devices
  - DDR3L at 1066 MT/s maximum speed for flip-chip devices
  - Up to 8GB memory address range
- General Purpose Memory Controller (GPMC), providing the following main features, among others:
  - Flexible 8/16-bit asynchronous memory interface with up to 4 chip selects (NOR, Muxed-NOR, SRAM, and so forth)
  - Support for synchronous and asynchronous interface protocols with external memories or devices
  - Support for IO speeds up to 100MHz under synchronous mode, and up to 133MHz with asynchronous mode

1.3.5 Connectivity Peripherals

- One PCIe® 2.0 controller, Gen2 compliant, with single SerDes lane running at 5GBaud/2.5GBaud. Supports both root complex mode and end point mode.
- Two USB 2.0 High Speed dual-role ports with integrated PHY, each supporting:
  - USB 2.0 peripheral at speeds HS (480 Mb/s) and FS (12 Mb/s)
  - USB 2.0 host at speeds HS (480 Mb/s), FS (12 Mb/s), and LS (1.5 Mb/s)

- Three I²C interfaces, each supporting Standard mode (up to 100 KHz) and Fast mode (up to 400 KHz), with full 7-bit address field.
- Four SPI interfaces, each supporting two chip selects, and up to 50 MHz in master mode, and up to 25MHz in slave mode operation.
- Three UART interfaces, each supporting 16550 compatibility, with operation at up to 3M baud:
  - UART0 supports 8 pins with full modem control
  - UART1 and UART2 are 4-pin interfaces
- Two General Purpose IO (GPIO) modules, providing up to 212 GPIO pins with muxing.

1.3.6 Media and Storage Interfaces

- One Quad-SPI (QSPI™) module, for communication with up to four external flash devices.
- Two MMC/SD ports, both acting as HS-MMC/SD initiator controllers, supporting JEDEC JESD84 v4.5-A441 and SD3.0 physical layer with SDA3.00 standards:
  - MMC0 controller supports only 3.3 V IO modes
  - MMC1 controller supports only 1.8 V IO modes

1.3.7 Audio Peripherals

- Three Multi-Channel Audio Serial Ports (McASP), each providing support for TDM, I^2S and similar formats, DIT output, and allowing transmit/receive clock rates up to 50 MHz:
  - McASP0 with 16 serial data pins, McASP1 with 10 serial data pins, and McASP2 with 6 serial data pins
- One Multi-Channel Buffered Serial Port (McBSP), providing support for TDM, I^2S, and similar formats, and allowing transmit/receive clock rates up to 50 MHz.

1.3.8 Automotive Peripherals

- Two CAN Controller Ports (DCAN), for interface with Controller Area Network bus. Each DCAN supporting CAN version 2 part A, B, with bit rates up to 1 MBit/s.
- One Media Local Bus (MLB) module, supporting both 3-pin (up to MOST50, 1024×Fs) and 6-pin (up to MOST150, 2048×Fs) versions of MediaLB® Physical Layer Specification v4.2, with speed limited to 100 Mbps.

1.3.9 Control Interfaces

- Six enhanced High Resolution PWM modules (ePWM), each supporting dedicated 16-bit time-base with Period/Frequency control, and up to two independent PWM outputs.
- Two 32-bit enhanced Capture modules (eCAP), each configurable as either one capture input, or as one auxiliary PWM output.
- Three 32-bit enhanced Quadrature Pulse Encoder modules (eQEP), for position, speed, and frequency measurements.

1.3.10 System Level Components

- Seven 64-bit Timer modules:
  - Two dedicated 64-bit timers, one per Arm and DSP core, serving as watch dogs, or for use as general purpose timers
  - Four 64-bit timers shared for general purposes
  - One 64-bit timer dedicated to PMMC
- Semaphore module, supporting total of 64 semaphores.
- Two Enhanced DMA controllers, providing totaly:
  - Two Channel Controllers (CC)
  - Two 128-bit Transfer Controllers (TC) per CC
- Boot ROM: 128KB for DSP core, and 256KB for Arm core.

Memory Map

This chapter summarizes the memory map address regions for the device.

Topic **Page**

2.1 Memory Map Summary

Table 2-1 shows the memory map address ranges of the device.

Table 2-1. Device Memory Map

Physical 40-bit Address		Bytes	Arm Cortex-A15 View	C66X_COREPAC View	Other Device Masters View
Start	End				
00 0000 0000	00 0003 FFFF	256K	ARM_ROM_0_DATA	Reserved	ARM_ROM_0_DATA
00 0004 0000	00 007F FFFF	8M-256K	Reserved	Reserved	Reserved
00 0080 0000	00 0087 FFFF	512K	Reserved	C66X_COREPAC_LOCAL_L2_SRAM	Reserved
00 0088 0000	00 008F FFFF	512K	Reserved	C66X_COREPAC_LOCAL_L2_SRAM	Reserved
00 0090 0000	00 00DF FFFF	5M	Reserved	Reserved	Reserved
00 00E0 0000	00 00E0 7FFF	32K	Reserved	C66X_COREPAC_LOCAL_L1P_SRAM	Reserved
00 00E0 8000	00 00EF FFFF	1M-32K	Reserved	Reserved	Reserved
00 00F0 0000	00 00F0 7FFF	32K	Reserved	C66X_COREPAC_LOCAL_L1D_SRAM	Reserved
00 00F0 8000	00 00FF FFFF	1M-32K	Reserved	Reserved	Reserved
00 0100 0000	00 0100 FFFF	64K	ARM EN registers	C66X_COREPAC registers	Reserved
00 0101 0000	00 010F FFFF	1M-64K	Reserved	C66X_COREPAC registers	Reserved
00 0110 0000	00 0110 FFFF	64K	ARM STM registers	C66X_COREPAC registers	Reserved
00 0111 0000	00 01BF FFFF	11M-64K	Reserved	C66X_COREPAC registers	Reserved
00 01C0 0000	00 01CF FFFF	1M	Reserved	Reserved	Reserved
00 01D0 0000	00 01D0 007F	128	TRACER_0	TRACER_0	TRACER_0
00 01D0 0080	00 01D0 7FFF	32K - 128	Reserved	Reserved	Reserved
00 01D0 8000	00 01D0 807F	128	TRACER_1	TRACER_1	TRACER_1
00 01D0 8080	00 01D0 FFFF	32K - 128	Reserved	Reserved	Reserved
00 01D1 0000	00 01D1 007F	128	TRACER_2	TRACER_2	TRACER_2
00 01D1 0080	00 01D1 7FFF	32K - 128	Reserved	Reserved	Reserved
00 01D1 8000	00 01D1 807F	128	TRACER_3	TRACER_3	TRACER_3
00 01D1 8080	00 01D1 FFFF	32K - 128	Reserved	Reserved	Reserved
00 01D2 0000	00 01D2 007F	128	TRACER_4	TRACER_4	TRACER_4
00 01D2 0080	00 01D2 7FFF	32K - 128	Reserved	Reserved	Reserved
00 01D2 8000	00 01D2 807F	128	TRACER_5	TRACER_5	TRACER_5
00 01D2 8080	00 01D2 FFFF	32K - 128	Reserved	Reserved	Reserved
00 01D3 0000	00 01D3 007F	128	TRACER_6	TRACER_6	TRACER_6
00 01D3 0080	00 01D3 7FFF	32K - 128	Reserved	Reserved	Reserved
00 01D3 8000	00 01D3 807F	128	TRACER_7	TRACER_7	TRACER_7
00 01D3 8080	00 01D3 FFFF	32K - 128	Reserved	Reserved	Reserved
00 01D4 0000	00 01D4 007F	128	TRACER_8	TRACER_8	TRACER_8
00 01D4 0080	00 01D4 7FFF	32K - 128	Reserved	Reserved	Reserved
00 01D4 8000	00 01D4 807F	128	TRACER_9	TRACER_9	TRACER_9
00 01D4 8080	00 01D4 FFFF	32K - 128	Reserved	Reserved	Reserved
00 01D5 0000	00 01D5 007F	128	TRACER_10	TRACER_10	TRACER_10
00 01D5 0080	00 01D5 7FFF	32K - 128	Reserved	Reserved	Reserved
00 01D5 8000	00 01D5 807F	128	TRACER_11	TRACER_11	TRACER_11
00 01D5 8080	00 01D5 FFFF	32K - 128	Reserved	Reserved	Reserved

Table 2-1. Device Memory Map (continued)

Physical 40-bit Address		Bytes	Arm Cortex-A15 View	C66X_COREPAC View	Other Device Masters View
Start	End				
00 01D6 0000	00 01D6 007F	128	TRACER_12	TRACER_12	TRACER_12
00 01D6 0080	00 01D6 7FFF	32K - 128	Reserved	Reserved	Reserved
00 01D6 8000	00 01D6 807F	128	TRACER_13	TRACER_13	TRACER_13
00 01D6 8080	00 01D6 FFFF	32K - 128	Reserved	Reserved	Reserved
00 01D7 0000	00 01D7 007F	128	TRACER_14	TRACER_14	TRACER_14
00 01D7 0080	00 01D7 7FFF	32K - 128	Reserved	Reserved	Reserved
00 01D7 8000	00 01D7 807F	128	Reserved	Reserved	Reserved
00 01D7 8080	00 01D7 FFFF	32K - 128	Reserved	Reserved	Reserved
00 01D8 0000	00 01D8 007F	128	Reserved	Reserved	Reserved
00 01D8 0080	00 01D8 7FFF	32K - 128	Reserved	Reserved	Reserved
00 01D8 8000	00 01D8 807F	128	Reserved	Reserved	Reserved
00 01D8 8080	00 01D8 8FFF	4K - 128	Reserved	Reserved	Reserved
00 01D9 0000	00 01D9 007F	128	Reserved	Reserved	Reserved
00 01D9 0080	00 01D9 7FFF	32K - 128	Reserved	Reserved	Reserved
00 01D9 8000	00 01D9 807F	128	Reserved	Reserved	Reserved
00 01D9 8080	00 01D9 FFFF	32K - 128	Reserved	Reserved	Reserved
00 01DA 0000	00 01DA 007F	128	Reserved	Reserved	Reserved
00 01DA 0080	00 01DA 7FFF	32K - 128	Reserved	Reserved	Reserved
00 01DA 8000	00 01DA 807F	128	Reserved	Reserved	Reserved
00 01DA 8080	00 01DA FFFF	32K - 128	Reserved	Reserved	Reserved
00 01DB 0000	00 01DB 007F	128	Reserved	Reserved	Reserved
00 01DB 0080	00 01DB 7FFF	32K - 128	Reserved	Reserved	Reserved
00 01DB 8000	00 01DB 807F	128	Reserved	Reserved	Reserved
00 01DB 8080	00 01DB 8FFF	4K - 128	Reserved	Reserved	Reserved
00 01DC 0000	00 01DC 007F	128	Reserved	Reserved	Reserved
00 01DC 0080	00 01DC 7FFF	32K - 128	Reserved	Reserved	Reserved
00 01DC 8000	00 01DC 807F	128	Reserved	Reserved	Reserved
00 01DC 8080	00 01DC FFFF	32K - 128	Reserved	Reserved	Reserved
00 01DD 0000	00 01DD 007F	128	Reserved	Reserved	Reserved
00 01DD 0080	00 01DD 7FFF	32K - 128	Reserved	Reserved	Reserved
00 01DD 8000	00 01DD 807F	128	Reserved	Reserved	Reserved
00 01DD 8080	00 01DD FFFF	32K - 128	Reserved	Reserved	Reserved
00 01DE 0000	00 01DE 007F	128	Reserved	Reserved	Reserved
00 01DE 0080	00 01DE 03FF	1K - 128	Reserved	Reserved	Reserved
00 01DE 0400	00 01DE 047F	128	Reserved	Reserved	Reserved
00 01DE 0480	00 01DE 07FF	1K - 128	Reserved	Reserved	Reserved
00 01DE 0800	00 01DE 087F	128	Reserved	Reserved	Reserved
00 01DE 0880	00 01DE 7FFF	30K - 128	Reserved	Reserved	Reserved
00 01DE 8000	00 01DE 807F	128	Reserved	Reserved	Reserved
00 01DE 8080	00 01DF FFFF	96K - 128	Reserved	Reserved	Reserved
00 01E0 0000	00 01E3 FFFF	256K	Reserved	Reserved	Reserved
00 01E4 0000	00 01E7 FFFF	256K	Reserved	Reserved	Reserved
00 01E8 0000	00 01E8 3FFF	16K	ARM_SS	ARM_SS	ARM_SS
00 01E8 4000	00 01E8 43FF	1K	MSMC_PBIST	MSMC_PBIST	MSMC_PBIST
00 01E8 4400	00 01EB FFFF	239K	Reserved	Reserved	Reserved
00 01EC 0000	00 01EF FFFF	256K	Reserved	Reserved	Reserved

Table 2-1. Device Memory Map (continued)

Physical 40-bit Address		Bytes	Arm Cortex-A15 View	C66X_COREPAC View	Other Device Masters View
Start	End				
00 01F0 0000	00 01F7 FFFF	512K	Reserved	Reserved	Reserved
00 01F8 0000	00 01FD FFFF	384K	Reserved	Reserved	Reserved
00 01FE 0000	00 01FF FFFF	128K	Reserved	Reserved	Reserved
00 0200 0000	00 020F FFFF	1M	Reserved	Reserved	Reserved
00 0210 0000	00 0215 FFFF	384K	Reserved	Reserved	Reserved
00 0216 0000	00 0217 FFFF	128K	Reserved	Reserved	Reserved
00 0218 0000	00 021A FFFF	192K	Reserved	Reserved	Reserved
00 021B 0000	00 021B FFFF	64K	Reserved	Reserved	Reserved
00 021C 0000	00 021C 03FF	1K	EQEP_0_CFG	EQEP_0_CFG	EQEP_0_CFG
00 021C 0400	00 021C 07FF	1K	EQEP_1_CFG	EQEP_1_CFG	EQEP_1_CFG
00 021C 0800	00 021C 0BFF	1K	EQEP_2_CFG	EQEP_2_CFG	EQEP_2_CFG
00 021C 0C00	00 021C 5FFF	21K	Reserved	Reserved	Reserved
00 021C 6000	00 021C 63FF	1K	MLB_0_CFG	MLB_0_CFG	MLB_0_CFG
00 021C 6400	00 021C 6FFF	3K	MLB_0_CFG	MLB_0_CFG	MLB_0_CFG
00 021C 7000	00 021C 7FFF	4K	Reserved	Reserved	Reserved
00 021C 8000	00 021C 83FF	1K	Reserved	Reserved	Reserved
00 021C 8400	00 021C FFFF	31K	Reserved	Reserved	Reserved
00 021D 0000	00 021D 007F	128	PWM_0_CFG	PWM_0_CFG	PWM_0_CFG
00 021D 0080	00 021D 00FF	128	PWM_0_EHR_CFG	PWM_0_EHR_CFG	PWM_0_EHR_CFG
00 021D 0100	00 021D 03FF	768	Reserved	Reserved	Reserved
00 021D 0400	00 021D 047F	128	PWM_1_CFG	PWM_1_CFG	PWM_1_CFG
00 021D 0480	00 021D 04FF	128	PWM_1_EHR_CFG	PWM_1_EHR_CFG	PWM_1_EHR_CFG
00 021D 0500	00 021D 07FF	768	Reserved	Reserved	Reserved
00 021D 0800	00 021D 087F	128	PWM_2_CFG	PWM_2_CFG	PWM_2_CFG
00 021D 0880	00 021D 08FF	128	PWM_2_EHR_CFG	PWM_2_EHR_CFG	PWM_2_EHR_CFG
00 021D 0900	00 021D 0BFF	768	Reserved	Reserved	Reserved
00 021D 0C00	00 021D 0C7F	128	PWM_3_CFG	PWM_3_CFG	PWM_3_CFG
00 021D 0C80	00 021D 0CFF	128	PWM_3_EHR_CFG	PWM_3_EHR_CFG	PWM_3_EHR_CFG
00 021D 0D00	00 021D 0FFF	768	Reserved	Reserved	Reserved
00 021D 1000	00 021D 107F	128	PWM_4_CFG	PWM_4_CFG	PWM_4_CFG
00 021D 1080	00 021D 10FF	128	PWM_4_EHR_CFG	PWM_4_EHR_CFG	PWM_4_EHR_CFG
00 021D 1100	00 021D 13FF	768	Reserved	Reserved	Reserved
00 021D 1400	00 021D 147F	128	PWM_5_CFG	PWM_5_CFG	PWM_5_CFG
00 021D 1480	00 021D 14FF	128	PWM_5_EHR_CFG	PWM_5_EHR_CFG	PWM_5_EHR_CFG
00 021D 1500	00 021D 17FF	768	Reserved	Reserved	Reserved
00 021D 1800	00 021D 1BFF	1K	ECAP_0_CFG	ECAP_0_CFG	ECAP_0_CFG
00 021D 1C00	00 021D 1FFF	1K	ECAP_1_CFG	ECAP_1_CFG	ECAP_1_CFG
00 021D 2000	00 021D 20FF	256	OTP_0_CFG	OTP_0_CFG	OTP_0_CFG
00 021D 2100	00 021D 23FF	768	Reserved	Reserved	Reserved
00 021D 2400	00 021D 24FF	256	Reserved	Reserved	Reserved
00 021D 2500	00 021D EFFF	51K - 256	Reserved	Reserved	Reserved
00 021D F000	00 021D F07F	128	Reserved	Reserved	Reserved
00 021D F080	00 021D FFFF	4K - 128	Reserved	Reserved	Reserved
00 021E 0000	00 021E 07FF	2K	ASRC_0_CFG	ASRC_0_CFG	ASRC_0_CFG
00 021E 0000	00 021E 07FF	2K	Reserved	Reserved	Reserved
00 021E 0800	00 021E FFFF	62K	Reserved	Reserved	Reserved

Table 2-1. Device Memory Map (continued)

Physical 40-bit Address		Bytes	Arm Cortex-A15 View	C66X_COREPAC View	Other Device Masters View
Start	End				
00 021F 0000	00 021F 07FF	2K	Reserved	Reserved	Reserved
00 021F 0800	00 021F 0FFF	2K	Reserved	Reserved	Reserved
00 021F 1000	00 021F 17FF	2K	Reserved	Reserved	Reserved
00 021F 1800	00 021F 3FFF	10K	Reserved	Reserved	Reserved
00 021F 4000	00 021F 47FF	2K	Reserved	Reserved	Reserved
00 021F 4800	00 021F 7FFF	14K	Reserved	Reserved	Reserved
00 021F 8000	00 021F 87FF	2K	Reserved	Reserved	Reserved
00 021F 8800	00 021F BFFF	14K	Reserved	Reserved	Reserved
00 021F C000	00 021F C7FF	2K	Reserved	Reserved	Reserved
00 021F C800	00 021F FFFF	14K	Reserved	Reserved	Reserved
00 0220 0000	00 0220 007F	128	TIMER_0_CFG	TIMER_0_CFG	TIMER_0_CFG
00 0220 0080	00 0220 FFFF	64K - 128	Reserved	Reserved	Reserved
00 0221 0000	00 0221 007F	128	TIMER_1_CFG	TIMER_1_CFG	TIMER_1_CFG
00 0221 0080	00 0221 FFFF	64K - 128	Reserved	Reserved	Reserved
00 0222 0000	00 0222 007F	128	TIMER_2_CFG	TIMER_2_CFG	TIMER_2_CFG
00 0222 0080	00 0222 FFFF	64K - 128	Reserved	Reserved	Reserved
00 0223 0000	00 0223 007F	128	TIMER_3_CFG	TIMER_3_CFG	TIMER_3_CFG
00 0223 0080	00 0223 FFFF	64K - 128	Reserved	Reserved	Reserved
00 0224 0000	00 0224 007F	128	TIMER_4_CFG	TIMER_4_CFG	TIMER_4_CFG
00 0224 0080	00 0224 FFFF	64K - 128	Reserved	Reserved	Reserved
00 0225 0000	00 0225 007F	128	TIMER_5_CFG	TIMER_5_CFG	TIMER_5_CFG
00 0225 0080	00 0225 FFFF	64K - 128	Reserved	Reserved	Reserved
00 0226 0000	00 0226 007F	128	TIMER_6_CFG	TIMER_6_CFG	TIMER_6_CFG
00 0226 0080	00 0226 FFFF	64K - 128	Reserved	Reserved	Reserved
00 0227 0000	00 0227 007F	128	Reserved	Reserved	Reserved
00 0227 0080	00 0227 FFFF	64K - 128	Reserved	Reserved	Reserved
00 0228 0000	00 0228 007F	128	Reserved	Reserved	Reserved
00 0228 0080	00 0228 FFFF	64K - 128	Reserved	Reserved	Reserved
00 0229 0000	00 0229 007F	128	Reserved	Reserved	Reserved
00 0229 0080	00 0229 FFFF	64K - 128	Reserved	Reserved	Reserved
00 022A 0000	00 022A 007F	128	Reserved	Reserved	Reserved
00 022A 0080	00 022A FFFF	64K - 128	Reserved	Reserved	Reserved
00 022B 0000	00 022B 007F	128	Reserved	Reserved	Reserved
00 022B 0080	00 022B FFFF	64K - 128	Reserved	Reserved	Reserved
00 022C 0000	00 022C 007F	128	Reserved	Reserved	Reserved
00 022C 0080	00 022C FFFF	64K - 128	Reserved	Reserved	Reserved
00 022D 0000	00 022D 007F	128	Reserved	Reserved	Reserved
00 022D 0080	00 022D FFFF	64K - 128	Reserved	Reserved	Reserved
00 022E 0000	00 022E 007F	128	Reserved	Reserved	Reserved
00 022E 0080	00 022E FFFF	64K - 128	Reserved	Reserved	Reserved
00 022F 0000	00 022F 007F	128	Reserved	Reserved	Reserved
00 022F 0080	00 022F 00FF	128	Reserved	Reserved	Reserved
00 022F 0100	00 022F 017F	128	Reserved	Reserved	Reserved
00 022F 0180	00 022F 01FF	128	Reserved	Reserved	Reserved
00 022F 0200	00 022F 027F	128	Reserved	Reserved	Reserved
00 022F 0280	00 022F FFFF	64K - 640	Reserved	Reserved	Reserved

Table 2-1. Device Memory Map (continued)

Physical 40-bit Address		Bytes	Arm Cortex-A15 View	C66X_COREPAC View	Other Device Masters View
Start	End				
00 0230 0000	00 0230 FFFF	64K	Reserved	Reserved	Reserved
00 0231 0000	00 0231 01FF	512	PLLC_CFG	PLLC_CFG	PLLC_CFG
00 0231 0200	00 0231 9FFF	40K - 512	Reserved	Reserved	Reserved
00 0231 A000	00 0231 BFFF	8K	Reserved	Reserved	Reserved
00 0231 C000	00 0231 DFFF	8K	Reserved	Reserved	Reserved
00 0231 E000	00 0231 FFFF	8K	Reserved	Reserved	Reserved
00 0232 0000	00 0232 3FFF	16K	PCIE_0_PHY_CFG	PCIE_0_PHY_CFG	PCIE_0_PHY_CFG
00 0232 4000	00 0232 5FFF	8K	Reserved	Reserved	Reserved
00 0232 6000	00 0232 7FFF	8K	Reserved	Reserved	Reserved
00 0232 8000	00 0232 8FFF	4K	Reserved	Reserved	Reserved
00 0232 9000	00 0232 9FFF	4K	DDR_0_PHY_CFG	DDR_0_PHY_CFG	DDR_0_PHY_CFG
00 0232 A000	00 0232 BFFF	8K	Reserved	Reserved	Reserved
00 0232 C000	00 0232 DFFF	8K	Reserved	Reserved	Reserved
00 0232 E000	00 0232 EFFF	4K	Reserved	Reserved	Reserved
00 0232 F000	00 0232 F7FF	2K	SR_0	SR_0	SR_0
00 0232 F800	00 0232 FFFF	2K	Reserved	Reserved	Reserved
00 0233 0000	00 0233 03FF	1K	Reserved	Reserved	Reserved
00 0233 0400	00 0233 07FF	1K	PCIE_0_ECC_CFG	PCIE_0_ECC_CFG	PCIE_0_ECC_CFG
00 0233 0800	00 0233 FFFF	62K	Reserved	Reserved	Reserved
00 0234 0000	00 0234 1FFF	8K	MCASP_0_CFG	MCASP_0_CFG	MCASP_0_CFG
00 0234 2000	00 0234 3FFF	8K	MCASP_1_CFG	MCASP_1_CFG	MCASP_1_CFG
00 0234 4000	00 0234 5FFF	8K	MCASP_2_CFG	MCASP_2_CFG	MCASP_2_CFG
00 0234 6000	00 0234 607F	128	MCBSP_0_CFG	MCBSP_0_CFG	MCBSP_0_CFG
00 0234 6080	00 0234 609F	32	MCBSP_0_FIFO_CFG	MCBSP_0_FIFO_CFG	MCBSP_0_FIFO_CFG
00 0234 60A0	00 0234 CFFF	28K - 160	Reserved	Reserved	Reserved
00 0234 D000	00 0234 FFFF	12K	Reserved	Reserved	Reserved
00 0235 0000	00 0235 0FFF	4K	PSC	PSC	PSC
00 0235 1000	00 0235 FFFF	60K	Reserved	Reserved	Reserved
00 0236 0000	00 0236 03FF	1K	MPU_0	MPU_0	MPU_0
00 0236 0400	00 0236 7FFF	31K	Reserved	Reserved	Reserved
00 0236 8000	00 0236 83FF	1K	MPU_1	MPU_1	MPU_1
00 0236 8400	00 0236 FFFF	31K	Reserved	Reserved	Reserved
00 0237 0000	00 0237 03FF	1K	MPU_2	MPU_2	MPU_2
00 0237 0400	00 0237 7FFF	31K	Reserved	Reserved	Reserved
00 0237 8000	00 0237 83FF	1K	MPU_3	MPU_3	MPU_3
00 0237 8400	00 0237 FFFF	31K	Reserved	Reserved	Reserved
00 0238 0000	00 0238 03FF	1K	MPU_4	MPU_4	MPU_4
00 0238 0400	00 0238 7FFF	31K	Reserved	Reserved	Reserved
00 0238 8000	00 0238 83FF	1K	MPU_5	MPU_5	MPU_5
00 0238 8400	00 0238 87FF	1K	MPU_6	MPU_6	MPU_6
00 0238 8800	00 0238 8BFF	1K	MPU_7	MPU_7	MPU_7
00 0238 8C00	00 0238 8FFF	1K	MPU_8	MPU_8	MPU_8
00 0238 9000	00 0238 93FF	1K	MPU_9	MPU_9	MPU_9
00 0238 9400	00 0238 97FF	1K	MPU_10	MPU_10	MPU_10
00 0238 9800	00 0238 9BFF	1K	MPU_11	MPU_11	MPU_11
00 0238 9C00	00 0238 9FFF	1K	MPU_12	MPU_12	MPU_12

Table 2-1. Device Memory Map (continued)

Physical 40-bit Address		Bytes	Arm Cortex-A15 View	C66X_COREPAC View	Other Device Masters View
Start	End				
00 0238 A000	00 0238 A3FF	1K	MPU_13	MPU_13	MPU_13
00 0238 A400	00 0238 A7FF	1K	MPU_14	MPU_14	MPU_14
00 0238 A800	00 0238 ABFF	1K	MPU_15	MPU_15	MPU_15
00 0238 AC00	00 0238 AFFF	1K	MPU_16	MPU_16	MPU_16
00 0238 B000	00 023F FFFF	468K	Reserved	Reserved	Reserved
00 0240 0000	00 0243 FFFF	256K	Reserved	Reserved	Reserved
00 0244 0000	00 0244 3FFF	16K	DBG_ADTF_0	DBG_ADTF_0	DBG_ADTF_0
00 0244 4000	00 0244 FFFF	48K	Reserved	Reserved	Reserved
00 0245 0000	00 0245 3FFF	16K	Reserved	Reserved	Reserved
00 0245 4000	00 0245 FFFF	48K	Reserved	Reserved	Reserved
00 0246 0000	00 0246 3FFF	16K	Reserved	Reserved	Reserved
00 0246 4000	00 0246 FFFF	48K	Reserved	Reserved	Reserved
00 0247 0000	00 0247 3FFF	16K	Reserved	Reserved	Reserved
00 0247 4000	00 0247 FFFF	48K	Reserved	Reserved	Reserved
00 0248 0000	00 0248 3FFF	16K	Reserved	Reserved	Reserved
00 0248 4000	00 0248 FFFF	48K	Reserved	Reserved	Reserved
00 0249 0000	00 0249 3FFF	16K	Reserved	Reserved	Reserved
00 0249 4000	00 0249 FFFF	48K	Reserved	Reserved	Reserved
00 024A 0000	00 024A 3FFF	16K	Reserved	Reserved	Reserved
00 024A 4000	00 024A FFFF	48K	Reserved	Reserved	Reserved
00 024B 0000	00 024B 3FFF	16K	Reserved	Reserved	Reserved
00 024B 4000	00 024B FFFF	48K	Reserved	Reserved	Reserved
00 024C 0000	00 024C 01FF	512	PBIST_COM_0	PBIST_COM_0	PBIST_COM_0
00 024C 0200	00 024C 03FF	512	Reserved	Reserved	Reserved
00 024C 0400	00 024C 07FF	1K	PBIST_CTL_0	PBIST_CTL_0	PBIST_CTL_0
00 024C 0800	00 024C 0BFF	1K	PBIST_CTL_1	PBIST_CTL_1	PBIST_CTL_1
00 024C 0C00	00 024C 0FFF	1K	PBIST_CTL_2	PBIST_CTL_2	PBIST_CTL_2
00 024C 1000	00 024C FFFF	60K	Reserved	Reserved	Reserved
00 024D 0000	00 024F FFFF	192K	Reserved	Reserved	Reserved
00 0250 0000	00 0250 7FFF	32K	SEC_CTL	SEC_CTL	SEC_CTL
00 0250 8000	00 0250 FFFF	32K	Reserved	Reserved	Reserved
00 0251 0000	00 0251 FFFF	64K	Reserved	Reserved	Reserved
00 0252 0000	00 0252 03FF	1K	Reserved	Reserved	Reserved
00 0252 0400	00 0252 FFFF	63K	Reserved	Reserved	Reserved
00 0253 0000	00 0253 007F	128	I2C_0_DATA_CFG	I2C_0_DATA_CFG	I2C_0_DATA_CFG
00 0253 0080	00 0253 03FF	1K - 128	Reserved	Reserved	Reserved
00 0253 0400	00 0253 047F	128	I2C_1_DATA_CFG	I2C_1_DATA_CFG	I2C_1_DATA_CFG
00 0253 0480	00 0253 07FF	1K - 128	Reserved	Reserved	Reserved
00 0253 0800	00 0253 087F	128	I2C_2_DATA_CFG	I2C_2_DATA_CFG	I2C_2_DATA_CFG
00 0253 0880	00 0253 0BFF	1K - 128	Reserved	Reserved	Reserved
00 0253 0C00	00 0253 0C3F	64	UART_0	UART_0	UART_0
00 0253 0C40	00 0253 0FFF	1K - 64	Reserved	Reserved	Reserved
00 0253 1000	00 0253 103F	64	UART_1	UART_1	UART_1
00 0253 1040	00 0253 13FF	1K - 64	Reserved	Reserved	Reserved
00 0253 1400	00 0253 143F	64	UART_2	UART_2	UART_2
00 0253 1440	00 0253 FFFF	59K - 64	Reserved	Reserved	Reserved

Table 2-1. Device Memory Map (continued)

Physical 40-bit Address		Bytes	Arm Cortex-A15 View	C66X_COREPAC View	Other Device Masters View
Start	End				
00 0254 0000	00 0255 FFFF	128K	DSSUL_0_CFG	DSSUL_0_CFG	DSSUL_0_CFG
00 0256 0000	00 0256 7FFF	32K	ARM_GIC_CFG	ARM_GIC_CFG	ARM_GIC_CFG
00 0256 8000	00 0257 FFFF	96K	Reserved	Reserved	Reserved
00 0258 0000	00 0259 FFFF	128K	USB_1_CFG(1)	USB_1_CFG(1)	USB_1_CFG(1)
00 025A 0000	00 025F FFFF	384K	Reserved	Reserved	Reserved
00 0260 0000	00 0260 1FFF	8K	CIC_0	CIC_0	CIC_0
00 0260 2000	00 0260 2FFF	4K	Reserved	Reserved	Reserved
00 0260 3000	00 0260 30FF	256	GPIO_0	GPIO_0	GPIO_0
00 0260 3100	00 0260 3FFF	4K - 256	Reserved	Reserved	Reserved
00 0260 4000	00 0260 5FFF	8K	DCAN_0_DATA	DCAN_0_DATA	DCAN_0_DATA
00 0260 6000	00 0260 7FFF	8K	Reserved	Reserved	Reserved
00 0260 8000	00 0260 9FFF	8K	DCAN_1_DATA	DCAN_1_DATA	DCAN_1_DATA
00 0260 A000	00 0260 A0FF	256	GPIO_1	GPIO_1	GPIO_1
00 0260 A100	00 0260 B0FF	4K	Reserved	Reserved	Reserved
00 0260 B100	00 0260 B1FF	256	Reserved	Reserved	Reserved
00 0260 B200	00 0260 B3FF	512	DCAN_0_CFG	DCAN_0_CFG	DCAN_0_CFG
00 0260 B400	00 0260 B5FF	512	DCAN_1_CFG	DCAN_1_CFG	DCAN_1_CFG
00 0260 B600	00 0260 BEFF	3K - 768	Reserved	Reserved	Reserved
00 0260 BF00	00 0260 BFFF	256	Reserved	Reserved	Reserved
00 0260 C000	00 0261 BFFF	64K	Reserved	Reserved	Reserved
00 0261 C000	00 0261 FFFF	16K	Reserved	Reserved	Reserved
00 0262 0000	00 0262 1FFF	8K	BOOT_CFG	BOOT_CFG	BOOT_CFG
00 0262 2000	00 0262 FFFF	56K	Reserved	Reserved	Reserved
00 0263 0000	00 0263 FFFF	64K	USB_0_PHY_CFG	USB_0_PHY_CFG	USB_0_PHY_CFG
00 0264 0000	00 0264 07FF	2K	SEMAPHORE	SEMAPHORE	SEMAPHORE
00 0264 0800	00 0264 FFFF	62K	Reserved	Reserved	Reserved
00 0265 0000	00 0265 FFFF	64K	USB_1_PHY_CFG	USB_1_PHY_CFG	USB_1_PHY_CFG
00 0266 0000	00 0267 FFFF	128K	Reserved	Reserved	Reserved
00 0268 0000	00 0269 FFFF	128K	USB_0_CFG(1)	USB_0_CFG(1)	USB_0_CFG(1)
00 026A 0000	00 026F FFFF	384K	Reserved	Reserved	Reserved
00 0270 0000	00 0270 7FFF	32K	EDMACC_0	EDMACC_0	EDMACC_0
00 0270 8000	00 0270 FFFF	32K	Reserved	Reserved	Reserved
00 0271 0000	00 0271 FFFF	64K	Reserved	Reserved	Reserved
00 0272 0000	00 0272 7FFF	32K	Reserved	Reserved	Reserved
00 0272 8000	00 0272 FFFF	32K	EDMACC_1	EDMACC_1	EDMACC_1
00 0273 0000	00 0273 FFFF	64K	Reserved	Reserved	Reserved
00 0274 0000	00 0274 7FFF	32K	Reserved	Reserved	Reserved
00 0274 8000	00 0275 FFFF	96K	Reserved	Reserved	Reserved
00 0276 0000	00 0276 03FF	1K	EDMATC_0	EDMATC_0	EDMATC_0
00 0276 0400	00 0276 7FFF	31K	Reserved	Reserved	Reserved
00 0276 8000	00 0276 83FF	1K	EDMATC_1	EDMATC_1	EDMATC_1
00 0276 8400	00 0276 FFFF	31K	Reserved	Reserved	Reserved
00 0277 0000	00 0277 03FF	1K	Reserved	Reserved	Reserved
00 0277 0400	00 0277 7FFF	31K	Reserved	Reserved	Reserved
00 0277 8000	00 0277 83FF	1K	Reserved	Reserved	Reserved

(1) This region contains USB Wrapper and USB Controller registers.

Table 2-1. Device Memory Map (continued)

Physical 40-bit Address		Bytes	Arm Cortex-A15 View	C66X_COREPAC View	Other Device Masters View
Start	End				
00 0277 8400	00 0277 FFFF	1K	Reserved	Reserved	Reserved
00 0278 0000	00 0278 03FF	1K	Reserved	Reserved	Reserved
00 0278 0400	00 0278 7FFF	31K	Reserved	Reserved	Reserved
00 0278 8000	00 0278 83FF	1K	Reserved	Reserved	Reserved
00 0278 8400	00 0278 FFFF	31K	Reserved	Reserved	Reserved
00 0279 0000	00 0279 03FF	1K	Reserved	Reserved	Reserved
00 0279 0400	00 0279 7FFF	31K	Reserved	Reserved	Reserved
00 0279 8000	00 0279 83FF	1K	Reserved	Reserved	Reserved
00 0279 8400	00 0279 FFFF	31K	Reserved	Reserved	Reserved
00 027A 0000	00 027A 03FF	1K	Reserved	Reserved	Reserved
00 027A 0400	00 027A 7FFF	31K	Reserved	Reserved	Reserved
00 027A 8000	00 027A 83FF	1K	Reserved	Reserved	Reserved
00 027A 8400	00 027A FFFF	31K	Reserved	Reserved	Reserved
00 027B 0000	00 027B 03FF	1K	EDMATC_2	EDMATC_2	EDMATC_2
00 027B 0400	00 027B 7FFF	31K	Reserved	Reserved	Reserved
00 027B 8000	00 027B 83FF	1K	EDMATC_3	EDMATC_3	EDMATC_3
00 027B 8400	00 027B 87FF	1K	Reserved	Reserved	Reserved
00 027B 8800	00 027B 8BFF	1K	Reserved	Reserved	Reserved
00 027B 8C00	00 027B FFFF	29K	Reserved	Reserved	Reserved
00 027C 0000	00 027C 03FF	1K	Reserved	Reserved	Reserved
00 027C 0400	00 027C FFFF	63K	Reserved	Reserved	Reserved
00 027D 0000	00 027D 3FFF	16K	Reserved	Reserved	Reserved
00 027D 4000	00 027D 7FFF	16K	Reserved	Reserved	Reserved
00 027D 8000	00 027D FFFF	32K	Reserved	Reserved	Reserved
00 027E 0000	00 027E 3FFF	16K	Reserved	Reserved	Reserved
00 027E 4000	00 027E FFFF	48K	Reserved	Reserved	Reserved
00 027F 0000	00 027F 3FFF	16K	Reserved	Reserved	Reserved
00 027F 4000	00 027F FFFF	48K	Reserved	Reserved	Reserved
00 0280 0000	00 0280 3FFF	16K	Reserved	Reserved	Reserved
00 0280 4000	00 0280 FFFF	48K	Reserved	Reserved	Reserved
00 0281 0000	00 0281 3FFF	16K	Reserved	Reserved	Reserved
00 0281 4000	00 0281 FFFF	48K	Reserved	Reserved	Reserved
00 0282 0000	00 0282 3FFF	16K	Reserved	Reserved	Reserved
00 0282 4000	00 0282 FFFF	48K	Reserved	Reserved	Reserved
00 0283 0000	00 0283 3FFF	16K	Reserved	Reserved	Reserved
00 0283 4000	00 0283 FFFF	48K	Reserved	Reserved	Reserved
00 0284 0000	00 0284 3FFF	16K	Reserved	Reserved	Reserved
00 0284 4000	00 0284 FFFF	48K	Reserved	Reserved	Reserved
00 0285 0000	00 0285 7FFF	32K	Reserved	Reserved	Reserved
00 0285 8000	00 0285 FFFF	32K	Reserved	Reserved	Reserved
00 0286 0000	00 0287 FFFF	128K	Reserved	Reserved	Reserved
00 0288 0000	00 0289 FFFF	128K	MESSAGEMANAGER_0_CFG1	MESSAGEMANAGER_0_CFG1	MESSAGEMANAGER_0_CFG1
00 028A 0000	00 028B FFFF	128K	MESSAGEMANAGER_0_CFG2	MESSAGEMANAGER_0_CFG2	MESSAGEMANAGER_0_CFG2
00 028C 0000	00 028C 0FFF	4K	MESSAGEMANAGER_0_CFG0	MESSAGEMANAGER_0_CFG0	MESSAGEMANAGER_0_CFG0

Table 2-1. Device Memory Map (continued)

Physical 40-bit Address		Bytes	Arm Cortex-A15 View	C66X_COREPAC View	Other Device Masters View
Start	End				
00 028C 1000	00 028C 1FFF	4K	MESSAGEMANAGER_0_CFG4	MESSAGEMANAGER_0_CFG4	MESSAGEMANAGER_0_CFG4
00 028C 2000	00 028C 2FFF	4K	MESSAGEMANAGER_0_CFG5	MESSAGEMANAGER_0_CFG5	MESSAGEMANAGER_0_CFG5
00 028C 3000	00 028C 33FF	1K	MESSAGEMANAGER_0_CFG7	MESSAGEMANAGER_0_CFG7	MESSAGEMANAGER_0_CFG7
00 028C 3400	00 028C 35FF	512	MESSAGEMANAGER_0_CFG6	MESSAGEMANAGER_0_CFG6	MESSAGEMANAGER_0_CFG6
00 028C 3600	00 028F FFFF	243K - 512	Reserved	Reserved	Reserved
00 0290 0000	00 0293 FFFF	256K	PMMC_0_CFG	PMMC_0_CFG	PMMC_0_CFG
00 0294 0000	00 0294 0FFF	4K	QSPI_0_CFG	QSPI_0_CFG	QSPI_0_CFG
00 0294 1000	00 029F FFFF	764K	Reserved	Reserved	Reserved
00 02A0 0000	00 02BF FFFF	2M	MESSAGEMANAGER_0_CFG3	MESSAGEMANAGER_0_CFG3	MESSAGEMANAGER_0_CFG3
00 02C0 0000	00 02C3 FFFF	256K	Reserved	Reserved	Reserved
00 02C4 0000	00 02C5 FFFF	128K	Reserved	Reserved	Reserved
00 02C6 0000	00 02C7 FFFF	128K	Reserved	Reserved	Reserved
00 02C8 0000	00 02CB FFFF	256K	Reserved	Reserved	Reserved
00 02CC 0000	00 02CD FFFF	128K	Reserved	Reserved	Reserved
00 02CE 0000	00 02EF FFFF	3M - 896K	Reserved	Reserved	Reserved
00 02F0 0000	00 02FF FFFF	1M	Reserved	Reserved	Reserved
00 0300 0000	00 030F FFFF	1M	DEBUG_CFG	DEBUG_CFG	DEBUG_CFG
00 0310 0000	00 03FF FFFF	15M	Reserved	Reserved	Reserved
00 0400 0000	00 04FF FFFF	16M	NSS_0_NAVSS	NSS_0_NAVSS	NSS_0_NAVSS
00 0500 0000	00 07FF FFFF	48M	Reserved	Reserved	Reserved
00 0800 0000	00 0801 FFFF	128K	Reserved	Reserved	XMC_CFG
00 0802 0000	00 0BBF FFFF	60M - 128K	Reserved	Reserved	Reserved
00 0BC0 0000	00 0BCF FFFF	1M	MSMC_CFG	MSMC_CFG	MSMC_CFG
00 0BD0 0000	00 0BFF FFFF	3M	Reserved	Reserved	Reserved
00 0C00 0000	00 0C0F FFFF	1M	MSMC_SRAM	MSMC_SRAM	MSMC_SRAM
00 0C10 0000	00 0C3F FFFF	3M	Reserved	Reserved	Reserved
00 0C40 0000	00 0C5F FFFF	2M	Reserved	Reserved	Reserved
00 0C60 0000	00 0FFF FFFF	58M	Reserved	Reserved	Reserved
00 1000 0000	00 107F FFFF	8M	Reserved	Reserved	Reserved
00 1080 0000	00 1087 FFFF	512K	C66X_COREPAC_0_L2_SRAM	C66X_COREPAC_0_L2_SRAM	C66X_COREPAC_0_L2_SRAM
00 1088 0000	00 108F FFFF	512K	C66X_COREPAC_0_L2_SRAM	C66X_COREPAC_0_L2_SRAM	C66X_COREPAC_0_L2_SRAM
00 1090 0000	00 10DF FFFF	5M	Reserved	Reserved	Reserved
00 10E0 0000	00 10E0 7FFF	32K	Reserved	Reserved	C66X_COREPAC_0_L1P_SRAM
00 10E0 8000	00 10EF FFFF	1M - 32K	Reserved	Reserved	Reserved
00 10F0 0000	00 10F0 7FFF	32K	Reserved	Reserved	INTERNAL_DSP0_L1D_SRAM_RAM
00 10F0 8000	00 117F FFFF	9M - 32K	Reserved	Reserved	Reserved
00 1180 0000	00 1187 FFFF	512K	Reserved	Reserved	Reserved
00 1188 0000	00 118F FFFF	512K	Reserved	Reserved	Reserved
00 1190 0000	00 11DF FFFF	5M	Reserved	Reserved	Reserved
00 11E0 0000	00 11E0 7FFF	32K	Reserved	Reserved	Reserved

Table 2-1. Device Memory Map (continued)

Physical 40-bit Address		Bytes	Arm Cortex-A15 View	C66X_COREPAC View	Other Device Masters View
Start	End				
00 11E0 8000	00 11EF FFFF	1M - 32K	Reserved	Reserved	Reserved
00 11F0 0000	00 11F0 7FFF	32K	Reserved	Reserved	Reserved
00 11F0 8000	00 11FF FFFF	1M - 32K	Reserved	Reserved	Reserved
00 1200 0000	00 127F FFFF	8M	Reserved	Reserved	Reserved
00 1280 0000	00 1287 FFFF	512K	Reserved	Reserved	Reserved
00 1288 0000	00 128F FFFF	512K	Reserved	Reserved	Reserved
00 1290 0000	00 12DF FFFF	5M	Reserved	Reserved	Reserved
00 12E0 0000	00 12E0 7FFF	32K	Reserved	Reserved	Reserved
00 12E0 8000	00 12EF FFFF	1M - 32K	Reserved	Reserved	Reserved
00 12F0 0000	00 12F0 7FFF	32K	Reserved	Reserved	Reserved
00 12F0 8000	00 12FF FFFF	1M - 32K	Reserved	Reserved	Reserved
00 1300 0000	00 137F FFFF	8M	Reserved	Reserved	Reserved
00 1380 0000	00 1387 FFFF	512K	Reserved	Reserved	Reserved
00 1388 0000	00 138F FFFF	512K	Reserved	Reserved	Reserved
00 1390 0000	00 13DF FFFF	5M	Reserved	Reserved	Reserved
00 13E0 0000	00 13E0 7FFF	32K	Reserved	Reserved	Reserved
00 13E0 8000	00 13EF FFFF	1M - 32K	Reserved	Reserved	Reserved
00 13F0 0000	00 13F0 7FFF	32K	Reserved	Reserved	Reserved
00 13F0 8000	00 147F FFFF	9M - 32K	Reserved	Reserved	Reserved
00 1480 0000	00 1487 FFFF	512K	Reserved	Reserved	Reserved
00 1488 0000	00 148F FFFF	512K	Reserved	Reserved	Reserved
00 1490 0000	00 14DF FFFF	5M	Reserved	Reserved	Reserved
00 14E0 0000	00 14E0 7FFF	32K	Reserved	Reserved	Reserved
00 14E0 8000	00 14EF FFFF	1M - 32K	Reserved	Reserved	Reserved
00 14F0 0000	00 14F0 7FFF	32K	Reserved	Reserved	Reserved
00 14F0 8000	00 157F FFFF	9M - 32K	Reserved	Reserved	Reserved
00 1580 0000	00 1587 FFFF	512K	Reserved	Reserved	Reserved
00 1588 0000	00 158F FFFF	512K	Reserved	Reserved	Reserved
00 1590 0000	00 15DF FFFF	5M	Reserved	Reserved	Reserved
00 15E0 0000	00 15E0 7FFF	32K	Reserved	Reserved	Reserved
00 15E0 8000	00 15EF FFFF	1M - 32K	Reserved	Reserved	Reserved
00 15F0 0000	00 15F0 7FFF	32K	Reserved	Reserved	Reserved
00 15F0 8000	00 167F FFFF	9M - 32K	Reserved	Reserved	Reserved
00 1680 0000	00 1687 FFFF	512K	Reserved	Reserved	Reserved
00 1688 0000	00 168F FFFF	512K	Reserved	Reserved	Reserved
00 1690 0000	00 16DF FFFF	5M	Reserved	Reserved	Reserved
00 16E0 0000	00 16E0 7FFF	32K	Reserved	Reserved	Reserved
00 16E0 8000	00 16EF FFFF	1M - 32K	Reserved	Reserved	Reserved
00 16F0 0000	00 16F0 7FFF	32K	Reserved	Reserved	Reserved
00 16F0 8000	00 177F FFFF	9M - 32K	Reserved	Reserved	Reserved
00 1780 0000	00 1787 FFFF	512K	Reserved	Reserved	Reserved
00 1788 0000	00 178F FFFF	512K	Reserved	Reserved	Reserved
00 1790 0000	00 17DF FFFF	5M	Reserved	Reserved	Reserved
00 17E0 0000	00 17E0 7FFF	32K	Reserved	Reserved	Reserved
00 17E0 8000	00 17EF FFFF	1M - 32K	Reserved	Reserved	Reserved
00 17F0 0000	00 17F0 7FFF	32K	Reserved	Reserved	Reserved

Table 2-1. Device Memory Map (continued)

Physical 40-bit Address		Bytes	Arm Cortex-A15 View	C66X_COREPAC View	Other Device Masters View
Start	End				
00 17F0 8000	00 1FFF FFFF	129M - 32K	Reserved	Reserved	Reserved
00 2000 0000	00 200F FFFF	1M	DBG_STM	DBG_STM	DBG_STM
00 2010 0000	00 2010 3FFF	16K	ASRC_0_S1	ASRC_0_S1	ASRC_0_S1
00 2010 0000	00 2010 3FFF	16K	Reserved	Reserved	Reserved
00 2010 4000	00 2010 43FF	1K	ASRC_0_S0	ASRC_0_S0	ASRC_0_S0
00 2010 4000	00 2010 43FF	1K	Reserved	Reserved	Reserved
00 2010 4400	00 201F FFFF	1M - 17K	Reserved	Reserved	Reserved
00 2020 0000	00 205F FFFF	4M	Reserved	Reserved	Reserved
00 2060 0000	00 206F FFFF	1M	DBG_TBR_SYS	DBG_TBR_SYS	DBG_TBR_SYS
00 2070 0000	00 2070 3FFF	16K	ARM_0_TBR_SYS	ARM_0_TBR_SYS	ARM_0_TBR_SYS
00 2070 4000	00 207F FFFF	1M - 16K	Reserved	Reserved	Reserved
00 2080 0000	00 2080 0FFF	4K	TETB_0_CFG	TETB_0_CFG	TETB_0_CFG
00 2080 1000	00 208F FFFF	1M - 4K	Reserved	Reserved	Reserved
00 2090 0000	00 209F FFFF	1M	M3_TBR_SYS	M3_TBR_SYS	M3_TBR_SYS
00 20A0 0000	00 20A3 FFFF	256K	Reserved	Reserved	Reserved
00 20A4 0000	00 20A4 FFFF	64K	Reserved	Reserved	Reserved
00 20A5 0000	00 20A7 FFFF	192K	Reserved	Reserved	Reserved
00 20A8 0000	00 20AB FFFF	256K	ICSS_0	ICSS_0	ICSS_0
00 20AC 0000	00 20AF FFFF	256K	ICSS_1	ICSS_1	ICSS_1
00 20B0 0000	00 20B3 FFFF	256K	DSP_ROM_0_SLV	DSP_ROM_0_SLV	DSP_ROM_0_SLV
00 20B4 0000	00 20BB FFFF	512K	Reserved	Reserved	Reserved
00 20BC 0000	00 20FF FFFF	5M - 768K	Reserved	Reserved	Reserved
00 2100 0000	00 2100 03FF	1K	Reserved	Reserved	Reserved
00 2100 0400	00 2100 05FF	512	Reserved	Reserved	Reserved
00 2100 0600	00 2100 07FF	512	Reserved	Reserved	Reserved
00 2100 0800	00 2100 09FF	512	Reserved	Reserved	Reserved
00 2100 0A00	00 2100 0BFF	512	Reserved	Reserved	Reserved
00 2100 0C00	00 2100 FFFF	61K	Reserved	Reserved	Reserved
00 2101 0000	00 2101 01FF	512	DDR_0_CFG	Reserved	DDR_0_CFG
00 2101 0200	00 2101 07FF	2K - 512	Reserved	Reserved	Reserved
00 2101 0800	00 2101 09FF	512	Reserved	Reserved	Reserved
00 2101 0A00	00 2101 0FFF	2K - 512	Reserved	Reserved	Reserved
00 2101 1000	00 2101 FFFF	60K	Reserved	Reserved	Reserved
00 2102 0000	00 2103 FFFF	128K	Reserved	Reserved	Reserved
00 2104 0000	00 213F FFFF	4M - 256K	Reserved	Reserved	Reserved
00 2140 0000	00 2140 00FF	256	Reserved	Reserved	Reserved
00 2140 0100	00 2140 01FF	256	Reserved	Reserved	Reserved
00 2140 0200	00 217F FFFF	4M - 512	Reserved	Reserved	Reserved
00 2180 0000	00 2180 3FFF	16K	PCIE_0_CFG	PCIE_0_CFG	PCIE_0_CFG
00 2180 4000	00 2180 40FF	256	MCASP_0_SLV	MCASP_0_SLV	MCASP_0_SLV
00 2180 4100	00 2180 43FF	768	Reserved	Reserved	Reserved
00 2180 4400	00 2180 44FF	256	MCASP_1_SLV	MCASP_1_SLV	MCASP_1_SLV
00 2180 4500	00 2180 47FF	768	Reserved	Reserved	Reserved
00 2180 4800	00 2180 48FF	256	MCASP_2_SLV	MCASP_2_SLV	MCASP_2_SLV
00 2180 4900	00 2180 4BFF	768	Reserved	Reserved	Reserved
00 2180 4C00	00 2180 4DFF	512	Reserved	Reserved	Reserved

Table 2-1. Device Memory Map (continued)

Physical 40-bit Address		Bytes	Arm Cortex-A15 View	C66X_COREPAC View	Other Device Masters View
Start	End				
00 2180 4E00	00 2180 4FFF	512	Reserved	Reserved	Reserved
00 2180 5000	00 2180 50FF	256	MCBSP_0_SLV	MCBSP_0_SLV	MCBSP_0_SLV
00 2180 5100	00 2180 53FF	768	Reserved	Reserved	Reserved
00 2180 5400	00 2180 55FF	512	SPI_0_SLV	SPI_0_SLV	SPI_0_SLV
00 2180 5600	00 2180 57FF	512	Reserved	Reserved	Reserved
00 2180 5800	00 2180 59FF	512	SPI_1_SLV	SPI_1_SLV	SPI_1_SLV
00 2180 5A00	00 2180 5BFF	512	Reserved	Reserved	Reserved
00 2180 5C00	00 2180 5DFF	512	SPI_2_SLV	SPI_2_SLV	SPI_2_SLV
00 2180 5E00	00 2180 5FFF	512	Reserved	Reserved	Reserved
00 2180 6000	00 2180 61FF	512	SPI_3_SLV	SPI_3_SLV	SPI_3_SLV
00 2180 6200	00 2180 63FF	512	Reserved	Reserved	Reserved
00 2180 6400	00 2180 7FFF	7K	Reserved	Reserved	Reserved
00 2181 8000	00 2181 83FF	1K	GPMC_0_CFG	GPMC_0_CFG	GPMC_0_CFG
00 2181 8400	00 21BF FFFF	4M - 97K	Reserved	Reserved	Reserved
00 21C0 0000	00 21FF FFFF	4M	Reserved	Reserved	Reserved
00 2200 0000	00 229F FFFF	10M	Reserved	Reserved	Reserved
00 22A0 0000	00 22A0 FFFF	64K	Reserved	Reserved	Reserved
00 22A1 0000	00 22AF FFFF	1M - 64K	Reserved	Reserved	Reserved
00 22B0 0000	00 22B0 FFFF	64K	Reserved	Reserved	Reserved
00 22B1 0000	00 22BF FFFF	1M - 64K	Reserved	Reserved	Reserved
00 22C0 0000	00 22C0 FFFF	64K	Reserved	Reserved	Reserved
00 22C1 0000	00 22CF FFFF	1M - 64K	Reserved	Reserved	Reserved
00 22D0 0000	00 22D0 FFFF	64K	Reserved	Reserved	Reserved
00 22D1 0000	00 22DF FFFF	1M - 64K	Reserved	Reserved	Reserved
00 22E0 0000	00 22E0 FFFF	64K	Reserved	Reserved	Reserved
00 22E1 0000	00 22EF FFFF	1M - 64K	Reserved	Reserved	Reserved
00 22F0 0000	00 22F0 FFFF	64K	Reserved	Reserved	Reserved
00 22F1 0000	00 22FF FFFF	1M - 64K	Reserved	Reserved	Reserved
00 2300 0000	00 2300 FFFF	64K	MMCSD_0_S	MMCSD_0_S	MMCSD_0_S
00 2301 0000	00 230F FFFF	1M - 64K	Reserved	Reserved	Reserved
00 2310 0000	00 2310 FFFF	64K	MMCSD_1_S	MMCSD_1_S	MMCSD_1_S
00 2311 0000	00 231F FFFF	1M - 64K	Reserved	Reserved	Reserved
00 2320 0000	00 2324 FFFF	320K	Reserved	Reserved	Reserved
00 2325 0000	00 239F FFFF	8M - 320K	Reserved	Reserved	Reserved
00 23A0 0000	00 23AF FFFF	1M	Reserved	Reserved	Reserved
00 23B0 0000	00 23BF FFFF	1M	Reserved	Reserved	Reserved
00 23C0 0000	00 23FF FFFF	4M	Reserved	Reserved	Reserved
00 2400 0000	00 27FF FFFF	64M	QSPI_0_DATA	QSPI_0_DATA	QSPI_0_DATA
00 2800 0000	00 2FFF FFFF	128M	DXB_0_S	DXB_0_S	DXB_0_S
00 3000 0000	00 3FFF FFFF	256M	GPMC_0_DATA	GPMC_0_DATA	GPMC_0_DATA
00 4000 0000	00 40FF FFFF	16M	Reserved	Reserved	Reserved
00 4100 0000	00 4FFF FFFF	240M	Reserved	Reserved	Reserved
00 5000 0000	00 50FF FFFF	16M	Reserved	Reserved	Reserved
00 5100 0000	00 51FF FFFF	16M	Reserved	Reserved	Reserved
00 5200 0000	00 52FF FFFF	16M	Reserved	Reserved	Reserved
00 5300 0000	00 53FF FFFF	16M	Reserved	Reserved	Reserved

Table 2-1. Device Memory Map (continued)

Physical 40-bit Address		Bytes	Arm Cortex-A15 View	C66X_COREPAC View	Other Device Masters View
Start	End				
00 5400 0000	00 54FF FFFF	16M	Reserved	Reserved	Reserved
00 5500 0000	00 55FF FFFF	16M	Reserved	Reserved	Reserved
00 5600 0000	00 56FF FFFF	16M	Reserved	Reserved	Reserved
00 5700 0000	00 57FF FFFF	16M	Reserved	Reserved	Reserved
00 5800 0000	00 58FF FFFF	16M	Reserved	Reserved	Reserved
00 5900 0000	00 59FF FFFF	16M	Reserved	Reserved	Reserved
00 5A00 0000	00 5AFF FFFF	16M	Reserved	Reserved	Reserved
00 5B00 0000	00 5BFF FFFF	16M	Reserved	Reserved	Reserved
00 5C00 0000	00 5CFF FFFF	16M	Reserved	Reserved	Reserved
00 5D00 0000	00 5DFF FFFF	16M	Reserved	Reserved	Reserved
00 5E00 0000	00 5EFF FFFF	16M	Reserved	Reserved	Reserved
00 5F00 0000	00 5FFF FFFF	16M	Reserved	Reserved	Reserved
00 6000 0000	00 63FF FFFF	64M	Reserved	Reserved	Reserved
00 6400 0000	00 67FF FFFF	64M	Reserved	Reserved	Reserved
00 6800 0000	00 6BFF FFFF	64M	Reserved	Reserved	Reserved
00 6C00 0000	00 6FFF FFFF	64M	Reserved	Reserved	Reserved
00 7000 0000	00 7FFF FFFF	256M	PCIE_0_DATA	PCIE_0_DATA	PCIE_0_DATA
00 8000 0000	00 FFFF FFFF	2G	DDR_0_DATA(2)	Reserved	DDR_0_DATA(3)
01 0000 0000	01 2100 FFFF	528M+64K	Reserved	Reserved	Reserved
01 2101 0000	01 2101 01FF	512K	DDR_0_CFG(4)	DDR_0_CFG(5)	DDR_0_CFG(6)
01 2101 0200	07 FFFF FFFF	32G-512	Reserved	Reserved	Reserved
08 0000 0000	09 FFFF FFFF	8G	DDR_0_DATA(7)	DDR_0_DATA(8)	DDR_0_DATA(9)
0A 0000 0000	FF FFFF FFFF	984G	Reserved	Reserved	Reserved

(2) The first 2GB of DDR_0_DATA. This region is alias of 08 0000 0000 - 08 7FFF FFFF. No IO coherency is supported.
(3) DDR_0_DATA (mapped to MSMC SES port). IO coherency is supported.
(4) This region is aliased to 00 2101 0000 - 00 2101 01FF.
(5) Access to 40-bit address requires XMC MPAX programming.
(6) Access to 40-bit address requires MSMC MPAX programming. MPAX from SES port need to re-map the region of 00 2101 0000 - 00 2101 01FF to this region.
(7) 8GB DDR_0_DATA data region. IO coherency is supported.
(8) 8GB DDR_0_DATA data region. Access to 40-bit address requires XMC MPAX programming.
(9) 8GB DDR_0_DATA data region. Access to 40-bit address requires MSMC MPAX programming. MPAX for SES port needs to re-map the 2GB address region 00 8000 0000 - 00 FFFF FFFF to 08 0000 0000 - 09 FFFF FFFF.

Interconnect

This chapter describes the System Interconnect and the Memory Protection Units (MPU) for the device.

Topic **Page**

3.1 System Interconnect

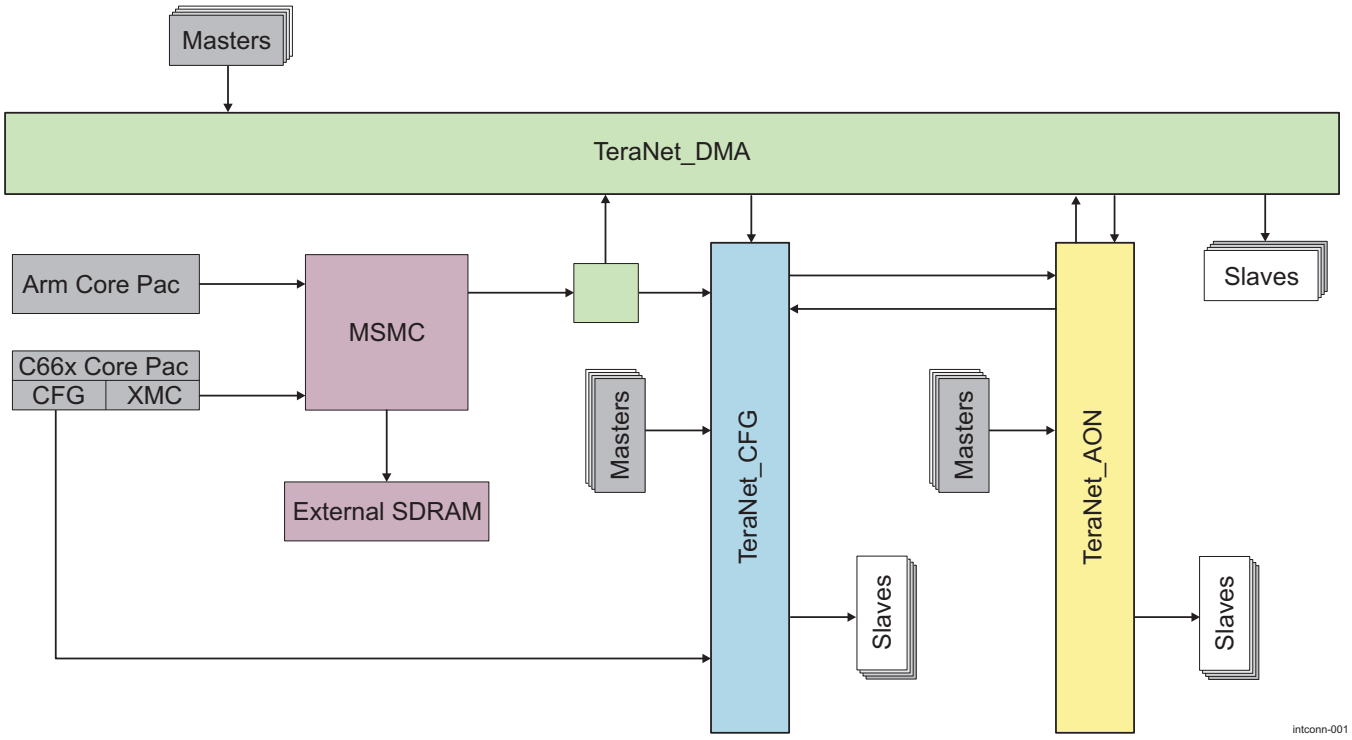
The following sections describe the device system interconnect.

3.1.1 System Interconnect Overview

All modules and subsystems in the device communicate with each other through the system interconnect. It is partitioned into three sections: main data (TeraNet_DMA), main configuration (TeraNet_CFG) and always on (TeraNet_AON) interconnect. Each of these three sections is composed by several TeraNets, which are non-blocking switch fabrics enabling fast and contention-free internal data movement. They also provide low-latency and concurrent data transfers between master and slave peripherals.

Figure 3-1 shows an overview of the device system interconnect. All modules and subsystems can be classified into two categories: masters and slaves. The masters are capable of initiating read and write transfers in the system and do not depend on the EDMA for their data transfers. The slaves on the other hand depend on the masters to perform transfers to and from them. They cannot generate read/write requests but can respond to these requests generating interrupts or DMA requests.

Figure 3-1. System Interconnect Overview



3.1.2 System Interconnect Integration

Table 3-1 and Table 3-2 summarize the integration of the system interconnect in the device.

Table 3-1. System Interconnect Integration Attributes

Module Instance	Attributes	
	Power Domain	Module Domain
TeraNet_DMA	PD4	LPSC5
TeraNet_CFG	PD4	LPSC5
TeraNet_AON	PD0	LPSC0

Table 3-2. System Interconnect Clocks and Resets

Clocks			
TeraNet Component	**Clock Signal**	**Source**	**Description**
TeraNet_M_1_256_0 TeraNet_M_1_256_1 TeraNet_M_1_256_2	CHIP_CLK1	PLL Controller	Clock to some components of TeraNet_DMA which frequency is equal to the CHIP_CLK1 clock.
TeraNet_M_3_128_0 TeraNet_M_3_128_1 TeraNet_M_3_128_2 TeraNet_M_3_128_3 TeraNet_M_3_128_4 TeraNet_M_3_32_0 TeraNet_P_3_32_1 TeraNet_P_3_32_3 TeraNet_P_3_32_5 TeraNet_P_3_32_6 TeraNet_P_3_32_7 TeraNet_P_3_32_8 TeraNet_P_3_32_9 TeraNet_P_3_32_10 TeraNet_P_3_32_11	CHIP_CLK1/3	PLL Controller	Clock to some components of TeraNet_DMA and TeraNet_CFG which frequency is equal to the CHIP_CLK1 clock divided by 3. This clock is also propagated to the MPUs.
TeraNet_P_6_32_0 TeraNet_P_6_32_1 TeraNet_P_6_32_2 TeraNet_P_6_32_3 TeraNet_P_6_32_4 TeraNet_P_6_32_5 TeraNet_P_6_32_6 TeraNet_P_6_32_7	CHIP_CLK1/6	PLL Controller	Clock to some components of TeraNet_DMA, TeraNet_CFG and TeraNet_AON which frequency is equal to the CHIP_CLK1 clock divided by 6.
Resets			
Module	**Reset Signal**	**Source**	**Description**
System Interconnect	MOD_G_RST	LPSC5	Reset to the bridges (for example, BR_MP_1, BR_OV_3, and so on) and TeraNet components.
	MOD_G_RST	LPSC0	Reset to the always on logic.

3.1.3 System Interconnect Functional Description

TeraNet_DMA includes most of the master ports and all the data slave end points. Figure 3-2 shows the connections between masters and slaves through the various sections of TeraNet_DMA.

Figure 3-2. TeraNet_DMA Master-Slave Connections

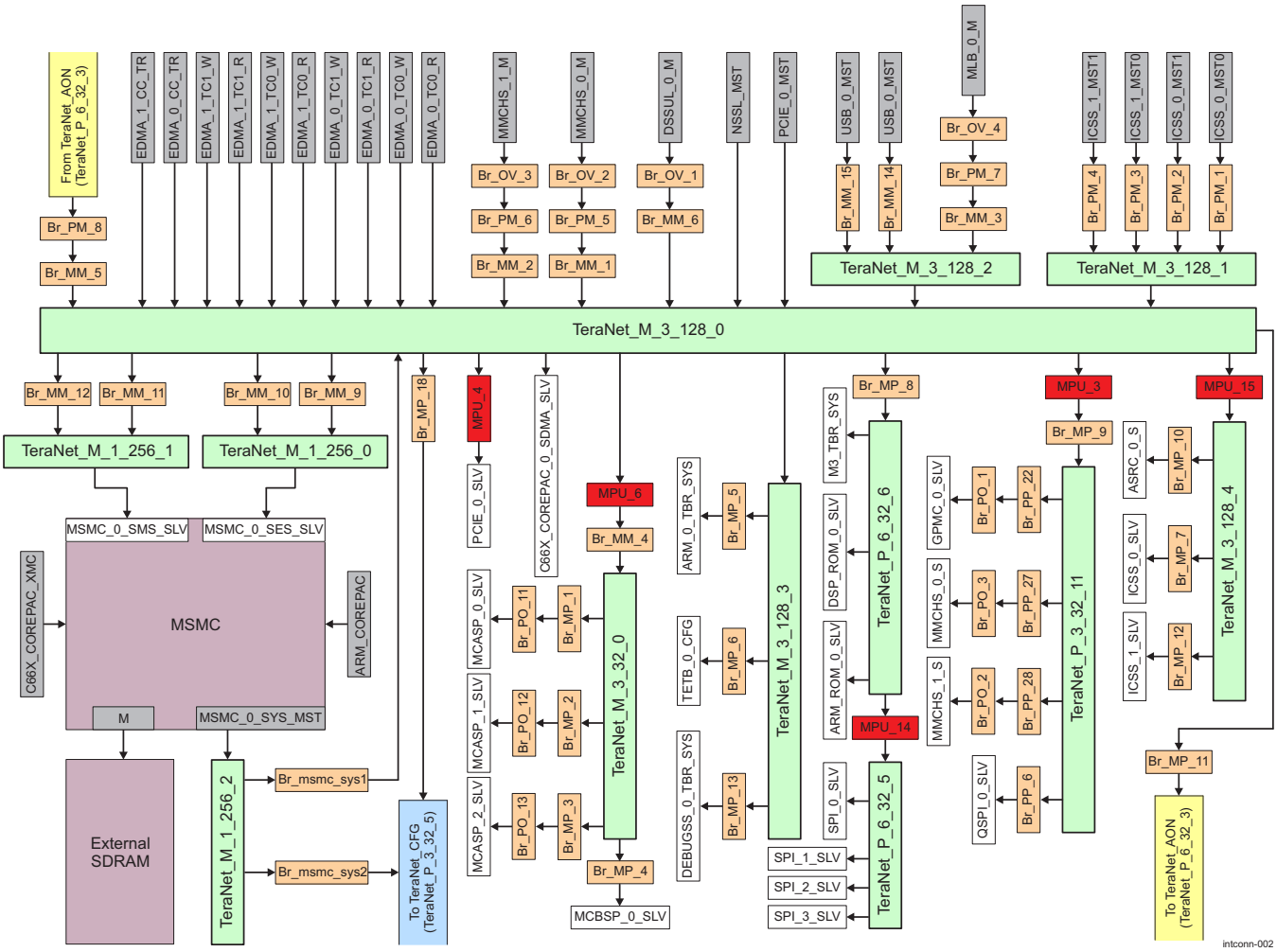


Table 3-3 lists the master and slave end point connections on TeraNet_DMA. A cell may contain one of the following:

- Y – There is a connection between this master and that slave
- N – There is no connection between this master and that slave.

Table 3-3. TeraNet_DMA Connectivity Matrix

Masters	Slaves on TeraNet_DMA																								
	ASRC_0_S	DEBUGSS_0_TBR_SYS	C66X_COREPAC_0_SDMA_SLV	MSMC_0_SES_SLV	MSMC_0_SMS_SLV	PCIE_0_SLV	GPMC_0_SLV	TETB_0_CFG	SPI_0_SLV	SPI_1_SLV	SPI_2_SLV	SPI_3_SLV	ARM_0_TBR_SYS	M3_TBR_SYS	MCASP_0_SLV	MCASP_1_SLV	MCASP_2_SLV	MCBSP_0_SLV	MMCHS_0_S	MMCHS_1_S	DSP_ROM_0_SLV	ARM_ROM_0_SLV	QSPI_0_SLV	ICSS_0_SLV	ICSS_1_SLV
EDMA_0_TC0_R	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_0_TC0_W	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_0_TC1_R	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_0_TC1_W	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_1_TC0_R	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_1_TC0_W	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_1_TC1_R	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_1_TC1_W	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
MSMC_0_SYS_MST	Y	Y	Y	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
PCIE_0_MST	N	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	Y	Y	Y
USB_0_MST	N	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	Y	N	N
USB_1_MST	N	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	Y	N	N
MLB_0_M	N	N	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
MMCHS_0_M	N	N	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
MMCHS_1_M	N	N	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
ICSS_0_MST0	N	N	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	N	N	Y	Y	Y	Y	Y	Y	N	N	Y	Y	Y
ICSS_1_MST0	N	N	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	N	N	Y	Y	Y	Y	Y	Y	N	N	Y	Y	Y
ICSS_0_MST1	N	N	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	N	N	Y	Y	Y	Y	Y	Y	N	N	Y	Y	Y
ICSS_1_MST1	N	N	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	N	N	Y	Y	Y	Y	Y	Y	N	N	Y	Y	Y
NSSL_MST	N	N	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
DSSUL_0_M	N	N	Y	Y	Y	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
EDMA_0_CC_TR	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
EDMA_1_CC_TR	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
C66X_COREPAC_0_CFG	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
CPT_(0-14)_M	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
DEBUGSS_0_DAP_MST	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
PMMC_0_M	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	N	N	Y	Y	Y	Y	Y	Y	N	N	Y	Y	Y

Color Code:	Masters on TeraNet_DMA	Masters on TeraNet_CFG	Masters on TeraNet_AON

The masters on the TeraNet_DMA pass through various bridges on the way to the MSMC slave ports (SMS and SES) and other TeraNets (TeraNet_AON and TeraNet_CFG). Table 3-4 lists the connections between the TeraNet_DMA masters and various bridges. This table gives the user the flexibility to select masters (typically EDMA) that use the same or different bridges. A cell may contain one of the following:

- Y – There is a connection between this master and the bridge
- N – There is no connection between this master and the bridge.

Table 3-4. TeraNet_DMA Bridge Connectivity Matrix

Masters	Br_MM_9 (to MSMC_0_SES_SLV)	Br_MM_10 (to MSMC_0_SES_SLV)	Br_MM_11 (to MSMC_0_SES_SLV)	Br_MM_12 (to MSMC_0_SES_SLV)	Br_MP_11 (to MSMC_0_SES_SLV)	Br_MP_18 (to MSMC_0_SES_SLV)
EDMA_0_TC0_R	Y	N	Y	N	Y	Y
EDMA_0_TC0_W	Y	N	Y	N	Y	Y
EDMA_0_TC1_R	N	Y	N	Y	Y	Y
EDMA_0_TC1_W	N	Y	N	Y	Y	Y
EDMA_1_TC0_R	Y	N	Y	N	Y	Y
EDMA_1_TC0_W	Y	N	Y	N	Y	Y
EDMA_1_TC1_R	N	Y	N	Y	Y	Y
EDMA_1_TC1_W	N	Y	N	Y	Y	Y
PCIE_0_MST	N	Y	N	Y	Y	Y
USB_0_MST (via TeraNet_M_3_128_2)	Y	N	Y	N	Y	Y
USB_1_MST (via TeraNet_M_3_128_2)	Y	N	Y	N	Y	Y
MLB_0_M (via TeraNet_M_3_128_2)	Y	N	Y	N	Y	Y
MMCHS_0_M (via Br_MM_1)	N	Y	N	Y	Y	N
MMCHS_1_M (via Br_MM_2)	N	Y	N	Y	Y	N
ICSS_0_MST0 (via TeraNet_M_3_128_1)	N	Y	N	Y	Y	N
ICSS_0_MST1 (via TeraNet_M_3_128_1)	N	Y	N	Y	Y	N
ICSS_1_MST0 (via TeraNet_M_3_128_1)	N	Y	N	Y	Y	N
ICSS_1_MST1 (via TeraNet_M_3_128_1)	N	Y	N	Y	Y	N
NSSL_MST	Y	N	Y	N	N	N
DSSUL_0_M (via Br_MM_6)	N	Y	N	Y	N	N
Br_MM_5 (from TeraNet_AON)	Y	N	Y	N	N	N

TeraNet_CFG is intended to include almost all of the configuration end points. All the configuration accesses to the slave end points need to be 32-bit address aligned. 8-bit and 16-bit accesses are not supported. Figure 3-3 shows the connections between masters and slaves through the various sections of TeraNet_CFG.

It should be taken into account that some peripherals have both data bus and configuration bus interface, while others only have one type of interface. In addition, the bus interface, width and speed varies from peripheral to peripheral.

NOTE: ELM is not supported on this family of devices.

Figure 3-3. TeraNet_CFG Master-Slave Connections

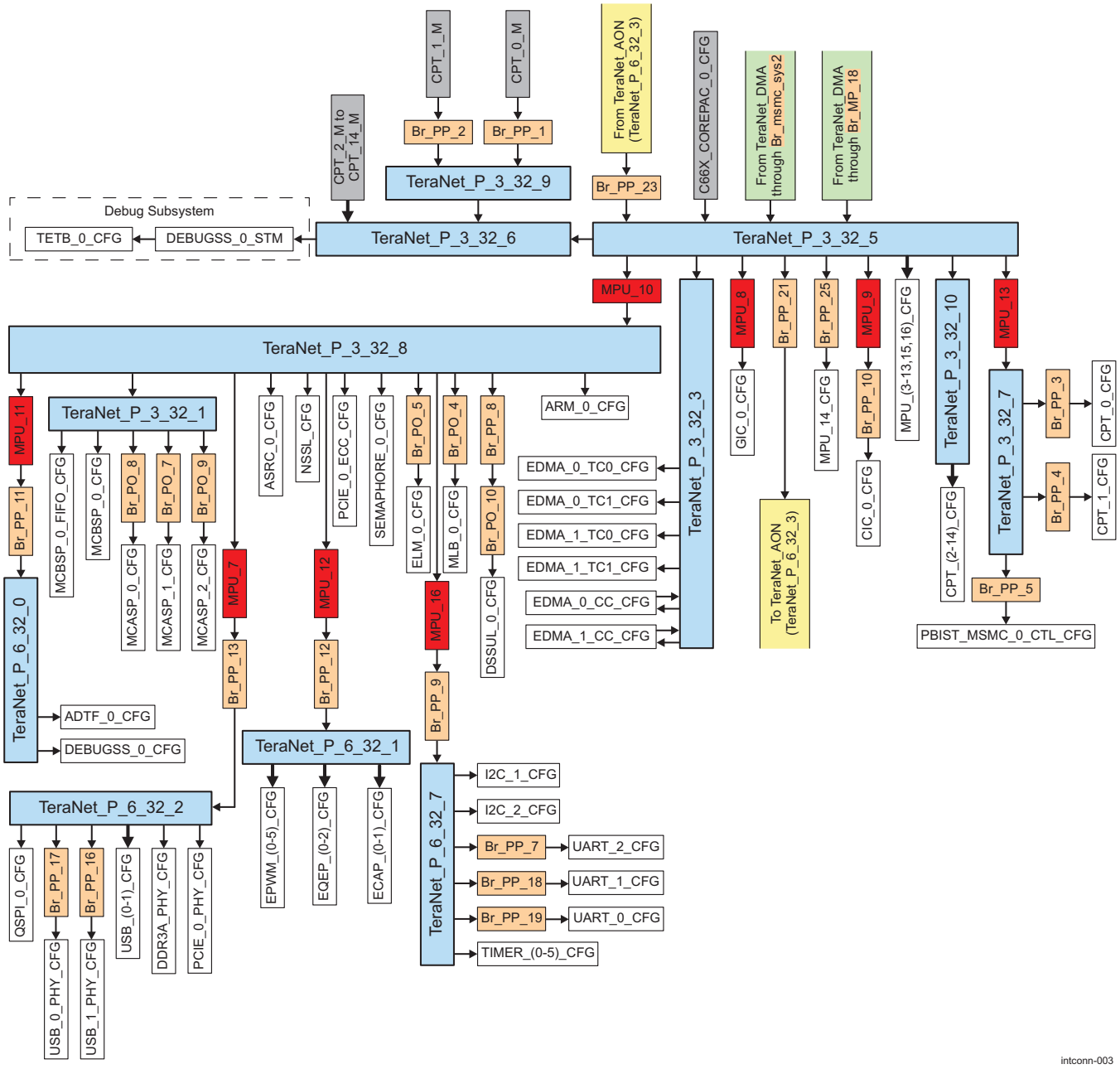


Table 3-5 and Table 3-6 list the master and slave end point connections on TeraNet_CFG. A cell may contain one of the following:

- Y – There is a connection between this master and that slave
- N – There is no connection between this master and that slave.

Table 3-5. TeraNet_CFG Connectivity Matrix (Part 1)

	Slaves on TeraNet_CFG																					
Masters	UART_0_CFG	UART_1_CFG	UART_2_CFG	NSSL_CFG	ELM_0_CFG	ASRC_0_CFG	QSPI_0_CFG	DEBUGSS_0_CFG	DEBUGSS_0_STM	PBIST_MSMC_0_CTL_CFG	CIC_0_CFG	PCIE_0_PHY_CFG	PCIE_0_ECC_CFG	SEMAPHORE_0_CFG	TIMER_(0-5)_CFG	CPT_(0-14)_CFG	DDR3A_PHY_CFG	ADTF_0_CFG	I2C_1_CFG	I2C_2_CFG	MPU_(3-16)_CFG	ARM_0_CFG
EDMA_0_TC0_R	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_0_TC0_W	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_0_TC1_R	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_0_TC1_W	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_1_TC0_R	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_1_TC0_W	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_1_TC1_R	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_1_TC1_W	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
MSMC_0_SYS_MST	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
PCIE_0_MST	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
USB_0_MST	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
USB_1_MST	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
MLB_0_M	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
MMCHS_0_M	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
MMCHS_1_M	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
ICSS_0_MST0	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ICSS_1_MST0	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ICSS_0_MST1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ICSS_1_MST1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
NSSL_MST	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
DSSUL_0_M	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
EDMA_0_CC_TR	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
EDMA_1_CC_TR	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
C66X_COREPAC_0_CFG	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CPT_(0-14)_M	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N
DEBUGSS_0_DAP_MST	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
PMMC_0_M	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Color Code:	Masters on TeraNet_DMA	Masters on TeraNet_CFG	Masters on TeraNet_AON

Table 3-6. TeraNet_CFG Connectivity Matrix (Part 2)

	Slaves on TeraNet_CFG																				
Masters	GIC_0_CFG	USB_0_CFG	USB_1_CFG	USB_0_PHY_CFG	USB_1_PHY_CFG	EPWM_(0-5)_CFG	EQEP_(0-2)_CFG	ECAP_(0-1)_CFG	MCASP_0_CFG	MCASP_1_CFG	MCASP_2_CFG	MCBSP_0_CFG	MCBSP_0_FIFO_CFG	MLB_0_CFG	EDMA_0_CC_CFG	EDMA_1_CC_CFG	EDMA_0_TC0_CFG	EDMA_0_TC1_CFG	EDMA_1_TC0_CFG	EDMA_1_TC1_CFG	DSSUL_0_CFG
EDMA_0_TC0_R	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_0_TC0_W	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_0_TC1_R	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_0_TC1_W	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_1_TC0_R	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_1_TC0_W	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_1_TC1_R	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_1_TC1_W	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
MSMC_0_SYS_MST	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
PCIE_0_MST	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
USB_0_MST	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
USB_1_MST	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
MLB_0_M	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
MMCHS_0_M	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
MMCHS_1_M	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
ICSS_0_MST0	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ICSS_1_MST0	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ICSS_0_MST1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ICSS_1_MST1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
NSSL_MST	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
DSSUL_0_M	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
EDMA_0_CC_TR	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	Y	N	N	N
EDMA_1_CC_TR	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	Y	N
C66X_COREPAC_0 _CFG	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CPT_(0-14)_M	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
DEBUGSS_0_DAP _MST	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
PMMC_0_M	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Color Code:	Masters on TeraNet_DMA	Masters on TeraNet_CFG	Masters on TeraNet_AON

Figure 3-4. TeraNet_AON Master-Slave Connections

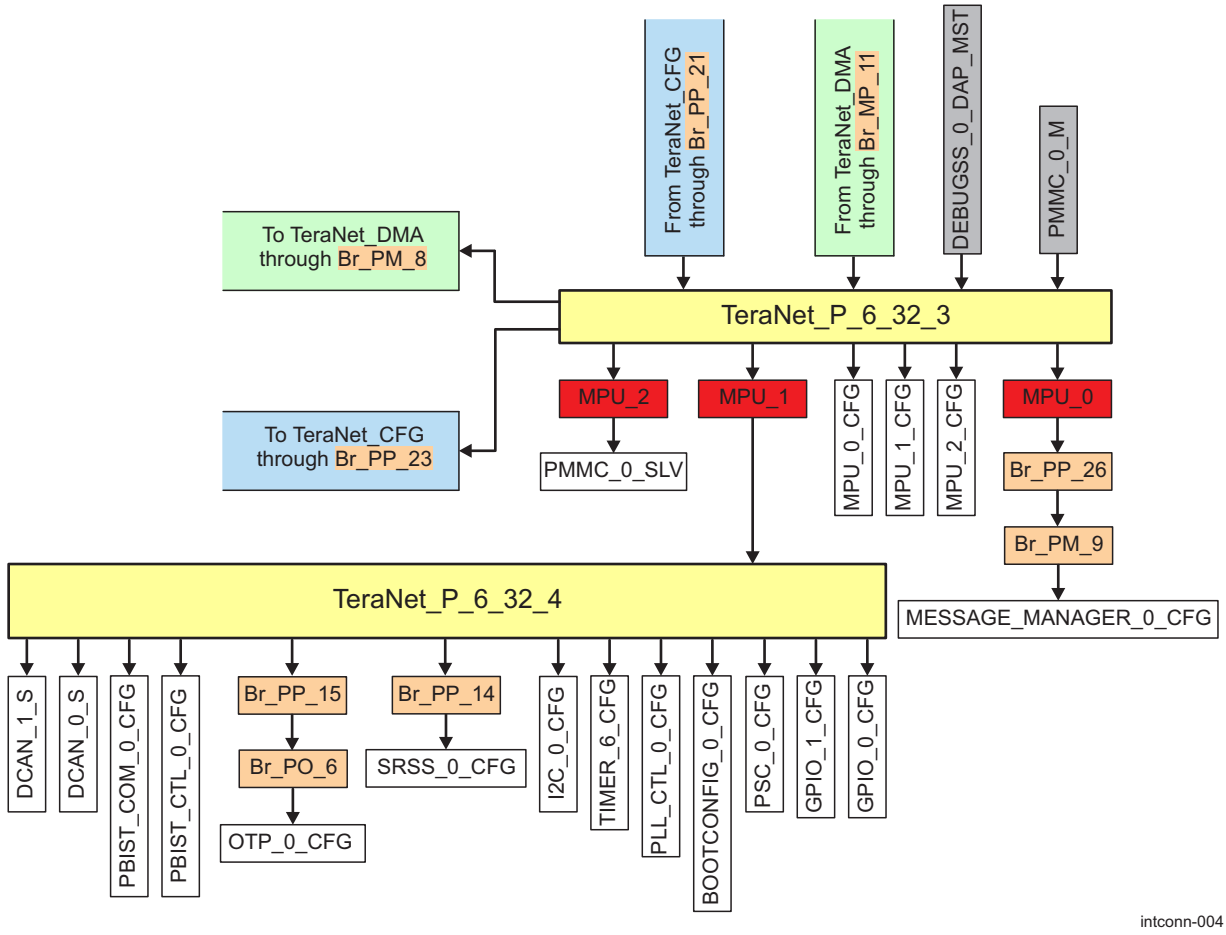


Table 3-7 lists the master and slave end point connections on TeraNet_AON. A cell may contain one of the following:

- Y – There is a connection between this master and that slave
- N – There is no connection between this master and that slave.

Table 3-7. TeraNet_AON Connectivity Matrix

	Slaves on TeraNet_AON																	
Masters	PLL_CTL_0_CFG	PSC_0_CFG	BOOTCONFIG_0_CFG	DCAN_0_S	DCAN_1_S	GPIO_0_CFG	GPIO_1_CFG	PMMC_0_SLV	PBIST_COM_0_CFG	PBIST_CTL_0_CFG	TIMER_6_CFG	I2C_0_CFG	MPU_0_CFG	MPU_1_CFG	MPU_2_CFG	SRSS_0_CFG	OTP_0_CFG	MESSAGE_MANAGER_0_CFG
EDMA_0_TC0_R	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_0_TC0_W	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_0_TC1_R	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_0_TC1_W	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_1_TC0_R	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Table 3-7. TeraNet_AON Connectivity Matrix (continued)

	Slaves on TeraNet_AON																	
EDMA_1_TC0_W	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_1_TC1_R	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
EDMA_1_TC1_W	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
MSMC_0_SYS_MST	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
PCIE_0_MST	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
USB_0_MST	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
USB_1_MST	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
MLB_0_M	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
MMCHS_0_M	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
MMCHS_1_M	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
ICSS_0_MST0	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ICSS_1_MST0	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ICSS_0_MST1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ICSS_1_MST1	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
NSSL_MST	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
DSSUL_0_M	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
EDMA_0_CC_TR	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
EDMA_1_CC_TR	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
C66X_COREPAC_0_CFG	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CPT_(0-14)_M	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
DEBUGSS_0_DAP_MST	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
PMMC_0_M	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Color Code:	Masters on TeraNet_DMA	Masters on TeraNet_CFG	Masters on TeraNet_AON

The MPUs (Memory Protection Units) shown on the "*Master-Slave Connections*" figures are used to prevent unauthorized access to or from a module. For more information about the device MPUs see Section 3.2, *Memory Protection Units*.

3.1.3.1 Transaction Priority for the TeraNet Masters

The transaction priority of certain masters on the TeraNet is controlled by registers which reside in the BOOT_CFG module. For more information see Section 5.1.3.1.4, *TeraNet Priority Registers*.

3.2 Memory Protection Units (MPU)

This section describes the Memory Protection Units (MPU) for the device.

3.2.1 MPU Overview

The Memory Protection Unit (MPU) protects data and configuration spaces by managing the access to these memories. It allows multiple ranges of protection with different access permissions to be defined. Each range can be made secure by limiting the access only to secure system masters. Access can also be granted or not to individual system masters based on their privilege ID or access types. The MPU can also record a detected fault or invalid access and notify the system through an interrupt.

The MPU has the following features:

- Supports multiple programmable address ranges
- Supports secure and debug access privileges
- Supports read, write, and execute access privileges
- Supports privilege ID associations with the programmable address ranges
- Generates an interrupt when there is addressing or protection violation and saves violating transfer parameters
- Supports L1/L2 cache accesses
- Supports protection of its own registers.

3.2.2 MPU Integration

This section describes the MPU modules integration in the device including information about clocks, resets, and hardware requests.

Figure 3-5 shows the integration of the MPU modules in the device.

Figure 3-5. MPU Integration

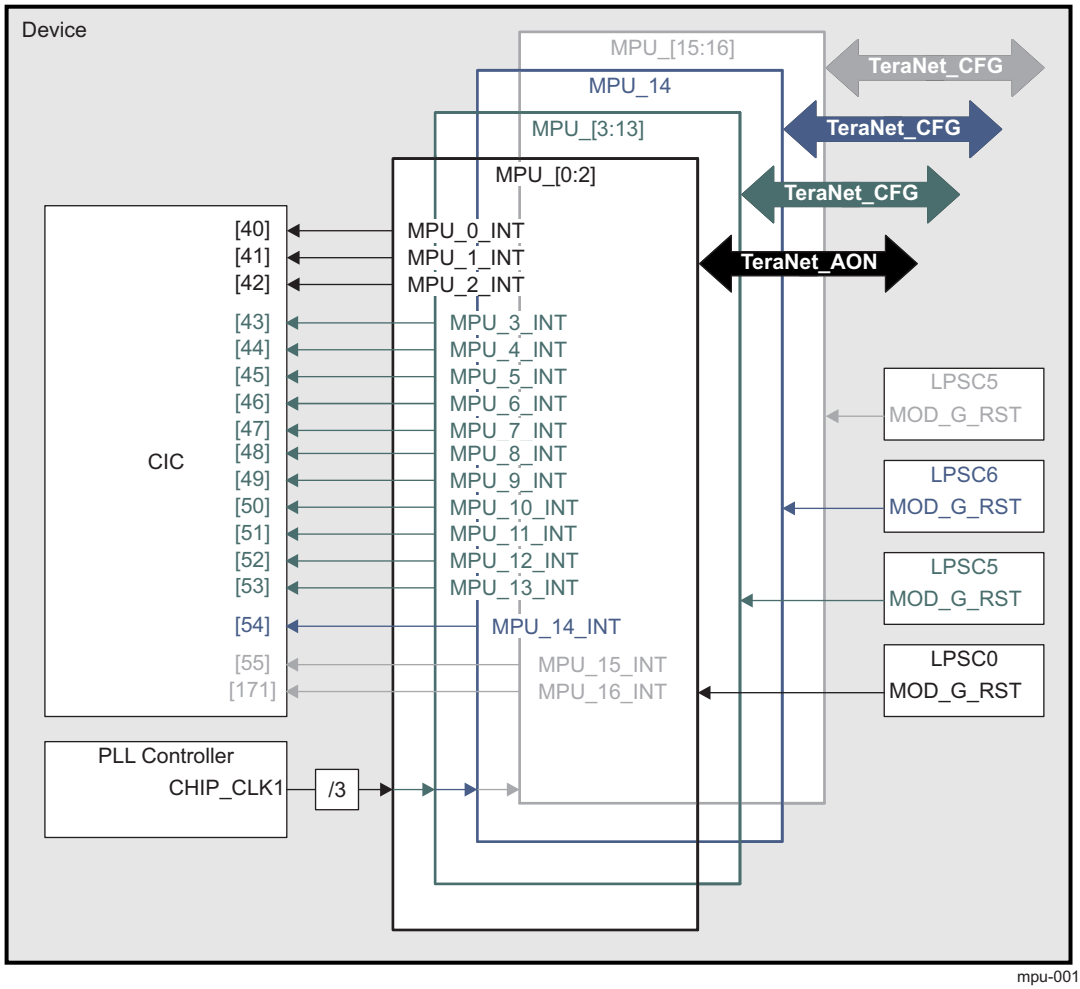


Table 3-8 through Table 3-10 summarize the integration of the MPU modules in the device.

Table 3-8. MPU Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
MPU_[0:2]	PD0	LPSC0	TeraNet_AON
MPU_[3:13]	PD4	LPSC5	TeraNet_CFG
MPU_14	PD5	LPSC6	TeraNet_CFG
MPU_[15:16]	PD4	LPSC5	TeraNet_CFG

Table 3-9. MPU Clocks and Resets

Clocks			
Module Instance	**Clock Signal**	**Source**	**Description**
MPU_[0:16]	CHIP_CLK1/3	PLL Controller	Clock for all MPU modules with frequency equal to CHIP_CLK1 divided by 3
Resets			
Module Instance	**Reset Signal**	**Source**	**Description**
MPU_[0:2]	MOD_G_RST	LPSC0	MPU_[0:2] module reset
MPU_[3:13]	MOD_G_RST	LPSC5	MPU_[3:13] module reset
MPU_14	MOD_G_RST	LPSC6	MPU_14 module reset
MPU_[15:16]	MOD_G_RST	LPSC5	MPU_[15:16] module reset

Table 3-10. MPU Hardware Requests

Interrupt Requests			
Module Instance	**Event Name**	**Mapped To Input Event [Number]**	**Description**
		CIC	
MPU_0	MPU_0_INT	[40]	MPU_0 interrupt indicating addressing violation or protection violation error
MPU_1	MPU_1_INT	[41]	MPU_1 interrupt indicating addressing violation or protection violation error
MPU_2	MPU_2_INT	[42]	MPU_2 interrupt indicating addressing violation or protection violation error
MPU_3	MPU_3_INT	[43]	MPU_3 interrupt indicating addressing violation or protection violation error
MPU_4	MPU_4_INT	[44]	MPU_4 interrupt indicating addressing violation or protection violation error
MPU_5	MPU_5_INT	[45]	MPU_5 interrupt indicating addressing violation or protection violation error
MPU_6	MPU_6_INT	[46]	MPU_6 interrupt indicating addressing violation or protection violation error
MPU_7	MPU_7_INT	[47]	MPU_7 interrupt indicating addressing violation or protection violation error
MPU_8	MPU_8_INT	[48]	MPU_8 interrupt indicating addressing violation or protection violation error
MPU_9	MPU_9_INT	[49]	MPU_9 interrupt indicating addressing violation or protection violation error
MPU_10	MPU_10_INT	[50]	MPU_10 interrupt indicating addressing violation or protection violation error
MPU_11	MPU_11_INT	[51]	MPU_11 interrupt indicating addressing violation or protection violation error
MPU_12	MPU_12_INT	[52]	MPU_12 interrupt indicating addressing violation or protection violation error
MPU_13	MPU_13_INT	[53]	MPU_13 interrupt indicating addressing violation or protection violation error
MPU_14	MPU_14_INT	[54]	MPU_14 interrupt indicating addressing violation or protection violation error
MPU_15	MPU_15_INT	[55]	MPU_15 interrupt indicating addressing violation or protection violation error
MPU_16	MPU_16_INT	[171]	MPU_16 interrupt indicating addressing violation or protection violation error

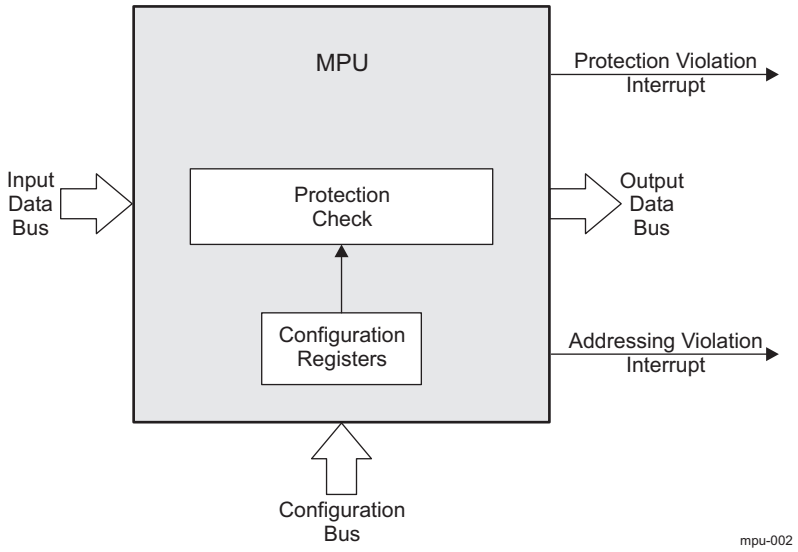
NOTE: For more information about interrupts, see Section 3.2.3.8.

3.2.3 MPU Functional Description

3.2.3.1 Functional Block Diagram

Figure 3-6 shows a block diagram of the MPU. An access to a protected memory must pass through the MPU. During an access, the MPU checks the address on the input data bus against programmable ranges. If allowed, the transfer is passed unmodified to the output data bus. If the transfer fails the protection check then the MPU does not pass the transfer to the output bus. To prevent a hang it services the transfer internally back to the input bus returning the fault status to the requestor and generates an interrupt about the fault. For read accesses the MPU returns zeros and fault status. For write accesses it receives all data and returns fault status.

Figure 3-6. MPU Block Diagram



3.2.3.2 Privilege Levels

The privilege level of a memory access determines what level of permissions the originator of the memory access might have. Two privilege levels are supported: supervisor and user.

Supervisor level is generally granted access to peripheral registers and the MPU configuration registers. User level is generally confined to the memory spaces that the OS specifically designates for its use.

Processor instruction and data accesses have a privilege level associated with them. The privilege level is inherited from the code running on the corresponding processor.

3.2.3.3 Memory Protection Ranges

The MPU divides its assigned memory into address ranges. Each MPU can support up to 16 programmable address ranges. The programmable address range allows software to program the start and end addresses.

Each programmable address range has the following set of registers:

- MPU_PROGx_MPSAR - specifies the start address of region *x*
- MPU_PROGx_MPEAR - specifies the end address of region *x*
- MPU_PROGx_MPPAR - specifies the permissions of region *x*

It is allowed to configure ranges such that they overlap each other. In this case all overlapped ranges must allow the access, otherwise the access is not allowed. The final permissions given to the access are the lowest among all types of permissions associated with these overlapped ranges.

NOTE: The amount of physical memory used by a particular design may be less than the maximum amount of memory supported by the device. In such cases, the not used memory range must be protected in order to prevent unintended (aliased) accesses to the memory in use. One of the programmable address ranges could be used to detect accesses to this not used memory range.

3.2.3.4 Permissions of Programmable Address Range

The fields of the MPU_PROGx_MPPAR register can be split into 3 groups:

- **Allowed privilege IDs** - bits MPU_PROGx_MPPAR[25-9]
- **Security** - bits MPU_PROGx_MPPAR[7-6]
- **Access types** - bits MPU_PROGx_MPPAR[5-0]

Allowed privilege IDs

With almost each master on the device there is an associated privilege ID (called also PrivID). This PrivID identifies the master for privilege purposes and accompanies all memory accesses made on behalf of that master. That is, when a master triggers a memory access command, the PrivID is carried alongside the command.

NOTE: There is no PrivID for the EDMA Controller.

Each AID bit from the MPU_PROGx_MPPAR[25-9] field specifies wheter the master accessing the given address range and having PrivID corresponding to the AID bit is checked for access permissions or not. If AID = 1 the access permissions specified via MPU_PROGx_MPPAR[7-6] and MPU_PROGx_MPPAR[5-0] bits are checked. If AID = 0 these permissions are not checked.

Security

The MPU_PROGx_MPPAR[7] NS and MPU_PROGx_MPPAR[6] EMU bits specify 3 valid protection levels for region *x* as described in Table 3-11. The term *debug access* refers to an access initiated through the emulation port of the device.

Table 3-11. Protection Levels

NS	EMU	Description
0	0	Region *x* is secure without debug: Only secure accesses are allowed. Debug accesses are not allowed. Non-secure accesses are also not allowed
0	1	Region *x* is secure with debug: Only secure and debug accesses are allowed. Non-secure accesses are not allowed
1	-	Region *x* is non-secure: All accesses (non-secure, secure and debug) are allowed.

Access types

The memory protection model defines three fundamental functional access types: read, write, and execute. Read and write refer to data accesses originating from the corresponding processor or master peripheral. Execute refers to accesses associated with an instruction fetch. The memory protection model allows to control the read, write, and execute permissions independently for both privilege levels - supervisor and user. This results in 6 permission bits shown in Table 3-12.

Table 3-12. Request Type Access Controls

Bit	Description
MPU_PROGx_MPPAR[5] SR	Supervisor may read
MPU_PROGx_MPPAR[4] SW	Supervisor may write

Table 3-12. Request Type Access Controls (continued)

Bit	Description
MPU_PROGx_MPPAR[3] SX	Supervisor may execute
MPU_PROGx_MPPAR[2] UR	User may read
MPU_PROGx_MPPAR[1] UW	User may write
MPU_PROGx_MPPAR[0] UX	User may execute

For each bit, a value of 1 permits the access type, and 0 denies it. So, setting the MPU_PROGx_MPPAR[0] UX bit to 1 means that a master in user mode may execute from region *x* specified by the MPU_PROGx_MPSAR and MPU_PROGx_MPEAR registers.

The MPU allows the programmer to specify each of these 6 bits separately. Thus 64 different combinations are possible but programs might not use all of them.

3.2.3.5 Protection Check

During a memory access, the MPU checks if the address range of the input transfer overlaps one of the address ranges defined via registers MPU_PROGx_MPSAR and MPU_PROGx_MPEAR. When the address of the input transfer is within a range the transfer parameters are checked against the address range permissions specified by register MPU_PROGx_MPPAR. The MPU first checks the PrivID against the AID settings. If the corresponding AID bit is 0, then the access permissions of this PrivID are not checked and the access is allowed or not depending on the value of the MPU_CONFIG[0] ASSUME_ALLOWED bit. If the AID bit is 1, then to detect an allowed access the transfer secure and debug parameters are checked against the MPU_PROGx_MPPAR[7] NS and MPU_PROGx_MPPAR[6] EMU values as described in Table 3-11.

For non-debug accesses the read, write, and execute permissions are also checked. There is a set of permissions for supervisor mode and another for user mode. For supervisor mode accesses, the MPU_PROGx_MPPAR[5] SR, MPU_PROGx_MPPAR[4] SW, and MPU_PROGx_MPPAR[3] SX bits are checked. For user mode accesses the MPU_PROGx_MPPAR[2] UR, MPU_PROGx_MPPAR[1] UW, and MPU_PROGx_MPPAR[0] UX bits are checked.

If the address of the input transfer does not match any address range defined via registers MPU_PROGx_MPSAR and MPU_PROGx_MPEAR, then the transfer is allowed or not depending on the value of the MPU_CONFIG[0] ASSUME_ALLOWED bit. If ASSUME_ALLOWED = 0 the transfer is not allowed. If ASSUME_ALLOWED = 1 the transfer is allowed.

In case that a transfer spans multiple address ranges, all overlapped ranges must allow the access, otherwise the access is not allowed. The final permissions given to the access are the lowest among all types of permissions associated with these overlapped ranges. Therefore, if a transfer matches two ranges, one that is RW and one that is RX, then the final permission is just R.

The MPU has a special mechanism for handling DSP L1/L2 cache controller read accesses. See Section 3.2.3.6 for more details.

3.2.3.6 DSP L1/L2 Cache Controller Read Accesses

A memory read access which originates from the DSP L1/L2 cache controller is treated differently to allow memory protection to be enforced by DSP level. This is because a subsequent memory access that hits in the cache does not pass through the MPU. Instead the memory access is serviced directly by the L1/L2 memory controllers.

During a cache memory read the permission settings stored in the MPU_PROGx_MPPAR register are passed to the L1/L2 memory controllers along with the read data. The permission settings returned by the MPU are taken from the MPU_PROGx_MPPAR register that covers the address range of the original request. Only the SR, SW, SX, UR, UW, UX, NS and EMU bits are passed.

If the request address is covered by multiple address ranges, then the returned value is the logical AND of the corresponding permission settings of all MPU_PROGx_MPPAR registers associated with these multiple address ranges. If the transfer address range is not covered by an address range, then the transfer is either allowed or not based on the value of MPU_CONFIG[0] ASSUME_ALLOWED bit.

3.2.3.7 Protection of the MPU Configuration Registers

Accesses to the MPU_PROGx_MPSAR, MPU_PROGx_MPEAR and MPU_PROGx_MPPAR registers are also protected. All non-debug writes must be by a supervisor master. If the MPU_PROGx_MPPAR[7] NS bit is 0, then all writes must be by a secure master. In addition, the NS bit can be modified only by a secure master. A register write with invalid permissions results in protection fault and interrupt generation.

A debug write is only allowed if NS = 1 or the EMU = 1 regardless of the secure or privilege attributes. Neither faults are recorded nor interrupts are generated for debug accesses.

3.2.3.8 MPU Interrupt Requests

Each MPU module has one interrupt request, the MPU_x_INT (x = 0 to 16), associated with the following registers:

- MPU_IRAWSTAT - interrupt raw status register
- MPU_IENSTAT - interrupt status register
- MPU_IENSET - interrupt enable register
- MPU_IENCLR - interrupt disable register

For the interrupt behavior of each MPU module the following applies:

- The MPU module asserts its interrupt line MPU_x_INT (x = 0 to 16) only if the interrupts are enabled by setting to 1h the corresponding bits in the MPU_IENSET register. These interrupts can be disabled by setting to 1h the corresponding bits in the MPU_IENCLR register.
- After an interrupt has been serviced, software must clear the corresponding status flag. This is done by setting to 1h the corresponding bit in the MPU_IENSTAT register which also clears the corresponding bit in the MPU_IRAWSTAT register. The status flags in the MPU_IRAWSTAT register are set even if the corresponding interrupt is disabled unlike those in the MPU_IENSTAT register, which are set only if the corresponding interrupt is enabled.
- An interrupt is also generated by the MPU, if certain bit in the MPU_IRAWSTAT register is set to 1h and the corresponding interrupt is enabled through the MPU_IENSET register. This feature is useful during user software debugging. In addition, even if interrupts are not enabled the corresponding raw flag in the MPU_IRAWSTAT register is set to 1h when an IRQ condition occurs.
- Even if interrupts are not enabled, a status bit in the MPU_IRAWSTAT register can also be cleared by setting to 1h the corresponding bit in the MPU_IENSTAT register.

Table 3-13 lists the interrupt events which can assert the MPU_x_INT interrupt line.

Table 3-13. MPU Events

Event Flag	Event Mask	Description
MPU_IRAWSTAT[1] ADDR_ERR MPU_IENSTAT[1] ENABLED_ADDR_ERR	MPU_IENSET[1] ADDR_ERR_EN MPU_IENCLR[1] ADDR_ERR_EN_CLR	Addressing violation interrupt. Occurs when an access to a non-existent location in the MPU configuration register space is performed.
MPU_IRAWSTAT[0] PROT_ERR MPU_IENSTAT[0] ENABLED_PROT_ERR	MPU_IENSET[0] PROT_ERR_EN MPU_IENCLR[0] PROT_ERR_EN_CLR	This interrupt occurs in case of protection violation of the programmable regions or of the MPU configuration register space.

When addressing violation or protection violation occurs, the error associated details are captured in the MPU_FLTADDRR and MPU_FLTSTAT registers. MPU_FLTADDRR contains the address of the first fault access and MPU_FLTSTAT contains status attributes associated with the first fault access. To clear the current fault the MPU_FLTCLR[0] CLEAR bit must be set to 1h. If not cleared, other faults are not recorded as the MPU stores fault information for only one fault. The MPU neither records another fault nor generates another interrupt until the existing fault has been cleared. This means that all other faults are ignored. For debug accesses neither faults are recorded nor interrupts are generated.

3.2.3.9 Emulation Considerations

Memory is protected against emulation accesses by the MPU. Emulator access to memory is controlled through the MPU_PROGx_MPPAR[7] NS and MPU_PROGx_MPPAR[6] EMU bits. See Table 3-11 for more details.

The MPU also protects its registers MPU_PROGx_MPSAR, MPU_PROGx_MPEAR and MPU_PROGx_MPPAR against emulation accesses. See Section 3.2.3.7 for more details.

3.2.3.10 MPU Memory Regions

The device contains 17 MPUs. Table 3-14 lists these MPUs and also the regions protected by each MPU.

Table 3-14. Memory Regions Protected by MPUs

MPU	Regions Protected by MPU
MPU_0	MESSAGE_MANAGER_0_CFG
MPU_1	TeraNet_P_6_32_4
MPU_2	PMMC_0_SLV
MPU_3	TeraNet_P_3_32_11
MPU_4	PCIE_0_SLV
MPU_5	MPU_5 is reserved and not used
MPU_6	TeraNet_M_3_32_0
MPU_7	TeraNet_P_6_32_2
MPU_8	GIC_0_CFG
MPU_9	CIC_0_CFG
MPU_10	TeraNet_P_3_32_8
MPU_11	TeraNet_P_6_32_0
MPU_12	TeraNet_P_6_32_1
MPU_13	TeraNet_P_3_32_7
MPU_14	TeraNet_P_6_32_5
MPU_15	TeraNet_M_3_128_4
MPU_16	TeraNet_P_6_32_7

3.2.3.11 Master IDs and PrivIDs

Table 3-15 shows the unique Master ID assigned to each master on the device. Not listed values are not used and should be considered as reserved IDs.

Table 3-16 shows the PrivID assigned to each master on the device.

NOTE: Master IDs are used to determine allowed connections between masters and slaves. They are also used to determine the master performed a fault transfer.

PrivIDs are used by the MPUs and MSMC to protect a given memory range against undesirable accesses.

Master IDs are unique to each master unlike PrivIDs which are shared among different masters.

Table 3-15. Master IDs

Master ID (decimal)	Master
0	MSMC_0_SYS_MST:C66X_COREPAC_0_MDMA_MST
8	MSMC_0_SYS_MST:A15_CPU0
16	C66X_COREPAC_0_CFG

Table 3-15. Master IDs (continued)

Master ID (decimal)	Master
26	PCIE_0_MST
27	ICSS_0_MST0
28	ICSS_0_MST1
29	ICSS_1_MST0
30	ICSS_1_MST1
31	PMMC_0_M
32-35	NSSL_MST
39	EDMA_0_CC_TR
40	EDMA_1_CC_TR
41	EDMA_0_TC0_R
42	EDMA_0_TC0_W
43	EDMA_0_TC1_R
44	EDMA_0_TC1_W
45	EDMA_1_TC0_R
46	EDMA_1_TC0_W
47	EDMA_1_TC1_R
48	EDMA_1_TC1_W
50	USB_0_MST
51	USB_1_MST
57	DEBUGSS_0_DAP_MST
59	MLB_0_M
62	MMCHS_0_M
63	MMCHS_1_M
96-111	DSSUL_0_M
140	CPT_0_M
141	CPT_1_M
142	CPT_2_M
143	CPT_3_M
144	CPT_4_M
145	CPT_5_M
146	CPT_6_M
147	CPT_7_M
148	CPT_8_M
149	CPT_9_M
150	CPT_10_M
151	CPT_11_M
152	CPT_12_M
153	CPT_13_M
154	CPT_14_M

Table 3-16. PrivIDs

PrivID (decimal)	Master
0	MSMC_0_SYS_MST:C66X_COREPAC_0_MDMA_MST
1	MSMC_0_SYS_MST:A15_CPU0
2	ICSS_0_MST0, ICSS_0_MST1
3	ICSS_1_MST0, ICSS_1_MST1

Table 3-16. PrivIDs (continued)

PrivID (decimal)	Master
4	NSSL_MST
5	PCIE_0_MST
6	USB_0_MST, USB_1_MST
7	Not assigned
8	MLB_0_M
9	PMMC_0_M
10	DSSUL_0_M
11	MMCHS_0_M, MMCHS_1_M
12	DEBUGSS_0_DAP_MST
13	Not assigned
14	Not assigned
15	Not assigned

3.2.4 MPU Registers

Table 3-18 through Table 3-21 lists the memory-mapped registers for the device MPUs. All register offset addresses not listed in these tables should be considered as reserved locations and the register contents should not be modified.

Table 3-17. MPU Instances

Instance	Base Address
MPU_0	0236 0000h
MPU_1	0236 8000h
MPU_2	0237 0000h
MPU_3	0237 8000h
MPU_4	0238 0000h
MPU_5	0238 8000h
MPU_6	0238 8400h
MPU_7	0238 8800h
MPU_8	0238 8C00h
MPU_9	0238 9000h
MPU_10	0238 9400h
MPU_11	0238 9800h
MPU_12	0238 9C00h
MPU_13	0238 A000h
MPU_14	0238 A400h
MPU_15	0238 A800h
MPU_16	0238 AC00h

Table 3-18. MPU Registers

Offset	Acronym	Register Name	MPU_0 Physical Address	MPU_1 Physical Address	MPU_2 Physical Address	MPU_3 Physical Address	Section
0h	MPU_REVID	Revision ID	0236 0000h	0236 8000h	0237 0000h	0237 8000h	Section 3.2.4.1
4h	MPU_CONFIG	Configuration	0236 0004h	0236 8004h	0237 0004h	0237 8004h	Section 3.2.4.2
10h	MPU_IRAWSTAT	Interrupt raw status/set	0236 0010h	0236 8010h	0237 0010h	0237 8010h	Section 3.2.4.3.1
14h	MPU_IENSTAT	Interrupt enable status/clear	0236 0014h	0236 8014h	0237 0014h	0237 8014h	Section 3.2.4.3.2
18h	MPU_IENSET	Interrupt enable	0236 0018h	0236 8018h	0237 0018h	0237 8018h	Section 3.2.4.3.3
1Ch	MPU_IENCLR	Interrupt enable clear	0236 001Ch	0236 801Ch	0237 001Ch	0237 801Ch	Section 3.2.4.3.4
20h	MPU_EOI	End of interrupt	0236 0020h	0236 8020h	0237 0020h	0237 8020h	Section 3.2.4.3.5
200h + (x*16)	MPU_PROGx_MPSAR	Programmable range x, start address (x = 0 to 15)	0236 0200h + (x*16)	0236 8200h + (x*16)	0237 0200h + (x*16)	0237 8200h + (x*16)	Section 3.2.4.4.1
204h + (x*16)	MPU_PROGx_MPEAR	Programmable range x, end address (x = 0 to 15)	0236 0204h + (x*16)	0236 8204h + (x*16)	0237 0204h + (x*16)	0237 8204h + (x*16)	Section 3.2.4.4.2

Table 3-18. MPU Registers (continued)

Offset	Acronym	Register Name	MPU_0 Physical Address	MPU_1 Physical Address	MPU_2 Physical Address	MPU_3 Physical Address	Section
208h + (x*16)	MPU_PROGx_MPPAR	Programmable range x, memory page protection attributes (x = 0 to 15)	0236 0208h + (x*16)	0236 8208h + (x*16)	0237 0208h + (x*16)	0237 8208h + (x*16)	Section 3.2.4.4.3
300h	MPU_FLTADDRR	Fault address	0236 0300h	0236 8300h	0237 0300h	0237 8300h	Section 3.2.4.5.1
304h	MPU_FLTSTAT	Fault status	0236 0304h	0236 8304h	0237 0304h	0237 8304h	Section 3.2.4.5.2
308h	MPU_FLTCLR	Fault clear	0236 0308h	0236 8308h	0237 0308h	0237 8308h	Section 3.2.4.5.3

Table 3-19. MPU Registers

Offset	Acronym	Register Name	MPU_4 Physical Address	MPU_5 Physical Address	MPU_6 Physical Address	MPU_7 Physical Address	Section
0h	MPU_REVID	Revision ID	0238 0000h	0238 8000h	0238 8400h	0238 8800h	Section 3.2.4.1
4h	MPU_CONFIG	Configuration	0238 0004h	0238 8004h	0238 8404h	0238 8804h	Section 3.2.4.2
10h	MPU_IRAWSTAT	Interrupt raw status/set	0238 0010h	0238 8010h	0238 8410h	0238 8810h	Section 3.2.4.3.1
14h	MPU_IENSTAT	Interrupt enable status/clear	0238 0014h	0238 8014h	0238 8414h	0238 8814h	Section 3.2.4.3.2
18h	MPU_IENSET	Interrupt enable	0238 0018h	0238 8018h	0238 8418h	0238 8818h	Section 3.2.4.3.3
1Ch	MPU_IENCLR	Interrupt enable clear	0238 001Ch	0238 801Ch	0238 841Ch	0238 881Ch	Section 3.2.4.3.4
20h	MPU_EOI	End of interrupt	0238 0020h	0238 8020h	0238 8420h	0238 8820h	Section 3.2.4.3.5
200h + (x*16)	MPU_PROGx_MPSAR	Programmable range x, start address (x = 0 to 15)	0238 0200h + (x*16)	0238 8200h + (x*16)	0238 8600h + (x*16)	0238 8A00h + (x*16)	Section 3.2.4.4.1
204h + (x*16)	MPU_PROGx_MPEAR	Programmable range x, end address (x = 0 to 15)	0238 0204h + (x*16)	0238 8204h + (x*16)	0238 8604h + (x*16)	0238 8A04h + (x*16)	Section 3.2.4.4.2
208h + (x*16)	MPU_PROGx_MPPAR	Programmable range x, memory page protection attributes (x = 0 to 15)	0238 0208h + (x*16)	0238 8208h + (x*16)	0238 8608h + (x*16)	0238 8A08h + (x*16)	Section 3.2.4.4.3
300h	MPU_FLTADDRR	Fault address	0238 0300h	0238 8300h	0238 8700h	0238 8B00h	Section 3.2.4.5.1
304h	MPU_FLTSTAT	Fault status	0238 0304h	0238 8304h	0238 8704h	0238 8B04h	Section 3.2.4.5.2
308h	MPU_FLTCLR	Fault clear	0238 0308h	0238 8308h	0238 8708h	0238 8B08h	Section 3.2.4.5.3

Table 3-20. MPU Registers

Offset	Acronym	Register Name	MPU_8 Physical Address	MPU_9 Physical Address	MPU_10 Physical Address	MPU_11 Physical Address	Section
0h	MPU_REVID	Revision ID	0238 8C00h	0238 9000h	0238 9400h	0238 9800h	Section 3.2.4.1
4h	MPU_CONFIG	Configuration	0238 8C04h	0238 9004h	0238 9404h	0238 9804h	Section 3.2.4.2
10h	MPU_IRAWSTAT	Interrupt raw status/set	0238 8C10h	0238 9010h	0238 9410h	0238 9810h	Section 3.2.4.3.1
14h	MPU_IENSTAT	Interrupt enable status/clear	0238 8C14h	0238 9014h	0238 9414h	0238 9814h	Section 3.2.4.3.2
18h	MPU_IENSET	Interrupt enable	0238 8C18h	0238 9018h	0238 9418h	0238 9818h	Section 3.2.4.3.3
1Ch	MPU_IENCLR	Interrupt enable clear	0238 8C1Ch	0238 901Ch	0238 941Ch	0238 981Ch	Section 3.2.4.3.4
20h	MPU_EOI	End of interrupt	0238 8C20h	0238 9020h	0238 9420h	0238 9820h	Section 3.2.4.3.5
200h + (x*16)	MPU_PROGx_MPSAR	Programmable range x, start address (x = 0 to 15)	0238 8E00h + (x*16)	0238 9200h + (x*16)	0238 9600h + (x*16)	0238 9A00h + (x*16)	Section 3.2.4.4.1
204h + (x*16)	MPU_PROGx_MPEAR	Programmable range x, end address (x = 0 to 15)	0238 8E04h + (x*16)	0238 9204h + (x*16)	0238 9604h + (x*16)	0238 9A04h + (x*16)	Section 3.2.4.4.2
208h + (x*16)	MPU_PROGx_MPPAR	Programmable range x, memory page protection attributes (x = 0 to 15)	0238 8E08h + (x*16)	0238 9208h + (x*16)	0238 9608h + (x*16)	0238 9A08h + (x*16)	Section 3.2.4.4.3
300h	MPU_FLTADDRR	Fault address	0238 8F00h	0238 9300h	0238 9700h	0238 9B00h	Section 3.2.4.5.1
304h	MPU_FLTSTAT	Fault status	0238 8F04h	0238 9304h	0238 9704h	0238 9B04h	Section 3.2.4.5.2
308h	MPU_FLTCLR	Fault clear	0238 8F08h	0238 9308h	0238 9708h	0238 9B08h	Section 3.2.4.5.3

Table 3-21. MPU Registers

Offset	Acronym	Register Name	MPU_12 Physical Address	MPU_13 Physical Address	MPU_14 Physical Address	MPU_15 Physical Address	MPU_16 Physical Address	Section
0h	MPU_REVID	Revision ID	0238 9C00h	0238 A000h	0238 A400h	0238 A800h	0238 AC00h	Section 3.2.4.1
4h	MPU_CONFIG	Configuration	0238 9C04h	0238 A004h	0238 A404h	0238 A804h	0238 AC04h	Section 3.2.4.2
10h	MPU_IRAWSTAT	Interrupt raw status/set	0238 9C10h	0238 A010h	0238 A410h	0238 A810h	0238 AC10h	Section 3.2.4.3.1
14h	MPU_IENSTAT	Interrupt enable status/clear	0238 9C14h	0238 A014h	0238 A414h	0238 A814h	0238 AC14h	Section 3.2.4.3.2
18h	MPU_IENSET	Interrupt enable	0238 9C18h	0238 A018h	0238 A418h	0238 A818h	0238 AC18h	Section 3.2.4.3.3
1Ch	MPU_IENCLR	Interrupt enable clear	0238 9C1Ch	0238 A01Ch	0238 A41Ch	0238 A81Ch	0238 AC1Ch	Section 3.2.4.3.4
20h	MPU_EOI	End of interrupt	0238 9C20h	0238 A020h	0238 A420h	0238 A820h	0238 AC20h	Section 3.2.4.3.5
200h + (x*16)	MPU_PROGx_MPSAR	Programmable range x, start address (x = 0 to 15)	0238 9E00h + (x*16)	0238 A200h + (x*16)	0238 A600h + (x*16)	0238 AA00h + (x*16)	0238 AE00h + (x*16)	Section 3.2.4.4.1

Table 3-21. MPU Registers (continued)

Offset	Acronym	Register Name	MPU_12 Physical Address	MPU_13 Physical Address	MPU_14 Physical Address	MPU_15 Physical Address	MPU_16 Physical Address	Section
204h + (x*16)	MPU_PROGx_MPEAR	Programmable range x, end address (x = 0 to 15)	0238 9E04h + (x*16)	0238 A204h + (x*16)	0238 A604h + (x*16)	0238 AA04h + (x*16)	0238 AE04h + (x*16)	Section 3.2.4.4.2
208h + (x*16)	MPU_PROGx_MPPAR	Programmable range x, memory page protection attributes (x = 0 to 15)	0238 9E08h + (x*16)	0238 A208h + (x*16)	0238 A608h + (x*16)	0238 AA08h + (x*16)	0238 AE08h + (x*16)	Section 3.2.4.4.3
300h	MPU_FLTADDRR	Fault address	0238 9F00h	0238 A300h	0238 A700h	0238 AB00h	0238 AF00h	Section 3.2.4.5.1
304h	MPU_FLTSTAT	Fault status	0238 9F04h	0238 A304h	0238 A704h	0238 AB04h	0238 AF04h	Section 3.2.4.5.2
308h	MPU_FLTCLR	Fault clear	0238 9F08h	0238 A308h	0238 A708h	0238 AB08h	0238 AF08h	Section 3.2.4.5.3

3.2.4.1 MPU_REVID Register (Offset = 0h) [reset = 4E814901h]

MPU_REVID is shown in Figure 3-7 and described in Table 3-23.

The revision ID register (MPU_REVID) contains MPU revision information.

Table 3-22. MPU_REVID Instances

Instance	Physical Address
MPU_0	0236 0000h
MPU_1	0236 8000h
MPU_2	0237 0000h
MPU_3	0237 8000h
MPU_4	0238 0000h
MPU_5	0238 8000h
MPU_6	0238 8400h
MPU_7	0238 8800h
MPU_8	0238 8C00h
MPU_9	0238 9000h
MPU_10	0238 9400h
MPU_11	0238 9800h
MPU_12	0238 9C00h
MPU_13	0238 A000h
MPU_14	0238 A400h
MPU_15	0238 A800h
MPU_16	0238 AC00h

Figure 3-7. MPU_REVID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-4E814901h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 3-23. MPU_REVID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4E814901h	TI internal data. Identifies revision of peripheral.

Table 3-24. Register Call Summary for MPU_REVID

MPU Registers
- MPU Registers: [0][1][2][3]
- MPU_REVID Register (Offset = 0h) [reset = 4E814901h]: [0][1]

3.2.4.2 MPU_CONFIG Register (Offset = 4h) [reset = See Table 3-27]

MPU_CONFIG is shown in Figure 3-8 and described in Table 3-26.

The configuration register (MPU_CONFIG) contains the configuration value of the MPU.

Table 3-25. MPU_CONFIG Instances

Instance	Physical Address
MPU_0	0236 0004h
MPU_1	0236 8004h
MPU_2	0237 0004h
MPU_3	0237 8004h
MPU_4	0238 0004h
MPU_5	0238 8004h
MPU_6	0238 8404h
MPU_7	0238 8804h
MPU_8	0238 8C04h
MPU_9	0238 9004h
MPU_10	0238 9404h
MPU_11	0238 9804h
MPU_12	0238 9C04h
MPU_13	0238 A004h
MPU_14	0238 A404h
MPU_15	0238 A804h
MPU_16	0238 AC04h

Figure 3-8. MPU_CONFIG Register

31	30	29	28	27	26	25	24
ADDR_WIDTH							
R-See Table 3-27							
23	22	21	20	19	18	17	16
NUM_FIXED				NUM_PROG			
R-See Table 3-27				R-See Table 3-27			
15	14	13	12	11	10	9	8
NUM_AIDS				RESERVED			
R-See Table 3-27				R-0h			
7	6	5	4	3	2	1	0
RESERVED							ASSUME_ALLOWED
R-0h							R-See Table 3-27

LEGEND: R = Read Only; *-n* = value after reset

Table 3-26. MPU_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	ADDR_WIDTH	R	See Table 3-27	Address alignment for range checking. 0 = 1 KB alignment 1 = 2 KB alignment ... 6 = 64 KB alignment

Table 3-26. MPU_CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-20	NUM_FIXED	R	See Table 3-27	Shows the number of fixed address ranges supproted by the corresponding MPU. 0 = 0 fixed address ranges supproted 1 = 1 fixed address range supproted 2 = 2 fixed address ranges supproted ... F = 15 fixed address ranges
19-16	NUM_PROG	R	See Table 3-27	Shows the number of programmable address ranges supproted by the corresponding MPU. 0 = 16 programmable address ranges 1 = 1 programmable address range 2 = 2 programmable address ranges 3 = 3 programmable address ranges ... F = 15 programmable address ranges
15-12	NUM_AIDS	R	See Table 3-27	Shows the number of supported AIDs. 0 = 16 supported AIDs 1 = 1 supported AID 2 = 2 supported AIDs 3 = 3 supported AIDs ... F = 15 supported AIDs
11-1	RESERVED	R	0h	Reserved. Always reads as 0.
0	ASSUME_ALLOWED	R	See Table 3-27	Assume allowed bit. When an address is not covered by any MPU protection range, this bit determines whether the transfer is assumed to be allowed or not. 0 = Assume disallowed 1 = Assume allowed

Table 3-27. Reset Values of the MPU_CONFIG Register Fields

MPU	ADDR_WIDTH (hex)	NUM_FIXED (hex)	NUM_PROG (hex)	NUM_AIDS (hex)	ASSUME_ALLOWED (hex)
MPU_0	0	0	0	0	1
MPU_1	0	0	0	0	1
MPU_2	0	0	0	0	1
MPU_3	0	0	0	0	1
MPU_4	0	0	0	0	1
MPU_5	MPU_5 is reserved and not used				
MPU_6	0	0	0	0	1
MPU_7	0	0	0	0	1
MPU_8	0	0	0	0	1
MPU_9	0	0	0	0	1
MPU_10	0	0	0	0	1
MPU_11	0	0	0	0	1
MPU_12	0	0	0	0	1
MPU_13	0	0	0	0	1
MPU_14	0	0	0	0	1
MPU_15	0	0	0	0	1
MPU_16	0	0	0	0	1

Table 3-28. Register Call Summary for MPU_CONFIG

MPU Registers • MPU Registers: [0][1][2][3] • MPU_CONFIG Register (Offset = 4h) [reset = See]: [0][1]
Memory Protection Unit Functional Description • Protection Check: [0][1] • DSP L1/L2 Cache Controller Read Accesses: [0]

3.2.4.3 MPU Interrupt Registers

3.2.4.3.1 MPU_IRAWSTAT Register (Offset = 10h) [reset = 0h]

MPU_IRAWSTAT is shown in Figure 3-9 and described in Table 3-30.

Reading the interrupt raw status/set register (MPU_IRAWSTAT) returns the status of all interrupts. Software can write to MPU_IRAWSTAT to manually set an interrupt. However, an interrupt is generated only if the interrupt is enabled. Writes of 0 have no effect.

Table 3-29. MPU_IRAWSTAT Instances

Instance	Physical Address
MPU_0	0236 0010h
MPU_1	0236 8010h
MPU_2	0237 0010h
MPU_3	0237 8010h
MPU_4	0238 0010h
MPU_5	0238 8010h
MPU_6	0238 8410h
MPU_7	0238 8810h
MPU_8	0238 8C10h
MPU_9	0238 9010h
MPU_10	0238 9410h
MPU_11	0238 9810h
MPU_12	0238 9C10h
MPU_13	0238 A010h
MPU_14	0238 A410h
MPU_15	0238 A810h
MPU_16	0238 AC10h

Figure 3-9. MPU_IRAWSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ADDR_ERR	PROT_ERR
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 3-30. MPU_IRAWSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved. Always reads as 0.
1	ADDR_ERR	R/W	0h	Addressing violation error. Raw status is read. • Write a 1 to set the status. Writing a 0 has no effect.

Table 3-30. MPU_IRAWSTAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	PROT_ERR	R/W	0h	Protection violation error. Raw status is read. • Write a 1 to set the status. Writing a 0 has no effect.

Table 3-31. Register Call Summary for MPU_IRAWSTAT

MPU Registers • MPU Registers: [0][1][2][3]
Memory Protection Unit Functional Description • MPU Interrupt Requests: [0][1][2][3][4][5][6][7]
MPU Interrupt Registers • MPU_IENSTAT Register (Offset = 14h) [reset = 0h]: [0] • MPU_IRAWSTAT Register (Offset = 10h) [reset = 0h]: [0][1][2]

3.2.4.3.2 MPU_IENSTAT Register (Offset = 14h) [reset = 0h]

MPU_IENSTAT is shown in Figure 3-10 and described in Table 3-33.

Reading the interrupt enable status/clear register (MPU_IENSTAT) returns the status of only those interrupts that are enabled. Software can write to the bits of MPU_IENSTAT to clear an interrupt; the interrupt is cleared both from MPU_IENSTAT and MPU_IRAWSTAT. Writes of 0 have no effect.

Table 3-32. MPU_IENSTAT Instances

Instance	Physical Address
MPU_0	0236 0014h
MPU_1	0236 8014h
MPU_2	0237 0014h
MPU_3	0237 8014h
MPU_4	0238 0014h
MPU_5	0238 8014h
MPU_6	0238 8414h
MPU_7	0238 8814h
MPU_8	0238 8C14h
MPU_9	0238 9014h
MPU_10	0238 9414h
MPU_11	0238 9814h
MPU_12	0238 9C14h
MPU_13	0238 A014h
MPU_14	0238 A414h
MPU_15	0238 A814h
MPU_16	0238 AC14h

Figure 3-10. MPU_IENSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ENABLED_ADDR_ERR	ENABLED_PROT_ERR
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 3-33. MPU_IENSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved. Always reads as 0.
1	ENABLED_ADDR_ERR	R/W	0h	Addressing violation error. Enabled status is read. • Write a 1 to clear the status. Writing a 0 has no effect.
0	ENABLED_PROT_ERR	R/W	0h	Protection violation error. Enabled status is read. • Write a 1 to clear the status. Writing a 0 has no effect.

Table 3-34. Register Call Summary for MPU_IENSTAT

MPU Registers • MPU Registers: [0][1][2][3]
Memory Protection Unit Functional Description • MPU Interrupt Requests: [0][1][2][3][4][5]
MPU Interrupt Registers • MPU_IENSTAT Register (Offset = 14h) [reset = 0h]: [0][1][2][3]

3.2.4.3.3 MPU_IENSET Register (Offset = 18h) [reset = 0h]

MPU_IENSET is shown in Figure 3-11 and described in Table 3-36.

Reading the interrupt enable register (MPU_IENSET) returns the interrupts that are enabled. Software can write to MPU_IENSET to enable an interrupt. Writes of 0 have no effect.

Table 3-35. MPU_IENSET Instances

Instance	Physical Address
MPU_0	0236 0018h
MPU_1	0236 8018h
MPU_2	0237 0018h
MPU_3	0237 8018h
MPU_4	0238 0018h
MPU_5	0238 8018h
MPU_6	0238 8418h
MPU_7	0238 8818h
MPU_8	0238 8C18h
MPU_9	0238 9018h
MPU_10	0238 9418h
MPU_11	0238 9818h
MPU_12	0238 9C18h
MPU_13	0238 A018h
MPU_14	0238 A418h
MPU_15	0238 A818h
MPU_16	0238 AC18h

Figure 3-11. MPU_IENSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ADDR_ERR_EN	PROT_ERR_EN
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 3-36. MPU_IENSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved. Always reads as 0.
1	ADDR_ERR_EN	R/W	0h	Addressing violation error enable. • Write a 1 to set the enable. Writing a 0 has no effect.
0	PROT_ERR_EN	R/W	0h	Protection violation error enable. • Write a 1 to set the enable. Writing a 0 has no effect.

Table 3-37. Register Call Summary for MPU_IENSET

MPU Registers • MPU Registers: [0][1][2][3]
Memory Protection Unit Functional Description • MPU Interrupt Requests: [0][1][2][3][4]
MPU Interrupt Registers • MPU_IENSET Register (Offset = 18h) [reset = 0h]: [0][1][2]

3.2.4.3.4 MPU_IENCLR Register (Offset = 1Ch) [reset = 0h]

MPU_IENCLR is shown in Figure 3-12 and described in Table 3-39.

Reading the interrupt enable clear register (MPU_IENCLR) returns the interrupts that are enabled. Software can write to MPU_IENCLR to clear/disable an interrupt. Writes of 0 have no effect.

Table 3-38. MPU_IENCLR Instances

Instance	Physical Address
MPU_0	0236 001Ch
MPU_1	0236 801Ch
MPU_2	0237 001Ch
MPU_3	0237 801Ch
MPU_4	0238 001Ch
MPU_5	0238 801Ch
MPU_6	0238 841Ch
MPU_7	0238 881Ch
MPU_8	0238 8C1Ch
MPU_9	0238 901Ch
MPU_10	0238 941Ch
MPU_11	0238 981Ch
MPU_12	0238 9C1Ch
MPU_13	0238 A01Ch
MPU_14	0238 A41Ch
MPU_15	0238 A81Ch
MPU_16	0238 AC1Ch

Figure 3-12. MPU_IENCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ADDR_ERR_EN_CLR	PROT_ERR_EN_CLR
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 3-39. MPU_IENCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved. Always reads as 0.
1	ADDR_ERR_EN_CLR	R/W	0h	Addressing violation error enable. • Write a 1 to clear the enable. Writing a 0 has no effect.
0	PROT_ERR_EN_CLR	R/W	0h	Protection violation error enable. • Write a 1 to clear the enable. Writing a 0 has no effect.

Table 3-40. Register Call Summary for MPU_IENCLR

MPU Registers • MPU Registers: [0][1][2][3]
Memory Protection Unit Functional Description • MPU Interrupt Requests: [0][1][2][3]
MPU Interrupt Registers • MPU_IENCLR Register (Offset = 1Ch) [reset = 0h]: [0][1][2]

3.2.4.3.5 MPU_EOI Register (Offset = 20h) [reset = 0h]

MPU_EOI is shown in Figure 3-13 and described in Table 3-42.

The MPU_EOI register allows software to indicate when the end of interrupt service is complete. The MPU_EOI vector value is dependent on the interrupt handling.

Table 3-41. MPU_EOI Instances

Instance	Physical Address
MPU_0	0236 0020h
MPU_1	0236 8020h
MPU_2	0237 0020h
MPU_3	0237 8020h
MPU_4	0238 0020h
MPU_5	0238 8020h
MPU_6	0238 8420h
MPU_7	0238 8820h
MPU_8	0238 8C20h
MPU_9	0238 9020h
MPU_10	0238 9420h
MPU_11	0238 9820h
MPU_12	0238 9C20h
MPU_13	0238 A020h
MPU_14	0238 A420h
MPU_15	0238 A820h
MPU_16	0238 AC20h

Figure 3-13. MPU_EOI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								EOI_VECTOR							
R-0h																								R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 3-42. MPU_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved. Always reads as 0.
7-0	EOI_VECTOR	R/W	0h	MPU_EOI vector value. Write this with the interrupt distribution value in the device. This drives the mpu_eoi_vector output signal. MPU_EOI Value is 0 for MPU.

Table 3-43. Register Call Summary for MPU_EOI

MPU Registers • MPU Registers: [0][1][2][3]
MPU Interrupt Registers • MPU_EOI Register (Offset = 20h) [reset = 0h]: [0][1][2][3][4]

3.2.4.4 MPU Programmable Range Registers

3.2.4.4.1 MPU_PROGx_MPSAR Register (Offset = 200h + x*16) [reset = See Table 3-46]

MPU_PROGx_MPSAR (*x* = 0 to 15) is shown in Figure 3-14 and described in Table 3-45.

The programmable address start register holds the start address for the range. This register is writable by a supervisor entity only. If NS = 0 (non-secure mode) in the associated MPPA register then the register is also only writable by a secure entity.

The start address must be aligned on a page boundary. The size of the page is 1 Kbyte. The size of the page determines the width of the address field in MPSAR and MPEAR.

Table 3-44. MPU_PROGx_MPSAR Instances

Instance	Physical Address
MPU_0	0236 0200h + (x*16)
MPU_1	0236 8200h + (x*16)
MPU_2	0237 0200h + (x*16)
MPU_3	0237 8200h + (x*16)
MPU_4	0238 0200h + (x*16)
MPU_5	0238 8200h + (x*16)
MPU_6	0238 8600h + (x*16)
MPU_7	0238 8A00h + (x*16)
MPU_8	0238 8E00h + (x*16)
MPU_9	0238 9200h + (x*16)
MPU_10	0238 9600h + (x*16)
MPU_11	0238 9A00h + (x*16)
MPU_12	0238 9E00h + (x*16)
MPU_13	0238 A200h + (x*16)
MPU_14	0238 A600h + (x*16)
MPU_15	0238 AA00h + (x*16)
MPU_16	0238 AE00h + (x*16)

Figure 3-14. MPU_PROGx_MPSAR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
START_ADDR																						RESERVED									
R/W-See Table 3-46																						R-0h									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 3-45. MPU_PROGx_MPSAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	START_ADDR	R/W	See Table 3-46	Start address for range *x*.
9-0	RESERVED	R	0h	Reserved. Always reads as 0.

Table 3-46. Reset Values of the MPU_PROGx_MPSAR Registers (x = 0 to 15)

Register	MPU_0	MPU_1	MPU_2	MPU_3	MPU_4	MPU_5
MPU_PROG0_MPSAR	02880000h	02603000h	02900000h	21818000h	21800000h	Reserved
MPU_PROG1_MPSAR	028A0000h	02350000h	02930000h	30000000h	70000000h	Reserved
MPU_PROG2_MPSAR	028C0000h	00000000h	02931000h	00000000h	00000000h	Reserved
MPU_PROG3_MPSAR	028C1000h	02620400h	02932000h	24000000h	00000000h	Reserved
MPU_PROG4_MPSAR	028C2000h	02500000h	02933000h	23000000h	00000000h	Reserved

Table 3-46. Reset Values of the MPU_PROGx_MPSAR Registers (x = 0 to 15) (continued)

Register	MPU_0	MPU_1	MPU_2	MPU_3	MPU_4	MPU_5
MPU_PROG5_MPSAR	028C3000h	02310000h	02934000h	23100000h	00000000h	Reserved
MPU_PROG6_MPSAR	028C3400h	02260000h	02935000h	00000000h	00000000h	Reserved
MPU_PROG7_MPSAR	02A00000h	02530000h	02936000h	00000000h	00000000h	Reserved
MPU_PROG8_MPSAR	00000000h	0232F000h	00000000h	00000000h	00000000h	Reserved
MPU_PROG9_MPSAR	00000000h	021D2000h	00000000h	00000000h	00000000h	Reserved
MPU_PROG10_MPSAR	00000000h	024C0000h	00000000h	00000000h	00000000h	Reserved
MPU_PROG11_MPSAR	00000000h	024C0400h	00000000h	00000000h	00000000h	Reserved
MPU_PROG12_MPSAR	00000000h	00000000h	00000000h	00000000h	00000000h	Reserved
MPU_PROG13_MPSAR	00000000h	00000000h	00000000h	00000000h	00000000h	Reserved
MPU_PROG14_MPSAR	00000000h	02604000h	00000000h	00000000h	00000000h	Reserved
MPU_PROG15_MPSAR	00000000h	02620000h	00000000h	00000000h	00000000h	Reserved
Register	**MPU_6**	**MPU_7**	**MPU_8**	**MPU_9**	**MPU_10**	**MPU_11**
MPU_PROG0_MPSAR	00000000h	00000000h	02560000h	02600000h	02340000h	02440000h
MPU_PROG1_MPSAR	00000000h	00000000h	00000000h	00000000h	02342000h	03000000h
MPU_PROG2_MPSAR	21804000h	00000000h	00000000h	00000000h	02344000h	00000000h
MPU_PROG3_MPSAR	21804400h	00000000h	00000000h	00000000h	02346000h	00000000h
MPU_PROG4_MPSAR	21804800h	02320000h	00000000h	00000000h	00000000h	00000000h
MPU_PROG5_MPSAR	21805000h	02329000h	00000000h	00000000h	00000000h	00000000h
MPU_PROG6_MPSAR	00000000h	02680000h	00000000h	00000000h	01E80000h	00000000h
MPU_PROG7_MPSAR	00000000h	02630000h	00000000h	00000000h	02540000h	00000000h
MPU_PROG8_MPSAR	00000000h	02580000h	00000000h	00000000h	021C6000h	00000000h
MPU_PROG9_MPSAR	00000000h	02650000h	00000000h	00000000h	021C8000h	00000000h
MPU_PROG10_MPSAR	00000000h	02940000h	00000000h	00000000h	02640000h	00000000h
MPU_PROG11_MPSAR	00000000h	00000000h	00000000h	00000000h	04000000h	00000000h
MPU_PROG12_MPSAR	00000000h	00000000h	00000000h	00000000h	021E0000h	00000000h
MPU_PROG13_MPSAR	00000000h	00000000h	00000000h	00000000h	02330400h	00000000h
MPU_PROG14_MPSAR	00000000h	00000000h	00000000h	00000000h	02640000h	00000000h
MPU_PROG15_MPSAR	00000000h	00000000h	00000000h	00000000h	00000000h	00000000h
Register	**MPU_12**	**MPU_13**	**MPU_14**	**MPU_15**	**MPU_16**	
MPU_PROG0_MPSAR	021D1800h	01E84000h	21805400h	20100000h	00000000h	
MPU_PROG1_MPSAR	021D1C00h	00000000h	21805800h	20A80000h	02200000h	
MPU_PROG2_MPSAR	021C0000h	00000000h	21805C00h	20AC0000h	02210000h	
MPU_PROG3_MPSAR	021C0400h	00000000h	21806000h	00000000h	02220000h	
MPU_PROG4_MPSAR	021C0800h	00000000h	00000000h	00000000h	02230000h	
MPU_PROG5_MPSAR	021D0000h	00000000h	00000000h	00000000h	02240000h	
MPU_PROG6_MPSAR	021D0400h	00000000h	00000000h	00000000h	02250000h	
MPU_PROG7_MPSAR	021D0800h	00000000h	00000000h	00000000h	02530400h	
MPU_PROG8_MPSAR	021D0C00h	00000000h	00000000h	00000000h	02530800h	
MPU_PROG9_MPSAR	021D1000h	00000000h	00000000h	00000000h	02530C00h	
MPU_PROG10_MPSAR	021D1400h	00000000h	00000000h	00000000h	02531000h	
MPU_PROG11_MPSAR	00000000h	00000000h	00000000h	00000000h	02531400h	
MPU_PROG12_MPSAR	00000000h	00000000h	00000000h	00000000h	00000000h	
MPU_PROG13_MPSAR	00000000h	00000000h	00000000h	00000000h	00000000h	
MPU_PROG14_MPSAR	00000000h	00000000h	00000000h	00000000h	00000000h	
MPU_PROG15_MPSAR	00000000h	00000000h	00000000h	00000000h	00000000h	

Table 3-47. Register Call Summary for MPU_PROGx_MPSAR

MPU Registers • MPU Registers: [0][1][2][3]
Memory Protection Unit Functional Description • Protection Check: [0][1] • Protection of the MPU Configuration Registers: [0] • Emulation Considerations: [0] • Permissions of Programmable Address Range: [0] • Memory Protection Ranges: [0]
MPU Programmable Range Registers • MPU_PROGx_MPSAR Register (Offset = 200h + x*16) [reset = See]: [0]

3.2.4.4.2 MPU_PROGx_MPEAR Register (Offset = 204h + x*16) [reset = See Table 3-50]

MPU_PROGx_MPEAR (*x* = 0 to 15) is shown in Figure 3-15 and described in Table 3-49.

The programmable address end register holds the end address for the range. This register is writable by a supervisor entity only. If NS = 0 (non-secure mode) in the associated MPPA register then the register is also only writable by a secure entity. The end address must be aligned on a page boundary.

Table 3-48. MPU_PROGx_MPEAR Instances

Instance	Physical Address
MPU_0	0236 0204h + (x*16)
MPU_1	0236 8204h + (x*16)
MPU_2	0237 0204h + (x*16)
MPU_3	0237 8204h + (x*16)
MPU_4	0238 0204h + (x*16)
MPU_5	0238 8204h + (x*16)
MPU_6	0238 8604h + (x*16)
MPU_7	0238 8A04h + (x*16)
MPU_8	0238 8E04h + (x*16)
MPU_9	0238 9204h + (x*16)
MPU_10	0238 9604h + (x*16)
MPU_11	0238 9A04h + (x*16)
MPU_12	0238 9E04h + (x*16)
MPU_13	0238 A204h + (x*16)
MPU_14	0238 A604h + (x*16)
MPU_15	0238 AA04h + (x*16)
MPU_16	0238 AE04h + (x*16)

Figure 3-15. MPU_PROGx_MPEAR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
END_ADDR																						RESERVED									
R/W-See Table 3-50																						R-3FFh									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 3-49. MPU_PROGx_MPEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	END_ADDR	R/W	See Table 3-50	End address for range *x*.
9-0	RESERVED	R	3FFh	Reserved. Each bit is always read as 1.

Table 3-50. Reset Values of the MPU_PROGx_MPEAR Registers (x = 0 to 15)

Register	MPU_0	MPU_1	MPU_2	MPU_3	MPU_4	MPU_5
MPU_PROG0_MPEAR	0289FFFFh	02603FFFh	0292FFFFh	218183FFh	21803FFFh	Reserved
MPU_PROG1_MPEAR	028BFFFFh	02350FFFh	02930FFFh	3FFFFFFFh	7FFFFFFFh	Reserved
MPU_PROG2_MPEAR	028C0FFFh	000003FFh	02931FFFh	000003FFh	000003FFh	Reserved
MPU_PROG3_MPEAR	028C1FFFh	02621FFFh	02932FFFh	27FFFFFFh	000003FFh	Reserved
MPU_PROG4_MPEAR	028C2FFFh	02507FFFh	02933FFFh	2300FFFFh	000003FFh	Reserved
MPU_PROG5_MPEAR	028C33FFh	023103FFh	02934FFFh	2310FFFFh	000003FFh	Reserved
MPU_PROG6_MPEAR	028C37FFh	022603FFh	02935FFFh	000003FFh	000003FFh	Reserved
MPU_PROG7_MPEAR	02BFFFFFh	025303FFh	0293FFFFh	000003FFh	000003FFh	Reserved
MPU_PROG8_MPEAR	000003FFh	0232F7FFh	000003FFh	000003FFh	000003FFh	Reserved

Table 3-50. Reset Values of the MPU_PROGx_MPEAR Registers (x = 0 to 15) (continued)

Register	MPU_0	MPU_1	MPU_2	MPU_3	MPU_4	MPU_5
MPU_PROG9_MPEAR	000003FFh	021D23FFh	000003FFh	000003FFh	000003FFh	Reserved
MPU_PROG10_MPEAR	000003FFh	024C03FFh	000003FFh	000003FFh	000003FFh	Reserved
MPU_PROG11_MPEAR	000003FFh	024C07FFh	000003FFh	000003FFh	000003FFh	Reserved
MPU_PROG12_MPEAR	000003FFh	000003FFh	000003FFh	000003FFh	000003FFh	Reserved
MPU_PROG13_MPEAR	000003FFh	000003FFh	000003FFh	000003FFh	000003FFh	Reserved
MPU_PROG14_MPEAR	000003FFh	0260BFFFh	000003FFh	000003FFh	000003FFh	Reserved
MPU_PROG15_MPEAR	000003FFh	026203FFh	000003FFh	000003FFh	000003FFh	Reserved
Register	**MPU_6**	**MPU_7**	**MPU_8**	**MPU_9**	**MPU_10**	**MPU_11**
MPU_PROG0_MPEAR	000003FFh	000003FFh	02567FFFh	02601FFFh	02341FFFh	02443FFFh
MPU_PROG1_MPEAR	000003FFh	000003FFh	000003FFh	000003FFh	02343FFFh	030FFFFFh
MPU_PROG2_MPEAR	218043FFh	000003FFh	000003FFh	000003FFh	02345FFFh	000003FFh
MPU_PROG3_MPEAR	218047FFh	000003FFh	000003FFh	000003FFh	023463FFh	000003FFh
MPU_PROG4_MPEAR	21804BFFh	02323FFFh	000003FFh	000003FFh	000003FFh	000003FFh
MPU_PROG5_MPEAR	218053FFh	02329FFFh	000003FFh	000003FFh	000003FFh	000003FFh
MPU_PROG6_MPEAR	000003FFh	0269FFFFh	000003FFh	000003FFh	01E83FFFh	000003FFh
MPU_PROG7_MPEAR	000003FFh	0263FFFFh	000003FFh	000003FFh	0255FFFFh	000003FFh
MPU_PROG8_MPEAR	000003FFh	0259FFFFh	000003FFh	000003FFh	021C6FFFh	000003FFh
MPU_PROG9_MPEAR	000003FFh	0265FFFFh	000003FFh	000003FFh	021CFFFFh	000003FFh
MPU_PROG10_MPEAR	000003FFh	02940FFFh	000003FFh	000003FFh	026407FFh	000003FFh
MPU_PROG11_MPEAR	000003FFh	000003FFh	000003FFh	000003FFh	04FFFFFFh	000003FFh
MPU_PROG12_MPEAR	000003FFh	000003FFh	000003FFh	000003FFh	021E07FFh	000003FFh
MPU_PROG13_MPEAR	000003FFh	000003FFh	000003FFh	000003FFh	023307FFh	000003FFh
MPU_PROG14_MPEAR	000003FFh	000003FFh	000003FFh	000003FFh	026407FFh	000003FFh
MPU_PROG15_MPEAR	000003FFh	000003FFh	000003FFh	000003FFh	000003FFh	000003FFh
Register	**MPU_12**	**MPU_13**	**MPU_14**	**MPU_15**	**MPU_16**	
MPU_PROG0_MPEAR	021D1BFFh	01E843FFh	218057FFh	201FFFFFh	000003FFh	
MPU_PROG1_MPEAR	021D1FFFh	000003FFh	21805BFFh	20ABFFFFh	022003FFh	
MPU_PROG2_MPEAR	021C03FFh	000003FFh	21805FFFh	20AFFFFFh	022103FFh	
MPU_PROG3_MPEAR	021C07FFh	000003FFh	218063FFh	000003FFh	022203FFh	
MPU_PROG4_MPEAR	021C0BFFh	000003FFh	000003FFh	000003FFh	022303FFh	
MPU_PROG5_MPEAR	021D03FFh	000003FFh	000003FFh	000003FFh	022403FFh	
MPU_PROG6_MPEAR	021D07FFh	000003FFh	000003FFh	000003FFh	022503FFh	
MPU_PROG7_MPEAR	021D0BFFh	000003FFh	000003FFh	000003FFh	025307FFh	
MPU_PROG8_MPEAR	021D0FFFh	000003FFh	000003FFh	000003FFh	02530BFFh	
MPU_PROG9_MPEAR	021D13FFh	000003FFh	000003FFh	000003FFh	02530FFFh	
MPU_PROG10_MPEAR	021D17FFh	000003FFh	000003FFh	000003FFh	025313FFh	
MPU_PROG11_MPEAR	000003FFh	000003FFh	000003FFh	000003FFh	025317FFh	
MPU_PROG12_MPEAR	000003FFh	000003FFh	000003FFh	000003FFh	000003FFh	
MPU_PROG13_MPEAR	000003FFh	000003FFh	000003FFh	000003FFh	000003FFh	
MPU_PROG14_MPEAR	000003FFh	000003FFh	000003FFh	000003FFh	000003FFh	
MPU_PROG15_MPEAR	000003FFh	000003FFh	000003FFh	000003FFh	000003FFh	

Table 3-51. Register Call Summary for MPU_PROGx_MPEAR

MPU Registers • MPU Registers: [0][1][2][3]
Memory Protection Unit Functional Description • Protection Check: [0][1] • Protection of the MPU Configuration Registers: [0] • Emulation Considerations: [0] • Permissions of Programmable Address Range: [0] • Memory Protection Ranges: [0]
MPU Programmable Range Registers • MPU_PROGx_MPEAR Register (Offset = 204h + x*16) [reset = See]: [0]

3.2.4.4.3 MPU_PROGx_MPPAR Register (Offset = 208h + x*16) [reset = See Table 3-54]

MPU_PROGx_MPPAR (*x* = 0 to 15) is shown in Figure 3-16 and described in Table 3-53.

This register holds the permissions for the region. It is writable only by a non-debug supervisor entity. If NS = 0 (secure mode) then the register is also only writable by a non-debug secure entity. The NS bit is only writable by a non-debug secure entity. For debug accesses the register is writable only when NS = 1 or EMU = 1.

Table 3-52. MPU_PROGx_MPPAR Instances

Instance	Physical Address
MPU_0	0236 0208h + (x*16)
MPU_1	0236 8208h + (x*16)
MPU_2	0237 0208h + (x*16)
MPU_3	0237 8208h + (x*16)
MPU_4	0238 0208h + (x*16)
MPU_5	0238 8208h + (x*16)
MPU_6	0238 8608h + (x*16)
MPU_7	0238 8A08h + (x*16)
MPU_8	0238 8E08h + (x*16)
MPU_9	0238 9208h + (x*16)
MPU_10	0238 9608h + (x*16)
MPU_11	0238 9A08h + (x*16)
MPU_12	0238 9E08h + (x*16)
MPU_13	0238 A208h + (x*16)
MPU_14	0238 A608h + (x*16)
MPU_15	0238 AA08h + (x*16)
MPU_16	0238 AE08h + (x*16)

Figure 3-16. MPU_PROGx_MPPAR Register

31	30	29	28	27	26	25	24
RESERVED						AID15	AID14
R-See Table 3-54						R/W-See Table 3-54	R/W-See Table 3-54
23	22	21	20	19	18	17	16
AID13	AID12	AID11	AID10	AID9	AID8	AID7	AID6
R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54
15	14	13	12	11	10	9	8
AID5	AID4	AID3	AID2	AID1	AID0	AIDX	RESERVED
R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54	R-See Table 3-54
7	6	5	4	3	2	1	0
NS	EMU	SR	SW	SX	UR	UW	UX
R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54	R/W-See Table 3-54

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 3-53. MPU_PROGx_MPPAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	See Table 3-54	Reserved. Always reads as 0.

Table 3-53. MPU_PROGx_MPPAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	AID15	R/W	See Table 3-54	Controls permission check of ID = 15 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.
24	AID14	R/W	See Table 3-54	Controls permission check of ID = 14 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.
23	AID13	R/W	See Table 3-54	Controls permission check of ID = 13 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.
22	AID12	R/W	See Table 3-54	Controls permission check of ID = 12 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.
21	AID11	R/W	See Table 3-54	Controls permission check of ID = 11 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.
20	AID10	R/W	See Table 3-54	Controls permission check of ID = 10 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.
19	AID9	R/W	See Table 3-54	Controls permission check of ID = 9 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.
18	AID8	R/W	See Table 3-54	Controls permission check of ID = 8 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.
17	AID7	R/W	See Table 3-54	Controls permission check of ID = 7 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.
16	AID6	R/W	See Table 3-54	Controls permission check of ID = 6 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.
15	AID5	R/W	See Table 3-54	Controls permission check of ID = 5 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.
14	AID4	R/W	See Table 3-54	Controls permission check of ID = 4 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.
13	AID3	R/W	See Table 3-54	Controls permission check of ID = 3 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.
12	AID2	R/W	See Table 3-54	Controls permission check of ID = 2 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.
11	AID1	R/W	See Table 3-54	Controls permission check of ID = 1 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.
10	AID0	R/W	See Table 3-54	Controls permission check of ID = 0 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.
9	AIDX	R/W	See Table 3-54	Controls permission check of ID >15 • 0 = AID is not checked for permissions. • 1 = AID is checked for permissions.

Table 3-53. MPU_PROGx_MPPAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	RESERVED	R	See Table 3-54	Reserved. Always reads as 0.
7	NS	R/W	See Table 3-54	Non-secure access permission • 0 = Only secure access allowed. • 1 = Non-secure access allowed.
6	EMU	R/W	See Table 3-54	Emulation (debug) access permission. This bit is ignored if NS = 1 • 0 = Debug access not allowed. • 1 = Debug access allowed.
5	SR	R/W	See Table 3-54	Supervisor Read permission • 0 = Access not allowed. • 1 = Access allowed.
4	SW	R/W	See Table 3-54	Supervisor Write permission • 0 = Access not allowed. • 1 = Access allowed.
3	SX	R/W	See Table 3-54	Supervisor Execute permission • 0 = Access not allowed. • 1 = Access allowed.
2	UR	R/W	See Table 3-54	User Read permission • 0 = Access not allowed. • 1 = Access allowed.
1	UW	R/W	See Table 3-54	User Write permission • 0 = Access not allowed. • 1 = Access allowed.
0	UX	R/W	See Table 3-54	User Execute permission 0 = Access not allowed. 1 = Access allowed.

Table 3-54. Reset Values of the MPU_PROGx_MPPAR Registers (x = 0 to 15)

Register	MPU_0	MPU_1	MPU_2	MPU_3	MPU_4	MPU_5
MPU_PROG0_MPPAR	03FFFC30h	03FFFCB6h	03FFFC30h	03FFFCB6h	03FFFCB6h	Reserved
MPU_PROG1_MPPAR	03FFFC30h	03FFFCB0h	03FFFC30h	03FFFCB6h	03FFFCB6h	Reserved
MPU_PROG2_MPPAR	03FFFC30h	03FFFCBFh	03FFFC30h	03FFFCBFh	03FFFCBFh	Reserved
MPU_PROG3_MPPAR	03FFFC30h	03FFFCB0h	03FFFCB6h	03FFFCB6h	03FFFCBFh	Reserved
MPU_PROG4_MPPAR	03FFFC70h	03FFFCB0h	03FFFC30h	03FFFCB6h	03FFFCBFh	Reserved
MPU_PROG5_MPPAR	03FFFCB6h	03FFFCB0h	03FFFC30h	03FFFCBFh	03FFFCBFh	Reserved
MPU_PROG6_MPPAR	03FFFC70h	03FFFCB6h	03FFFC30h	03FFFCBFh	03FFFCBFh	Reserved
MPU_PROG7_MPPAR	03FFFCB6h	03FFFCB6h	03FFFC00h	03FFFCBFh	03FFFCBFh	Reserved
MPU_PROG8_MPPAR	03FFFCBFh	03FFFCB0h	03FFFCBFh	03FFFCBFh	03FFFCBFh	Reserved
MPU_PROG9_MPPAR	03FFFCBFh	03FFFC30h	03FFFCBFh	03FFFCBFh	03FFFCBFh	Reserved
MPU_PROG10_MPPAR	03FFFCBFh	03FFFCB0h	03FFFCBFh	03FFFCBFh	03FFFCBFh	Reserved
MPU_PROG11_MPPAR	03FFFCBFh	03FFFCB0h	03FFFCBFh	03FFFCBFh	03FFFCBFh	Reserved
MPU_PROG12_MPPAR	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	Reserved
MPU_PROG13_MPPAR	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	Reserved
MPU_PROG14_MPPAR	03FFFCBFh	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCBFh	Reserved
MPU_PROG15_MPPAR	03FFFCBFh	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCBFh	Reserved
Register	**MPU_6**	**MPU_7**	**MPU_8**	**MPU_9**	**MPU_10**	**MPU_11**
MPU_PROG0_MPPAR	03FFFCBFh	03FFFCBFh	03FFFCB6h	03FFFCB6h	03FFFCB6h	03FFFCB6h
MPU_PROG1_MPPAR	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCB6h	03FFFCB6h
MPU_PROG2_MPPAR	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCB6h	03FFFCBFh

Table 3-54. Reset Values of the MPU_PROGx_MPPAR Registers (x = 0 to 15) (continued)

Register	MPU_0	MPU_1	MPU_2	MPU_3	MPU_4	MPU_5
MPU_PROG3_MPPAR	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCB6h	03FFFCBFh
MPU_PROG4_MPPAR	03FFFCB6h	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh
MPU_PROG5_MPPAR	03FFFCB6h	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh
MPU_PROG6_MPPAR	03FFFCBFh	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCB6h	03FFFCBFh
MPU_PROG7_MPPAR	03FFFCBFh	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCB6h	03FFFCBFh
MPU_PROG8_MPPAR	03FFFCBFh	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCB6h	03FFFCBFh
MPU_PROG9_MPPAR	03FFFCBFh	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCB6h	03FFFCBFh
MPU_PROG10_MPPAR	03FFFCBFh	03FFFCB6h	03FFFCBFh	03FFFCBFh	00083CB6h	03FFFCBFh
MPU_PROG11_MPPAR	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCB6h	03FFFCBFh
MPU_PROG12_MPPAR	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCB6h	03FFFCBFh
MPU_PROG13_MPPAR	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCB6h	03FFFCBFh
MPU_PROG14_MPPAR	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	03F7C080h	03FFFCBFh
MPU_PROG15_MPPAR	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh

Register	MPU_12	MPU_13	MPU_14	MPU_15	MPU_16	
MPU_PROG0_MPPAR	03FFFCB6h	03FFFCB0h	03FFFCB6h	03FFFCB6h	03FFFCBFh	
MPU_PROG1_MPPAR	03FFFCB6h	03FFFCBFh	03FFFCB6h	03FFFCB6h	03FFFCB6h	
MPU_PROG2_MPPAR	03FFFCB6h	03FFFCBFh	03FFFCB6h	03FFFCB6h	03FFFCB6h	
MPU_PROG3_MPPAR	03FFFCB6h	03FFFCBFh	03FFFCB6h	03FFFCBFh	03FFFCB6h	
MPU_PROG4_MPPAR	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCB6h	
MPU_PROG5_MPPAR	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCB6h	
MPU_PROG6_MPPAR	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCB6h	
MPU_PROG7_MPPAR	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCB6h	
MPU_PROG8_MPPAR	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCB6h	
MPU_PROG9_MPPAR	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCB6h	
MPU_PROG10_MPPAR	03FFFCB6h	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCB6h	
MPU_PROG11_MPPAR	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCB6h	
MPU_PROG12_MPPAR	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	
MPU_PROG13_MPPAR	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	
MPU_PROG14_MPPAR	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	
MPU_PROG15_MPPAR	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	03FFFCBFh	

Table 3-55. Register Call Summary for MPU_PROGx_MPPAR

MPU Registers • MPU Registers: [0][1][2][3]
Memory Protection Unit Functional Description • Protection Check: [0][1][2][3][4][5][6][7][8] • Permissions of Programmable Address Range: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15] • Memory Protection Ranges: [0] • Protection of the MPU Configuration Registers: [0][1] • DSP L1/L2 Cache Controller Read Accesses: [0][1][2] • Emulation Considerations: [0][1][2]
MPU Programmable Range Registers • MPU_PROGx_MPPAR Register (Offset = 208h + x*16) [reset = See]: [0]

3.2.4.5 MPU Fault Registers

3.2.4.5.1 MPU_FLTADDRR Register (Offset = 300h) [reset = 0h]

MPU_FLTADDRR is shown in Figure 3-17 and described in Table 3-57.

The fault address register holds the address of the first protection fault transfer.

Table 3-56. MPU_FLTADDRR Instances

Instance	Physical Address
MPU_0	0236 0300h
MPU_1	0236 8300h
MPU_2	0237 0300h
MPU_3	0237 8300h
MPU_4	0238 0300h
MPU_5	0238 8300h
MPU_6	0238 8700h
MPU_7	0238 8B00h
MPU_8	0238 8F00h
MPU_9	0238 9300h
MPU_10	0238 9700h
MPU_11	0238 9B00h
MPU_12	0238 9F00h
MPU_13	0238 A300h
MPU_14	0238 A700h
MPU_15	0238 AB00h
MPU_16	0238 AF00h

Figure 3-17. MPU_FLTADDRR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 3-57. MPU_FLTADDRR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R	0h	Memory address of fault.

Table 3-58. Register Call Summary for MPU_FLTADDRR

MPU Registers • MPU Registers: [0][1][2][3]
MPU Fault Registers • MPU_FLTADDRR Register (Offset = 300h) [reset = 0h]: [0]
Memory Protection Unit Functional Description • MPU Interrupt Requests: [0][1]

3.2.4.5.2 MPU_FLTSTAT Register (Offset = 304h) [reset = --000000h]

MPU_FLTSTAT is shown in Figure 3-18 and described in Table 3-60.

The fault status register holds the status and attributes of the first protection fault transfer.

Table 3-59. MPU_FLTSTAT Instances

Instance	Physical Address
MPU_0	0236 0304h
MPU_1	0236 8304h
MPU_2	0237 0304h
MPU_3	0237 8304h
MPU_4	0238 0304h
MPU_5	0238 8304h
MPU_6	0238 8704h
MPU_7	0238 8B04h
MPU_8	0238 8F04h
MPU_9	0238 9304h
MPU_10	0238 9704h
MPU_11	0238 9B04h
MPU_12	0238 9F04h
MPU_13	0238 A304h
MPU_14	0238 A704h
MPU_15	0238 AB04h
MPU_16	0238 AF04h

Figure 3-18. MPU_FLTSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R--h							
23	22	21	20	19	18	17	16
MSTID							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			PRIVID				RESERVED
R-0h			R-0h				R-0h
7	6	5	4	3	2	1	0
NS	RESERVED	TYPE					
R-0h	R-0h	R-0h					

LEGEND: R = Read Only; *-n* = value after reset

Table 3-60. MPU_FLTSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	-h	Reserved.
23-16	MSTID	R	0h	Master ID of fault transfer.
15-13	RESERVED	R	0h	Reserved. Always reads as 0.
12-9	PRIVID	R	0h	Privilege ID of fault transfer.
8	RESERVED	R	0h	Reserved. Always reads as 0.
7	NS	R	0h	Security level of fault transfer. • 0 = Secure access • 1 = Non-secure access

Table 3-60. MPU_FLTSTAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	RESERVED	R	0h	Reserved. Always reads as 0.
5-0	TYPE	R	0h	Fault type. • 100000b Supervisor read fault • 010000b Supervisor write fault • 001000b Supervisor execute fault • 000100b User read fault • 000010b User write fault • 000001b User execute fault • 111111b Relaxed cache line fill fault • 010010b Relaxed cache write back fault • 000000b No fault

Table 3-61. Register Call Summary for MPU_FLTSTAT

MPU Registers • MPU Registers: [0][1][2][3]
MPU Fault Registers • MPU_FLTCLR Register (Offset = 308h) [reset = 0h]: [0][1] • MPU_FLTSTAT Register (Offset = 304h) [reset = --000000h]: [0]
Memory Protection Unit Functional Description • MPU Interrupt Requests: [0][1]

3.2.4.5.3 MPU_FLTCLR Register (Offset = 308h) [reset = 0h]

MPU_FLTCLR is shown in Figure 3-19 and described in Table 3-63.

This register allows software to clear the current fault so that another can be captured in the MPU_FLTSTAT register. Only the MPU_FLTSTAT[5-0] TYPE field is cleared when 1 is written to the MPU_FLTCLR[0] CLEAR bit. Clearing this bit is also required for being able to generate another interrupts.

Table 3-62. MPU_FLTCLR Instances

Instance	Physical Address
MPU_0	0236 0308h
MPU_1	0236 8308h
MPU_2	0237 0308h
MPU_3	0237 8308h
MPU_4	0238 0308h
MPU_5	0238 8308h
MPU_6	0238 8708h
MPU_7	0238 8B08h
MPU_8	0238 8F08h
MPU_9	0238 9308h
MPU_10	0238 9708h
MPU_11	0238 9B08h
MPU_12	0238 9F08h
MPU_13	0238 A308h
MPU_14	0238 A708h
MPU_15	0238 AB08h
MPU_16	0238 AF08h

Figure 3-19. MPU_FLTCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CLEAR
R-0h							W-0h

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 3-63. MPU_FLTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved. Always reads as 0.
0	CLEAR	W	0h	Command to clear the current fault. Writing 0 has no effect. • 0 = No effect • 1 = Clear the current fault

Table 3-64. Register Call Summary for MPU_FLTCLR

MPU Registers • MPU Registers: [0][1][2][3]
MPU Fault Registers • MPU_FLTCLR Register (Offset = 308h) [reset = 0h]: [0][1]
Memory Protection Unit Functional Description • MPU Interrupt Requests: [0]

Initialization

This chapter describes the non-secure features of the on-chip ROM boot loader (BootROM).

Topic **Page**

4.1 Overview

ROM Boot Loader (BootROM) is software code that resides in the on-chip read only memory (ROM) to assist the customer in transferring and executing their application code. In order to accommodate various system scenarios, the BootROM supports several boot modes. These boot modes can be broadly classified as:

- Host boot modes
- Memory boot modes.

During a host boot, the device is configured to receive code from a host via the selected interface. Either the host writes the application code directly into internal memory or the BootROM receives the application code on the selected interface and stores it in internal memory.

During a memory boot, the device transfers code from non-volatile memory to internal memory for execution.

In all boot modes, the entire boot operation can be partitioned into two sections:

- Initialization phase
- Boot process.

During initialization, the BootROM configures the device resources as needed to support the boot process. The resources used depend on the boot mode requirements. During the boot process the boot image is loaded into device memory and executed. The actions performed during the initialization and boot processes depend on the following factors:

- The trigger that initiated the boot operation
- The boot mode and location of the boot image (host or memory) as defined by the configuration specified on the BOOTMODE pins of the device.

Main configuration source for boot after power-up are the BOOTMODE pins sampled automatically after reset release and stored in the device status register DEVSTAT. At BootROM startup, these pins are read from the DEVSTAT register to create the boot peripheral list and the boot configuration tables used later to initialize and startup the PLLs and boot peripherals.

This chapter covers:

- The different triggers that can initiate the boot operation
- The initialization process
- The boot process based on the location of the image
- The specific boot process features for different boot modes.

If the image is in an external host, the boot process varies depending on the host knowledge of the boot device memory map.

The BootROM also provides a multi-stage boot mechanism.

4.1.1 Bootloader Modes

Table 4-1 shows the BootROM boot peripherals supported in the device.

Table 4-1. BootROM Boot Modes

Boot Peripheral	Peripheral Instances	Media/Notes
QSPI	1	QSPI flash
SPI	4	EEPROM
UART	3	External host
GPMC	1	NOR flash
I^2C	3	I^2C master boot - EEPROM I^2C slave boot - External host
Ethernet	1	BOOTP mode
PCIE	1	
MMCSD	2	Embedded MMC (eMMC) or SD card
USB	2	USB boot - device mode boot using DFU (device firmware update)

4.2 BootROM Initialization Phase

As previously mentioned, the boot process can be divided into two steps. The first step is the initialization process, which depends on the type of reset that triggers the boot. The second step is the boot mode specific process (see Section 4.3, *Boot Process, Boot Modes and Pin Usage*).

The start address of the ARMSS BootROM is 0x00000000.

4.2.1 Reset Types

In device, resets are used as the trigger for initiating the boot process and the boot process varies based on the type of the reset. There are four types of reset supported in the device architecture:

- Power-on reset (POR)
- Hard reset (warm reset initiated by either hardware or software)
- Soft reset (a subset of hard reset)
- Local reset to DSP or PMMC.

The first three types of reset are considered global resets because they affect the entire device, while the local reset affects only DSP or PMMC.

NOTE: The ARMSS may perform local reset of the DSP, but the ARMSS cannot be reset by a local reset.

For local reset, the boot process is not triggered. For further details on the reset types, see Section 5.3, *Reset Management*.

Irrespective of the global reset type, the boot process is executed by the respective boot master Core 0.

4.2.1.1 BOOTMODE Pins

External pins on the device are used at device power-up to allow customers to select the desired boot mode. There are also pins that specify some of the configuration parameters used during that boot mode. The value of these pins is latched into the device status register DEVSTAT during POR. Some boot modes also include a Minimum Boot Pin Select pin (MIN) that can be used to select predifined in ROM parameters instead of selecting them with the all BOOTMODE pins.

4.2.2 Initialization Flow

This section is divided into two sub-sections:

- Initialization Process After Power On Reset (POR)
- Initialization Process After Hard Reset or Soft Reset.

4.2.2.1 Initialization Process After Power On Reset

4.2.2.1.1 Device Initialization

POR resets the entire device. Everything on the device is reset to its default state. POR is initiated by either the PORn or RESETFULLn external pins. The PORn pin is asserted during power up of the device while the RESETFULLn pin can be asserted by a host to reset the device after it is already powered up. PORn causes all peripherals to be initialized to their default states and boot processing to be started. The state of the BOOTMODE pins are latched into the device status register (DEVSTAT) at POR. The BootROM uses the values in DEVSTAT to select how boot is performed. All initialization and boot processing is performed by ARMSS Core0.

The BootROM uses the boot configuration information from the DEVSTAT register to determine the initialization flow to perform. Initialization executed by the BootROM includes:

- The BootROM enables reset isolation in all peripherals that support it. The power state of these peripherals is not changed. Section 5.3, *Reset Management* lists the peripherals that support reset isolation.

- The BootROM enables the power and clock domains for any peripherals required during boot.
- Various PLLs may be configured depending on values specified in BOOTMODE pins. BOOTMODE pins and DEVSTAT register will contain information indicating the PLL, the input clock frequency, and when that PLL should be configured.
  - For all modes, except for SLEEP, I2C, and SPI where it is optional, the MAIN PLL is programmed so that MAIN PLL runs at the frequency indicated by the DEVICE_SPEED field in the EFUSE_BOOTROM register.
  - When the ARMSS is the boot master and MAIN PLL is locked, the ARM PLL is programmed so that the output of ARM PLL provides the frequency indicated by the ARM_SPEED field in the EFUSE_BOOTROM register.
- All interrupts are disabled except for IPC interrupts and the host interrupts that are used for an external host boot modes (PCIe)
- DSP executes an IDLE command.
- ARMSS cache is enabled
- MMU is enabled
- The BootROM uses the boot configuration information in DEVSTAT to setup and initialize a boot parameter table that is used to control the boot process. This table is stored in MSMC SRAM. Some information in the table is initialized based on the configuration parameters in DEVSTAT while the remaining information is default values based only on the boot mode. The format of the table varies depending on the boot mode. All start with a few entries that are common to all boot modes. For information about the boot parameter table see Section 4.3.3, *Boot Parameter Tables*.

4.2.2.2 Initialization Process After Hard Reset or Soft Reset

As previously mentioned, a POR reset initialized by a PORn or RESETFULLn will sample the BOOTMODE pins. The sampled information is stored internally and written to the DEVSTAT register.

After Soft/Hard resets,BootROM will carry out the same initialization process mentioned in Section 4.2.2.1, *Initialization Process After Power On Reset* followed by the boot specific process.

The only difference in this boot process compared to one triggered by the POR reset is that the BOOTMODE pins are not sampled to update the DEVSTAT register. In addition, the reset controller will not reset any peripherals that are reset-isolated in the device.

NOTE: Any update to the DEVSTAT register made by software, prior to hard/soft reset, will be lost.

If the reset is a soft reset, some of the registers and the memory are preserved in addition to the modules that are not reset by the hard reset. In addition, the reset controller will not reset any peripherals that are reset-isolated in the device.

Refer to Section 5.3, *Reset Management* for detailed descriptions of the different reset types.

4.2.3 Multi-Stage Boot

The BootROM also provides a multi-stage boot mechanism. This feature can be used when a single stage boot does not provide the desired flexibility. The typical use case for a multi-stage boot is to perform a basic boot using the boot mode and parameter settings available in DEVSTAT, let the downloaded code modify the boot parameter table to select new boot parameters or a new boot mode, and then branch to the boot re-entry point to perform a second boot.

The initial boot parameter table is built by the BootROM as described in Section 4.3, *Boot Process, Boot Modes and Pin Usage*. The boot parameter table is not modified by the BootROM when the re-entry point is used.

4.3 Boot Process, Boot Modes and Pin Usage

Because the BootROM supports many boot modes, a set of BOOTMODE pins is used to select a specific boot mode and also provide a minimal configuration for the specified boot mode.

The values on these pins are latched into the Device Status Register (DEVSTAT) as the device comes out of global reset. For more information on these BOOTMODE pins see Section 4.3.2, *BOOTMODE Pins Description*.

NOTE: The following system conditions must be met to perform device boot:

- The USB transceivers (USB2.0), internal to device, are powered upon reset.
- The SD card cage must be appropriately powered before entering the SD card boot feature on any reset.
- Devices must be appropriately powered and be up and ready (reset completed) at device startup:
  - eMMC
  - QSPI
  - NOR
  - Ethernet PHY

Failing to meet these requirements may result in boot fail and performing a backup boot (if available for that mode).

4.3.1 Boot Process Flow

The BootROM boot process flow is shown in Figure 4-1.

Figure 4-1. Boot Process

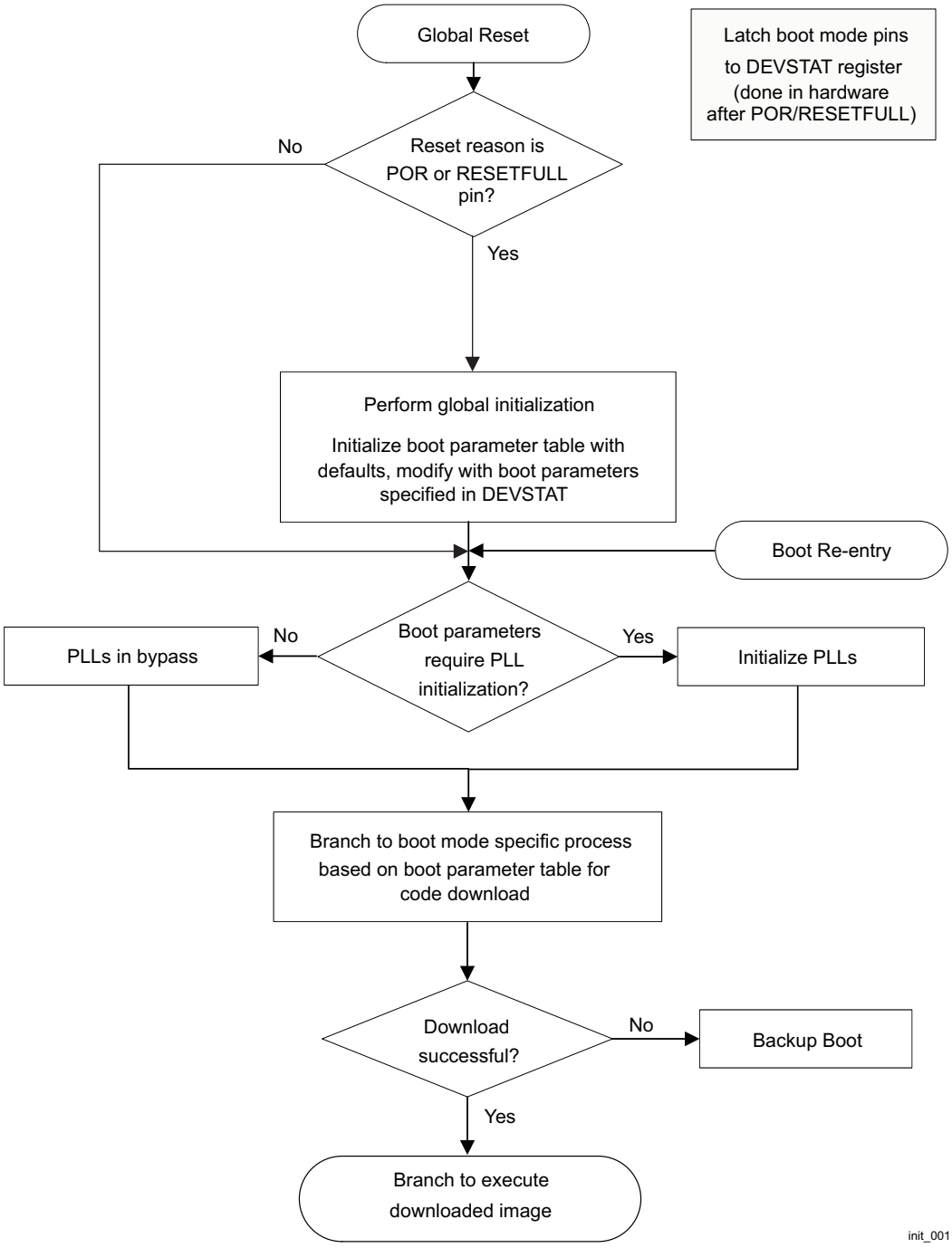
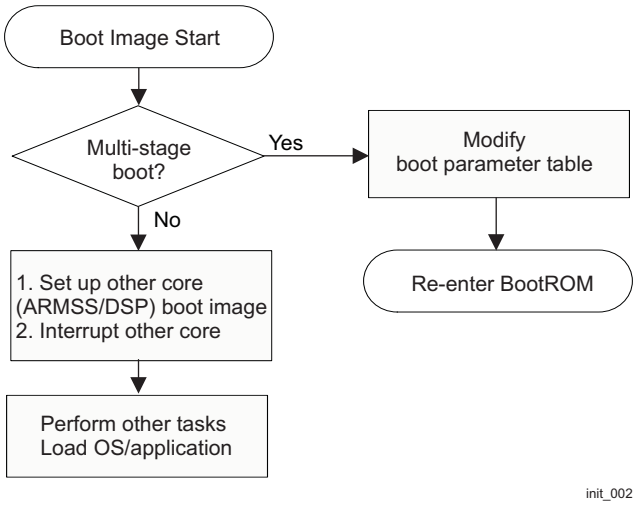


Figure 4-2 describes the external bootloader typical tasks.

Figure 4-2. External Bootloader Tasks



4.3.2 BOOTMODE Pins Description

BOOTMODE pins provide means to select the boot mode before the device is powered up. They can be divided into the following categories:

- BOOTMODE[3:0] – Select the requested boot mode, that is, the peripheral/memory to boot from.
- BOOTMODE[4] – This is the minimum (MIN) configuration pin. MIN pin allows to not to set all [15:0] BOOTMODE pins, and thus saving DIP switches or pull-ups/pull-downs on the board. A default value, stored in ROM code, is loaded in place of certain BOOTMODE pin values. See the particular boot mode section for the MIN default values.
- BOOTMODE[15:6] – These pins provide additional boot options and are used in conjuction with the boot mode selected. See Figure 4-3 and the corresponding boot mode section.

NOTE: DEVSTAT register reflects the BOOTMODE values sampled after last cold reset.

It is important to keep in mind that BOOTMODE[15:0] **pins** map to DEVSTAT[16-1] BOOTMODE **bits** of the DEVSTAT register.

NOTE: The BootROM will try to proceed with the closest supported configuration at a detection of invalid configuration.

Figure 4-3 shows the BOOTMODE pin mapping to their functions in the different boot modes.

Figure 4-3. BOOTMODE Pin Mapping

<table>
<tr><th colspan="15">BOOTMODE Pins</th><th rowspan="2">Boot Mode</th></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8-6</th><th>5</th><th>4</th><th>3-0</th></tr>
<tr><td colspan="2">BACKUP</td><td>TOUT</td><td colspan="2">I²C PORT</td><td>I²C SLAVE</td><td>PLL</td><td rowspan="2">REF CLK</td><td rowspan="16">Boot Master (0 = ARMSS)</td><td rowspan="16">MIN</td><td>0000</td><td>SLEEP/I²C SLAVE Section 4.3.2.1</td></tr>
<tr><td colspan="2">BACKUP</td><td>TOUT</td><td colspan="4">BAR CONFIG</td><td>0001</td><td>PCIe Section 4.3.2.2</td></tr>
<tr><td colspan="2" rowspan="2">PORT</td><td colspan="2" rowspan="2">BUS ADDR</td><td colspan="3" rowspan="2">PIDX/OFFSET</td><td>000</td><td>0010</td><td>I²C Section 4.3.2.3</td></tr>
<tr><td>REF CLK</td><td>0011</td><td>I²C PLL Section 4.3.2.3</td></tr>
<tr><td colspan="2" rowspan="3">PORT</td><td rowspan="3">CSEL</td><td colspan="2" rowspan="3">WIDTH</td><td colspan="2" rowspan="3">MODE</td><td>PIDX/ OFFSET</td><td>0100</td><td>SPI Section 4.3.2.4</td></tr>
<tr><td rowspan="4">REF CLK</td><td>0101</td><td rowspan="3">SPI PLL Section 4.3.2.5</td></tr>
<tr><td>0110</td></tr>
<tr><td>0111</td></tr>
<tr><td>BACKUP</td><td colspan="2">CSEL</td><td>ADDR WIDTH</td><td colspan="3">COMMAND PIN</td><td>1000</td><td>QSPI 48 MHz Section 4.3.2.6</td></tr>
<tr><td colspan="15"></td><td>1001</td><td>QSPI 96 MHz Section 4.3.2.6</td></tr>
<tr><td colspan="2">ADP/AD/AAD</td><td colspan="2">CSEL</td><td>WAIT</td><td>WIDTH</td><td>BASE</td><td>1010</td><td>XIP Section 4.3.2.7</td></tr>
<tr><td>0</td><td>PHY CFG</td><td>RMII CLK</td><td>S_SPD</td><td>H_DUP</td><td colspan="2">External I/F</td><td>1100</td><td>Ethernet Section 4.3.2.8</td></tr>
<tr><td colspan="3">PHY Frequency SEL</td><td colspan="2">REF_SRC</td><td>D_MOD</td><td>PORT</td><td>1101</td><td>USB Section 4.3.2.9</td></tr>
<tr><td>BACKUP</td><td>B_ACK</td><td>BOOT</td><td colspan="2">Card Type</td><td>1 bit</td><td>PORT</td><td>1110</td><td>MMCSD Section 4.3.2.10</td></tr>
<tr><td>0</td><td colspan="2">BACKUP</td><td colspan="2">Tout</td><td colspan="2">PORT</td><td>1111</td><td>UART Section 4.3.2.11</td></tr>
</table>

4.3.2.1 Sleep/I²C Slave Boot Device Configuration

Figure 4-4 shows the BOOTMODE[15:6] pins assignment to functions when BOOTMODE[3:0] select the Sleep (no boot) or I²C modes.

Figure 4-4. Sleep/I²C Slave Boot Mode Configuration Fields

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BACKUP		TOUT	I²C PORT		I²C SLAVE	PLL	REF CLK			BOOT Master	MIN	0000			

Table 4-2. Sleep/I²C Slave Boot Configuration Field Description

Pins	Field	Description	MIN Default Value
15:14	BACKUP	• 0 = I²C Master boot. I²C PLL mode used if PLL enable is set • 1 = SPI boot used as backup after timeout • 2 = XIP boot used as backup after timeout	-
13	TOUT	• 0 = I²C Slave boot never times out • 1 = I²C Slave boot times out after 1 second	-
12:11	I²C PORT	I²C port to use for slave boot (Port0 - Port2)	-
10	I²C SLAVE	Select Sleep/I²C Slave Boot Mode • 0 = Sleep boot (no boot) • 1 = I²C Slave Boot	-
9	PLL	• 0 = No PLLs are programmed • 1 = Currently Disable PLL(s) are programmed	-
8:6	REF CLK	PLL reference clock. See Table 4-16, *Reference Clock Values*.	0
5	BOOT Master	Boot Master select • 0 = ARMSS is boot master	-
4	MIN	Minimum boot configuration select bit. • 0 = All fields must be independently configured. • 1 = A predetermined set of values is configured	-
3:0	Boot Devices	Boot Devices[3:0]: • 0000 = Sleep or I²C Slave depending on BOOTMODE[10]	-

4.3.2.2 PCIE Boot Device Configuration

Figure 4-5 shows the BOOTMODE[15:6] pins assignment to functions when BOOTMODE[3:0] select the PCIe mode.

Figure 4-5. PCIE Boot Mode Configuration Fields

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BACKUP		TOUT	BAR CONFIG				REF CLK			BOOT Master	MIN	0001			

Table 4-3. PCIe Boot Configuration Field Description

Pins	Field	Description	MIN Default Value
15:14	BACKUP	Backup boot device. Reserved, PCIe boot does not have a backup boot option.	-
13	TOUT	Enable time out: • 0 = Timeout disabled • 1 = Reserved	-
12:9	BAR CONFIG	See Table 4-4, *BAR Config/PCIE Window Sizes*.	0
8:6	REF CLK	PLL reference clock. See Table 4-16, *Reference Clock Values*.	0
5	BOOT Master	Boot Master select • 0 = ARMSS is boot master	-
4	MIN	Minimum boot configuration select bit. • 0 = All fields must be independently configured • 1 = A predetermined set of values is configured	-
3:0	Boot Devices	Boot Devices[3:0]: • 0001 = PCIe boot mode	-

Table 4-4 shows the BAR configuration in PCIe boot mode.

Table 4-4. BAR Config/PCIE Window Sizes

		32 bit Address Translation					64 bit Address Translation	
BAR CONFIG	BAR0	BAR1	BAR2	BAR3	BAR4	BAR5	BAR1/BAR2	BAR3/BAR4
0b0000	PCIE Registers	32	32	32	32	Clone of BAR4		
0b0001		16	16	32	64			
0b0010		16	32	32	64			
0b0011		32	32	32	64			
0b0100		16	16	64	64			
0b0101		16	32	64	64			
0b0110		32	32	64	64			
0b0111		32	32	64	128			
0b1000		64	64	128	256			
0b1001		4	128	128	128			
0b1010		4	128	128	256			
0b1011		4	128	256	256			
0b1100							256	256
0b1101							512	512
0b1110							1024	1024
0b1111							2048	2048

4.3.2.3 I²C Master Boot Device Configuration

Figure 4-6 shows the BOOTMODE[15:6] pins assignment to functions when BOOTMODE[3:0] select the I²C master mode.

Figure 4-6. I²C Master Boot Mode Configuration Fields

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PORT		BUS ADDR		PIDX/OFFSET			RESERVED			BOOT Master	MIN	0010			
PORT		BUS ADDR		PIDX/OFFSET			REF CLK			BOOT Master	MIN	0011			

Table 4-5. I²C Mode Boot Configuration Field Description

Pins	Field	Description	MIN Default Value
15:14	PORT	I²C Port to use during I²C boot.	0
13:12	BUS ADDR	I²C Master bus address to use to access EEPROM. The I²C bus address will be 0x50 + this value.	0
11:9	PIDX/OFFSET	For ARMSS master the initial read is 0x2000 × this value.	0
8:6	REF CLK	PLL reference clock. See Table 4-16, *Reference Clock Values*. Don't care field in no-PLL mode.	0
5	BOOT Master	Boot Master select • 0 = ARMSS is boot master	-
4	MIN	Minimum boot configuration select bit. • 0 = All fields must be independently configured • 1 = A predetermined set of values is configured	-
3:0	Boot Devices	Boot Devices[3:0]: • 0010 = I²C without PLL • 0011 = I²C with PLL	-

4.3.2.4 SPI without PLL Boot Device Configuration

Figure 4-7 shows the BOOTMODE[15:6] pins assignment to functions when BOOTMODE[3:0] select the SPI no-PLL mode.

Figure 4-7. SPI without PLL Boot Mode Configuration Fields

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PORT		CSEL	WIDTH		MODE		PIDX/OFFSET			BOOT Master	MIN	0100			

Table 4-6. SPI without PLL Mode Boot Configuration Field Descriptions

Pins	Field	Description	MIN Default Value
15:14	PORT	SPI Port [0-3] used for boot.	0
13	CSEL	Active boot chip select [0-1].	0
12:11	WIDTH	Addresses bit selection: • 0 = 16 bit SPI addresses • 1 = 24 bit SPI addresses • 2-3 = 32 bit SPI addresses	-
10:9	MODE	Mode Selection: • 0 = Data is output on the rising edge of SPICLK. Input data is latched on the falling edge. • 1 = Data is output one half-cycle before the first rising edge of SPICLK and on subsequent falling edges. Input data is latched on the rising edge of SPICLK. • 2 = Data is output on the falling edge of SPICLK. Input Data is latched on the rising edge. • 3 = Data is output one half-cycle before the first falling edge of SPICLK and on subsequent rising edges. Input data is latched on the falling edge of SPICLK.	-
8:6	PIDX/OFFSET	For ARMSS master boot the offset value is based on the SPI address width as shown in Table 4-7.	0
5	BOOT Master	Boot Master select • 0 = ARMSS is boot master	-
4	MIN	Minimum boot configuration select bit. • 0 = All fields must be independently configured • 1 = A predetermined set of values is configured	-
3:0	Boot Devices	Boot Devices[3:0]: • 0100 = SPI without PLL	-

Table 4-7. SPI Initial Read Address

Address Width (bits)	Read offset
16	0x2000 × OFFSET index
24	0x80000 × OFFSET index
32	0x400000 × OFFSET index

4.3.2.5 SPI with PLL Boot Device Configuration

Figure 4-8 shows the BOOTMODE[15:6] pins assignment to functions when BOOTMODE[3:0] select the SPI with PLL mode.

Figure 4-8. SPI with PLL Boot Mode Configuration Fields

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PORT		CSEL	WIDTH		MODE		REF CLK			BOOT Master	MIN	0101 0110 0111			

Table 4-8. SPI with PLL Mode Boot Configuration Field Description

Pins	Field	Description	MIN Default Value
15:14	PORT	SPI Port [0-3] used for boot.	0
13	CSEL	Active boot chip select [0-1].	0
12:11	WIDTH	Addresses width selection: • 0 = 16-bit SPI addresses • 1 = 24-bit SPI addresses • 2-3 = 32-bit SPI addresses	-
10:9	MODE	Mode Selection: • 0 = Data is output on the rising edge of SPICLK. Input data is latched on the falling edge. • 1 = Data is output one half-cycle before the first rising edge of SPICLK and on subsequent falling edges. Input data is latched on the rising edge of SPICLK. • 2 = Data is output on the falling edge of SPICLK. Input Data is latched on the rising edge. • 3 = Data is output one half-cycle before the first falling edge of SPICLK and on subsequent rising edges. Input data is latched on the falling edge of SPICLK.	-
8:6	REF CLK	PLL reference clock. See Table 4-16, *Reference Clock Values*.	0
5	BOOT Master	Boot Master select • 0 = ARMSS is boot master	-
4	MIN	Minimum boot configuration select bit. • 0 = All fields must be independently configured • 1 = A predetermined set of values is configured	-
3:0	Boot Devices	Boot Devices[3:0]: • 0101 = SPI with PLL, OFFSET index = 0 • 0110 = SPI with PLL, OFFSET index = 1 • 0111 = SPI with PLL, OFFSET index = 2 For ARMSS boot, the offset value is based on the SPI address width as shown in Table 4-7.	-

4.3.2.6 QSPI Boot Device Configuration

Figure 4-9 shows the BOOTMODE[15:6] pins assignment to functions when BOOTMODE[3:0] select the QSPI mode.

Figure 4-9. QSPI Boot Mode Configuration Fields

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BACKUP	CSEL		ADDR WIDTH	COMMAND PIN			REF CLK			BOOT Master	MIN	1000 1001			

Table 4-9. QSPI Mode Boot Configuration Field Description

Pins	Field	Description	MIN Default Value
15	BACKUP	• 0 = USB boot used as backup after timeout • 1 = UART boot used as backup after timeout	-
14:13	CSEL	Active boot chip select (CS0-CS3).	0
12	ADDR WIDTH	Address width selection: • 0 = 24-bit Address width • 1 = 32-bit Address width	-
11:9	COMMAND PIN	QSPI Command and Pin Configuration See Table 4-10, *QSPI Command/Pin Configuration*	0
8:6	REF CLK	PLL reference clock. See Table 4-16, *Reference Clock Values*	0
5	BOOT Master	Boot Master select • 0 = ARMSS is boot master	-
4	MIN	Minimum boot configuration select bit. • 0 = All fields must be independently configured • 1 = A predetermined set of values is configured	-
3:0	Boot Devices	Boot Devices[3:0]: • 1000 = QSPI 48 MHz • 1001 = QSPI 96 MHz	-

Table 4-10. QSPI Command/Pin Configuration

COMMAND PIN Pin Value	CMD Value	CMD Name	Dummy Cycles	Mode Byte	Number of Pins Driven for CMD/Mode	Number of Pins Driven for Address	Number of Pins Used for Data
0	0x0B	Fast read	8	None	1	1	1
1	0x3B	Read dual out (DOR)	8	None	1	1	2
2	0x6B	Read quad out (QOR)	8	None	1	1	4
3	0xBB	Dual I/O Read (DIOR)	8	None	1	2	2
4	0xEB	Quad I/O Read	8	None	1	4	4
5	0xBB	Dual I/O read	4	Yes (sent as 0)	1	2	2
6	0xEB	Quad I/O read	5	Yes (sent as 0)	1	4	4
7	0xEB	QPI read	6	Yes (sent as 0)	4	4	4

4.3.2.7 XIP Boot Device Configuration

For XIP boot the 256-Mbyte address space is divided into four 64-Mbyte chip select regions. A different chip select region size can be selected through a multi-stage boot.

Figure 4-10 shows the BOOTMODE[15:6] pins assignment to functions when BOOTMODE[3:0] select the XIP/NOR mode.

Figure 4-10. XIP Boot Mode Configuration Fields

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADP/AD/AAD Mux		CSEL		WAIT	WIDTH	BASE	REF CLK			BOOT Master	MIN	1010			

Table 4-11. XIP Mode Boot Configuration Field Description

Pins	Field	Description	MIN Default Value
15:14	ADP/AD/AAD	Mode selection: • 0 = ADP mode (address data parallel) • 1 = AD mode (address/data mux) • 2 = AAD mode (address/address/data mux) • 3 = ADP mode	-
13:12	CSEL	Active boot chip select (CS0-CS3).	0
11	WAIT	Enable extended wait monitor: • 0 = disable • 1 = enable	-
10	WIDTH	Bus width selection: • 0 = 16-bit bus width • 1 = 8-bit bus width	-
9	BASE	Base address boot selection: • 0 = Boot from base address for chip select • 1 = Boot from base address + 4 Mbyte for chip select	-
8:6	REF CLK	PLL reference clock. See Table 4-16, *Reference Clock Values*.	0
5	BOOT Master	Boot Master select • 0 = ARMSS is boot master	-
4	MIN	Minimum boot configuration select bit. • 0 = All fields must be independently configured • 1 = A predetermined set of values is configured	-
3:0	Boot Devices	Boot Devices[3:0]: • 1010 = XIP	-

NOTE: Table 4-26 defines which GPMC_WAIT[x] input and its polarity is used during XIP boot based on the state of the CSEL and WAIT configuration inputs.

4.3.2.8 Ethernet Boot Device Configuration

Figure 4-11 shows the BOOTMODE[15:6] pins assignment to functions when BOOTMODE[3:0] select the Ethernet mode.

Figure 4-11. Ethernet Boot Mode Configuration Fields

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Phy CFG	RMII CLK	S_SPD	H_DUP	External I/F		REF CLK			BOOT Master	MIN	1100			

Table 4-12. Ethernet Mode Boot Configuration Field Description

Pins	Field	Description	MIN Default Value
15	RSVD	Reserved. Set to 0.	-
14	PHY CFG	Physical configuration: • 0 = Do not read RMII speed duplex configuration from the PHY • 1 = Read the RMII speed duplex configuration from the PHY	-
13	RMII CLK	RMII Clock configuration: • 0 = RMII ref clock configured as an output • 1 = RMII ref clock configured as an input	-
12	S_SPD	Speed configuration: • 0 = Run at highest speed (1000 or 100 Mbps) • 1 = Run at lowest speed (100 or 10 Mbps)	-
11	H_DUP	Duplex selection: • 0 = Full duplex • 1 = Half duplex	-
10:9	External I/F	External Interface selection: • 0 = RGMII with internal delay • 1 = RGMII without internal delay • 2 = RMII • 3 = G/MII	-
8:6	REF CLK	PLL reference clock. See Table 4-16, *Reference Clock Values*.	0
5	BOOT Master	Boot Master select • 0 = ARMSS is boot master	-
4	MIN	Minimum boot configuration select bit. • 0 = All fields must be independently configured • 1 = A predetermined set of values is configured	-
3:0	Boot Devices	Boot Devices[3:0]: • 1100 = Ethernet	-

4.3.2.9 USB Boot Device Configuration

Figure 4-12 shows the BOOTMODE[15:6] pins assignment to functions when BOOTMODE[3:0] select the USB mode.

Figure 4-12. USB Boot Mode Configuration Fields

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Phy Frequency SEL			REF_SRC		0	PORT	REF CLK			0	MIN	1101			

Table 4-13. USB Mode Boot Configuration Field Description

Pins	Field	Description	MIN Default Value
15:13	Phy Frequency SEL	Frequency selection: • 0 = Reserved • 1 = Reserved • 2 = 12 MHz (with internal clock selects UART PLL) • 3 = 19.2 MHz (with internal clock selects UART PLL) • 4 = Reserved • 5 = 24 MHz (with internal clock selects UART PLL) • 6 = Reserved • 7 = 50 MHz (with internal clock selects NSS PLL)	-
12:11	REF_SRC	Clock selection: • 0 = Reserved • 1 = External clock connected to XO pin • 2 = Internal clock • 3 = Reserved	-
10	D_MOD	DFU support enable: • 0 = Device firmware upgrade (DFU) support enabled (device mode only)	-
9	PORT	Port enable: • 0 = Enable only port 0 • 1 = Enable only port 1	-
8:6	REF CLK	PLL reference clock. See Table 4-16, *Reference Clock Values*.	0
5	BOOT Master	Boot Master select • 0 = ARMSS is boot master	-
4	MIN	Minimum boot configuration select bit. • 0 = All fields must be independently configured • 1 = A predetermined set of values is configured	-
3:0	Boot Devices	Boot Devices[3:0] • 1101 = USB	-

NOTE: When the internal reference clock is selected the choice of NSS versus UART PLL is made based on the desired reference frequency.

4.3.2.10 MMCSD Boot Device Configuration

Figure 4-13 shows the BOOTMODE[15:6] pins assignment to functions when BOOTMODE[3:0] select the MMCSD mode.

Figure 4-13. MMCSD Boot Mode Configuration Fields

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BACKUP	B_ACK	BOOT	Card Type		1 bit	PORT	REF CLK			0	MIN	1110			

Table 4-14. MMCSD Mode Boot Configuration Field Description

Pins	Field	Description	MIN Default Value
15	BACKUP	• 0 = USB boot used as backup after timeout • 1 = UART boot used as backup after timeout	-
14	B_ACK	Boot Acknowledgement: • 0 = No Boot Acknowledgement expected from MMC • 1 = Boot Acknowledgement expected, will timeout if not received	-
13	BOOT	Boot: • 0 = eMMC boot sector is not used for boot • 1 = Boot data is read from eMMC boot sector	-
12:11	Card Type	Card type selection: • 0 = Auto detect • 1 = MMC card • 2 = SD card • 3 = Auto detect	-
10	1 bit	• 0 = Use maximum available data lines • 1 = Do complete boot in 1 bit mode	-
9	PORT	Boot port: • 0 = Boot from port 0 • 1 = Boot from port 1	-
8:6	REF CLK	PLL reference clock. See Table 4-16, *Reference Clock Values*.	0
5	BOOT Master	Boot Master select • 0 = ARMSS is boot master	-
4	MIN	Minimum boot configuration select bit. • 0 = All fields must be independently configured • 1 = A predetermined set of values is configured	-
3:0	Boot Devices	Boot Devices[3:0]: • 1110 = MMCSD	-

NOTE: Only MMC supports the boot sector (and not SD). Setting the BOOT bit implies the card type is MMC.

4.3.2.11 UART Boot Device Configuration

Figure 4-14 shows the BOOTMODE[15:6] pins assignment to functions when BOOTMODE[3:0] select the UART mode.

Figure 4-14. UART Boot Mode Configuration Fields

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	BACKUP		TOUT		PORT		REF CLK			BOOT Master	MIN	1111			

Table 4-15. UART Mode Boot Configuration Field Description

Pins	Field	Description	MIN Default Value
15	RSVD	Reserved. Set to 0.	-
14:13	BACKUP	Boot device after timeout: • 0 = I^2C Boot after timeout • 1 = SPI Boot after timeout • 2 = XIP Boot after timeout • 3 = Reserved	-
12:11	TOUT	Timeout configuration: • 0 = Never timeout • 1 = Timeout after 12 seconds • 2 = Timeout after 30 seconds • 3 = Timeout after 60 seconds	-
10:9	PORT	Port select: • 0 = Boot on UART 0 • 1 = Boot on UART 1 • 2 = Boot on UART 2 • 3 = Boot on UART 0	-
8:6	REF CLK	PLL reference clock. See Table 4-16, *Reference Clock Values*.	0
5	BOOT Master	Boot Master select • 0 = ARMSS is boot master	-
4	MIN	Minimum boot configuration select bit. • 0 = All fields must be independently configured • 1 = A predetermined set of values is configured	-
3:0	Boot Devices	Boot Devices[3:0]: • 1111 = UART	-

4.3.2.12 PLL Configuration

- Possible ARMSS Speeds are 200, 400, 600, 800, 1000 MHz
- Possible DSP Speeds are 400, 600, 800, 1000 MHz

The default speed to start with after cold reset is derived from EFUSE_BOOTROM register.

To select the PLL reference clock, see Table 4-16. This setting must be made according to the actual clock speed provided by system board. That is, the crystal mounted on board, or the clock supplied by an external clock generator.

Table 4-16. Reference Clock Values

REF CLK Boot Pin Value	PLL Reference Clock, MHz	Clock Source
0	19.2	System oscillator or external clock input(1)
1	24	System oscillator or external clock input
2	25	System oscillator or external clock input
3	26	System oscillator or external clock input
4-7	Reserved	Reserved

(1) See Section 5.4, *Clock Management.*

4.3.3 Boot Parameter Tables

The ROM Bootloader (BootROM) uses a set of tables to carry out the boot process. The boot parameter table is the most common format the BootROM employs to determine the boot flow. These boot parameter tables have certain parameters common across all the boot modes, while the rest of the parameters are unique to the boot modes. The common entries in the boot parameter table are shown in Table 4-17.

The boot tables reside at a fixed location in memory which is described in Section 4.4, *Boot RAM Memory Map*.

Table 4-17. Boot Parameter Table Common Parameters

Byte Offset	Name	Default Value	Description
0	Length	-	The length of the boot parameter table, including the length field itself, in bytes.
2	Checksum	0	The 16 bits ones complement of the ones complement of the entire table. A value of 0 will disable checksum verification of the table by the boot ROM.
4	Boot Mode	See Table 4-18.	Internal values used by BootROM for different boot modes. See Table 4-18.
6	Port Num	0	Identifies the peripheral port number to boot from, if applicable. For example, ports of I2C, SPI, UART, and others.
8	PLL 0 CFG, MSW	According to PLL reference clock speed and EFUSE_BOOTROM register	MAIN PLL configuration, MSW. See Figure 4-15.
10	PLL 0 CFG, LSW		MAIN PLL configuration, LSW
12	PLL 1 CFG, MSW		ARM PLL configuration, MSW
14	PLL 1 CFG, LSW		ARM PLL configuration, LSW
16	System Frequency	From EFUSE_BOOTROM register	The frequency of the system clock in MHz
18	Core Frequency	From EFUSE_BOOTROM register	The frequency of the core clock in MHz
20	Boot Master	-	Non-zero if current core is the boot master

Table 4-18 lists the possible boot modes used in the boot parameter tables.

Table 4-18. Boot Mode Values

Boot Mode Value (decimal)	Boot Mode
30	PCIe
40	I^2C Master
41	I^2C Slave
42	I^2C Master Write (Transmit mode)
50	SPI
70	XIP
80	Reserved
90	Ethernet, BOOTP/TFTP protocol.
100	Sleep
110	UART
140	USB
150	MMCSD
160	QSPI

Figure 4-15 and Table 4-19 describe the PLL configuration fields.

Figure 4-15. PLL Configuration Fields

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTL		MULTIPLIER														PRE_DIVIDE								POST_DIVIDE							

Table 4-19. PLL Configuration Field Description

Bit	Field	Description
31-30	CTL	Initialization of PLL: • 0 = Do Not Initialize the PLL • 1 = Initialize the PLL only if it was not already enabled • 2 = Unconditionally initialize the PLL • 3 = Unconditionally disable the PLL
29-16	MULTIPLIER	The PLL multiplier to use. This is the *true multiplier value*, not the multiplier (value -1) used in the register programming of the PLL. Possible multiplier values are [0-16383].
15-8	PRE_DIVIDE	The PLL pre-divider value to use. The *true divider value.* Possible values are [0-255].
7-0	POST_DIVIDE	The PLL post divider value to use. Only values 1 and 2 are supported in this device.

4.3.3.1 Sleep Boot Parameter Table

Table 4-20 shows the boot parameter table for Sleep boot (no boot). Must be preceded with the common boot parameters described in Table 4-17.

Table 4-20. Sleep Boot Parameter Table

Byte Offset	Name	Default Value	Description	Configured Through BOOTMODE Pins?
22	Options	0	Currently unused	NO

4.3.3.2 PCIE Boot Parameter Table

Table 4-21 shows the boot parameter table for PCIe boot. Must be preceded with the common boot parameters described in Table 4-17.

Table 4-21. PCIE Boot Parameter Table

Byte Offset	Name	Default Value	Description	Configured Through BOOTMODE Pins?
22	Options	0	bit 0 Reserved bit 1 NO_INIT • 0 = PCIe subsystem is configured by boot code • 1 = PCIe subsystem is not configured by boot code bits 15-2 Reserved	NO
24	Address WITDH	From pins	Only values 32 or 64 supported	YES in conjunction with BAR sizes
26	Link Rate	2500	Only 2500 MHz supported	NO
28	Ref Clock	From pins		YES
30	Window 1 Size	From pins, see Section 4.3.2.2, *PCIE Boot Device Configuration*	PCIE BAR 1 Size	YES
32	Window 2 Size		PCIE BAR 2 Size	YES
34	Window 3 Size		PCIE BAR 3 Size (valid only with 32-bit address)	YES
36	Window 4 Size		PCIE BAR 4 Size (valid only with 32-bit address)	YES
38	Window 5 Size		PCIE BAR 5 Size (valid only with 32-bit address)	YES
40	Vendor ID	0x104C	PCIe Vendor ID Value	NO

Table 4-21. PCIE Boot Parameter Table (continued)

Byte Offset	Name	Default Value	Description	Configured Through BOOTMODE Pins?
42	Device ID	0xB00A	PCIe Device ID Value	NO
44	Class Code, Rev MSW	0x0480	Class code, class 4	NO
46	Class Code, Rev, LSW	0x0001	Revision Id	NO
48	RSVD	0	Reserved	NA
50	RSVD	1	Reserved	NA
52	RSVD	0	Reserved	NA
54	RSVD	0	Reserved	NA
56	RSVD	0	Reserved	NA
58	RSVD	0	Reserved	NA
60	Timeout	See Section 4.3.2.2, *PCIE Boot Device Configuration*	Boot timeout, seconds	NO

4.3.3.3 I^2C/I^2C Slave/I^2C Master Write Boot Parameter Table

Table 4-22 shows the boot parameter table for I2C boot. Must be preceded with the common boot parameters described in Table 4-17.

Table 4-22. I^2C/I^2C Slave/I^2C Master Write Boot Parameter Table

Byte Offset	Name	Default Value	Description	Configured Through BOOTMODE Pins?
22	Options	3	bits 2-0 BOOT TYPE • 3 = GP Format Read • 4 = Slave Option receive • Others = Reserved	NO
24	Read Addr	See Section 4.3.2	The 16-bit read address (when in master boot)	NO
26	Bus Addr	See Section 4.3.2	The I^2C Bus address (when in master boot)	NO
28	Write Addr	0x11	Transmit write address (when in master transmit mode)	NO
30	Local Addr	0x10 (I^2C Slave), From pins (I^2C)	I^2C Bus address of the device I^2C port	YES (I^2C)
32	I^2C Frequency	20	I^2C Bus frequency driven by the device I^2C port	NO
34	Next Read Addr	0	Read address after completion of a CONFIG table	NO
36	Next Bus Addr	0	Bus address after completion of a CONFIG table	NO
38	Address delay	0x200	CPU cycle delay between address write and data read	NO

The Boot type field refers to how the boot code handles data read from the I^2C interface.

The following mode is supported in ARMSS master boot mode:

- GP format read – the data read from the I^2C interface is treated as GP header formatted data.

In slave receive option mode, the data read from the I^2C interface is treated as a new options field, which replaces the existing field (which must have indicated slave receive options) and the boot is restarted (with presumably boot data following).

4.3.3.4 SPI Boot Parameter Table

Table 4-23 shows the boot parameter table for SPI boot. Must be preceded with the common boot parameters described in Table 4-17.

Table 4-23. SPI Boot Parameter Table

Byte Offset	Name	Default Value	Description	Configured Through BOOTMODE Pins?
22	Options	3	bits 1-0 • 0 = Data read is formatted as a Boot Parameter Table • 1 = Data read is formatted as Boot Table Data • 2 = Data read is formatted as a BOOT CONFIG table • 3 = Data read is formatted as GP Header data bits 15-2 = Reserved	NO
24	Address Width	See Section 4.3.2	The 16-bit read address (master boot)	YES
26	NPins	8	Number of pins driven (clock, data in, data out, chip selects)	YES
28	CSEL	See Section 4.3.2	Bit field specifying chip select – with a 0 bit indicating the active select	YES
30	Mode	See Section 4.3.2	Clock Phase/Polarity setting	YES
32	C2T delay	1	SPI C2T value	NO
34	Bus Frequency (100 kHz)	5	500-kHz bus frequency by default	NO
36	Address, MSW	See Section 4.3.2	Read address, MSW	YES
38	Address, LSW		Read address, LSW	YES
40	Next CSEL	0	CSEL used after BOOT CONFIG is done	NO
42	Next read, MSW	0	Read used after BOOT CONFIG is done	NO
44	Next read, LSW	0	Read used after BOOT CONFIG is done	NO
46	SPI read command	0x03	Command used to read EEPROM	NO
48	Num Dummy bytes	0	Number of dummy bytes sent after read command	NO
50	Initial Commands	0	Not supported	NO

4.3.3.5 QSPI Boot Parameter Table

Table 4-24 shows the boot parameter table for QSPI boot. Must be preceded with the common boot parameters described in Table 4-17.

Table 4-24. QSPI Boot Parameter Table

Byte Offset	Name	Default Value	Description	Configured Through BOOTMODE Pins?
22	Options	See Description	bits 1-0 FORMAT • 0 = Data read is formatted as a Boot Parameter Table • 1 = Data read is formatted as Boot Table Data • 2 = Data read is GP header data (ARMSS default) • 3 = Reserved bits 3-2 DATA IO WIDTH • 0 = Data received in 1 IO pin (default if Addr Width boot pin is 0) • 1 = Data received in 2 IO pins • 2-3 = Data received on 4 IO pins (default if Addr Width boot pin is 1) bits 5-4 ADDRESS IO WIDTH • 0 = Address value sent on 1 IO pin (default if Addr Width boot pin is 0) • 1 = Address value sent on 2 IO pins • 2-3 = Address value sent on 4 IO pins (default if Addr Width boot pin is 1) bits 7-6 INSTRUCTION IO WIDTH • 0 = Instruction sent on 1 IO pin (default if Addr Width boot pin is 0) • 1 = Instruction sent on 2 IO pins • 2-3 = Instruction sent on 4 IO pins (default if Addr Width boot pin is 1) bit 8 MODE_EN • 0 = No mode bytes are sent after the address • 1 = One mode byte is sent after the address value (default) bits 15-9 Reserved	NO
24	Address Width	See Section 4.3.2	16, 24 or 32 bit address width	YES
26	CSEL	See Section 4.3.2	Bitfield specifying chip select – bit 0 indicating the active chip select	YES
28	Mode	See Section 4.3.2	Clock Phase/Polarity setting	YES
30	Bus Freq (100 kHz)	480	48 MHz bus frequency	NO
32	Address, MSW	0	Read address, MSW	NO
34	Address, LSW	0	Read address, LSW	NO
36	QSPI PLL CFG MSW	See Section 4.3.2	QSPI PLL Configuration. See Figure 4-15 and Table 4-19.	NO
38	QSPI PLL CFG LSW			NO
40	Module Freq, (100 kHz)	3840	384 MHz module clock	NO
42	Read Command	0x03 or 0xEB	Read command. Default based on 1 bit vs 4 bit selection	NO
44	Mode Byte value	0	Mode Byte value	NO
46	Dummy bytes	0 (1bit) or 2 (48MHz) or 8 (96MHz)	Number of dummy bytes sent after read command	NO
48	Delay, MSW	0x0808 or 0x1010(1)	Chip select delays, MSW: bits 7-0 CSDADS: Chip Select De-Assert Different Slave bits 15-8 CSDA: Chip Select De-Assert	NO

(1) The default values are chosen to be 1 QSPI bit clock. Based on a 384-MHz module clock this comes to 0x8 for 48 MHz QSPI clock or 0x10 for a 96 MHz QSPI clock.

Table 4-24. QSPI Boot Parameter Table (continued)

Byte Offset	Name	Default Value	Description	Configured Through BOOTMODE Pins?
50	Delay, LSW	0x0808 or 0x1010[1]	Chip select delay, LSW: bits 7-0 CSSOT: Chip Select Start of Transfer delay bits 15-8 CSEOT: Chip Select End of Transfer delay	NO
52	CONFIG Command Array	0	Not supported	NO

4.3.3.6 XIP Boot Parameter Table

Table 4-25 shows the boot parameter table for NOR XIP boot. Must be preceded with the common boot parameters described in Table 4-17.

Table 4-25. XIP Boot Parameter Table

Byte Offset	Name	Default Value	Description	Configured Through BOOTMODE Pins?
22	Options	0	bit 0 REF_CFG • 0 = GPMC CONFIG values in parameter table are not used • 1 = GPMC CONFIG values in parameter table configures the GPMC bits 15-1 Reserved	NO
24	Type	1	GPMC type, must be 1	NO
26	Address, MSW	See Section 4.3.2	XIP memory address, MSW	YES
28	Address, LSW		XIP memory address, LSW	YES
30	Chip Select		Active chip select (0-3)	YES
32	Mem width		Memory width. Values 8 or 16 are valid.	YES
34	Wait enable		Extended wait enable 0 = disabled Other values = enabled	YES
36	Chip Select Size	64	Chip select size, MB. Only 128, 64, 32 and 16 are valid.	NO
38	GPMC Config 1 MSW	0	GPMC_CONFIG1 register, MSW	NO
40	GPMC Config 1 LSW	0	GPMC_CONFIG1 register, LSW	NO
42	GPMC Config 2 MSW	0	GPMC_CONFIG2 register, MSW	NO
44	GPMC Config 2 LSW	0	GPMC_CONFIG2 register, LSW	NO
46	GPMC Config 3 MSW	0	GPMC_CONFIG3 register, MSW	NO
48	GPMC Config 3 LSW	0	GPMC_CONFIG3 register, LSW	NO
50	GPMC Config 4 MSW	0	GPMC_CONFIG4 register, MSW	NO
52	GPMC Config 4 LSW	0	GPMC_CONFIG4 register, LSW	NO
54	GPMC Config 5 MSW	0	GPMC_CONFIG5 register, MSW	NO

Table 4-25. XIP Boot Parameter Table (continued)

Byte Offset	Name	Default Value	Description	Configured Through BOOTMODE Pins?
56	GPMC Config 5 LSW	0	GPMC_CONFIG5 register, LSW	NO
58	GPMC Config 6 MSW	0	GPMC_CONFIG6 register, MSW	NO
60	GPMC Config 6 LSW	0	GPMC_CONFIG6 register, LSW	NO
62	GPMC Config 7 MSW	0	GPMC_CONFIG7 register, MSW	NO
64	GPMC Config 7 LSW	0	GPMC_CONFIG7 register, LSW	NO
66	GPMC Config MSW	0	GPMC_CONFIG register, MSW	NO
68	GPMC Config LSW	0	GPMC_CONFIG register, LSW	NO

BootROM configures the GPMC chip selects with the WAIT pins and WAIT pin polarities as shown in Table 4-26. These settings can be later overriden in the GPMC Config registers.

Table 4-26. Wait Pin Configuration

Chip Select	Wait Pin Mapping	Wait Pin Polarity
CS0	WAIT0	Active-Low
CS1	WAIT1	Active-High
CS2	WAIT0	Active-Low
CS3	WAIT1	Active-High

NOTE: The boot ROM always configures region 0. The GPMC base address and mask fields are setup to match the requested chip select.

4.3.3.7 Ethernet BOOTP/TFTP Boot Parameter Table

Table 4-27 shows the boot parameter table for Ethernet BOOTP/TFTP boot. Must be preceded with the common boot parameters described in Table 4-17.

Table 4-27. BOOTP/TFTP Boot Parameter Table

Byte Offset	Name	Default Value	Description	Configured Through BOOTMODE Pins?
22	Options	See Section 4.3.2	bit 0 NO BOOTP • 0 = BOOTP send to determine IP configuration and TFTP information • 1 = BOOTP is not sent. IP configuration and TFTP information already present. bits 4-1 Reserved bits 6-5 INIT • 0 = Fully initialize Ethernet subsystem • 1 = Initialize only currently un-initialized Ethernet subsystem components • 2 = Reserved • 3 = Do not initialize Ethernet subsystem bits 10-7 INTERFACE • 0 = Autodetect if possible • 1 = RMII • 2 = G/MII • 3 = RGMII • 4-7 = Reserved bits 15-11 Reserved	NO
24	MAC Addr, MSW	From E-FUSE	16 MSBs of the MAC address	NO
26	MAC Addr, middle		16 middle bits of the MAC address	NO
28	MAC Addr, LSW		16 LBSs of the MAC address	NO
30	Multi MAC, MSW	0xFFFF	16 MSBs of the multicast MAC address	NO
32	Multi MAC, middle	0xFFFF	16 middle bits of the multicast MAC address	NO
34	Multi MAC, LSW	0xFFFF	16 LSBs of the multicast MAC address	NO
36	BOOTP timeout, MS	4000	4-sec Initial BOOTP timeout	NO
38	TFTP ack timeout, MS	1000	1-sec TFTP timeout	NO
40	Total timeout, sec	90	90-second timeout	NO
42	BOOTP retry limit	10	Retry BOOTP 10 times	NO
44	TFTP retry limit	10	Retry TFTP 10 times	NO
46	Vendor String	“TCI c66x Bootp Boot”	BOOTP request vendor string field	NO
78	ID String	From E-FUSE	The e-fuse MAC address, not the MAC address from this table	NO
98	Lane Enable Mask	See Section 4.3.2	One bit per lane. • 0 - Lane disabled • 1 - Lane enabled	NO
100-111	Reserved	0	Reserved	NA
112	Eth ref, MSW	See Section 4.3.2	G/MII reference clock frequency, MHz. Only 12500 and 15625 are supported.	NO
114	Eth Ref, LSW			NO
116	NSS PLL, MSW		NSS PLL Configuration. Layout is same for MAIN PLL shown in Figure 4-15	NO
118	NSS PLL, LSW			NO
120	Blob Base, MSW	0x0C00	Address to receive image, MSW	NO
122	Blob Base, LSW	0x0000	Address to receive image, LSW	NO

4.3.3.8 USB Boot Parameter Table

Table 4-28 shows the boot parameter table for USB boot. Must be preceded with the common boot parameters described in Table 4-17.

Table 4-28. USB Boot Parameter Table

Byte Offset	Name	Default Value	Description	Configured Through BOOTMODE Pins?
22	Options	0	Currently unused	NA
24	USB Clk SRC	-	USB clock source 0 = external crystal, 1 = external oscillator, 2 = UART PLL, 3 = NSS PLL	YES
26	USB PLL output freq, MHz	-	PLL output frequency. 384 MHz for UART, 1000 MHz for NSS	NO
28	USB ref clock, 100 kHz	-	Value 9.6-, 10-, 12-, 19.2-, 20-, 24-, and 50-MHz are supported (×10 in this field)	YES
30	USB internal source clock divide	-	When clock source is 2 or 3, this is the additional divide factor (1 = divide by 1, 2 by 2, and so forth)	NO
32	USB PLL MSW	-	UART or NSS PLL (depending on USB Clk SRC) config, MSW	NO
34	USB PLL LSW	-	UART or NSS PLL config, LSW	NO
36	Timeout	See Section 4.3.2	Boot timeout value in ms. 0 disables timeout	NO
38	Mode		1 = DFU mode (in device (peripheral) mode) All other values are reserved	YES
40	ID Vendor	0x451	USB Vendor ID value	NO
42	ID Product	0xBB06 (from JTAGID register, PARTNUMBER field)	USB Product ID value	NO
44	BCD Device	0x0000 (from JTAGID register, VARIANT field)	Binary-coded-decimal device release version	NO
46	String OFFSET Vendor	0	Index into RAM string block containing the vendor string	NO
48	String OFFSET product	18	Index into RAM string block containing the product string	NO
50	Serial number OFFSET	27	Index into RAM string block containing the serial number string	NO

4.3.3.9 MMCSD Boot Parameter Table

Table 4-29 shows the boot parameter table for MMC or SD card boot. Must be preceded with the common boot parameters described in Table 4-17.

Table 4-29. MMCSD Boot Parameter Table

Byte Offset	Name	Default Value	Description	Configured Through BOOTMODE Pins?
22	Options	See Section 4.3.2	bit 0 1-bit • 0 = Detect and use full bus width • 1 = Boot in 1-bit mode bits 2-1 MEDIA • 0 = Auto detect • 1 = MMC • 2 = SD • 3 = Auto detect bit 3 BOOT • 0 = Do not boot from MMC boot sectors • 1 = Boot from MMC boot sectors (MEDIA must be set to 1 = MMC) bit 4 BOOT_ACK • 0 = Do not configure for boot acknowledgement • 1 = Configure for boot acknowledgement, timeout if not received bits 15-5 Reserved	YES
24	MMCSD PLL MSW	See Section 4.3.2	UART PLL setup	YES
26	MMCSD PLL LSW			YES
28	Discovery Bus Freq	400	Initial clock speed, kHz	NO
30	Module Freq	3840	Module Frequency, 100 kHz units	NO
32	File Name	“\tiboot.bin”	Boot filename for SD card in UNICODE. Each entry is 16 bits, with a 16-bit 0x0000 terminator	NO
122	Reserved	X	Reserved	NA

4.3.3.10 UART Boot Parameter Table

Table 4-30 shows the boot parameter table for UART boot. Must be preceded with the common boot parameters described in Table 4-17.

Table 4-30. UART Boot Parameter Table

Byte Offset	Name	Default Value	Description	Configured Through BOOTMODE Pins?
22	Options	0	Currently unused	NA
24	Data Format	0	bit 0 Data Format • 0 = Data Format is BLOB. bits 15-1 Reserved	NO
26	Protocol	0	0 = XMODEM Other values are reserved	NO
28	Ping limit	See Section 4.3.2	Number of pings before failure	NO
30	Max Error	10	Error limit before failure	NO
32	Nack Timeout, sec	3	Timeout value for pings, ACKs/NACKs	NO
34	Char Timeout, msec	2	Milliseconds for inter-character timeout	NO
36	Data bits	8	Only 8 bits is supported	NO

Table 4-30. UART Boot Parameter Table (continued)

Byte Offset	Name	Default Value	Description	Configured Through BOOTMODE Pins?
38	Parity	0	bits 1-0 Parity • 0 = No Parity • 1 = Odd parity • 2 = Even Parity bits 15-2 Reserved	NO
40	Num Stop bits x2	2	2× the number of stop bits used. Values 2, 3, and 4 supported	NO
42	Over sample	16	16× and 13× only supported oversample factors	NO
44	Flow control	0	bit 0 Flow Control • 0 = No Flow Control • 1 = Hardware RTS/CTS flow control bits 15-1 Reserved	NO
46	Data rate, MSW	0x0001	Baud rate, MSW. Default value = 115200	NO
48	Data rate, LSW	0x2C00	Baud rate, LSW	NO
50	Blob Base, MSW	0x0C00	For blob format, base address, MSW	NO
52	Blob Base, LSW	0x0000	Blob Base, LSW	NO
54	UART PLL, MSW	See Section 4.3.2	UART PLL configuration, MSW	YES
56	UART PLL, LSW		UART PLL configuration, LSW	YES
58	UART PLL output	3840	UART PLL output in 100-kHz units	NO

4.3.3.11 DDR_EMIF Configuration Table

The BootROM also provides an option to configure the DDR table before loading the image into the external memory. ROM function is provided for this purpose (Section 4.5, *BootROM Function Calls*). The configuration table for DDR_EMIF is shown in Table 4-31.

Table 4-31. DDR_EMIF Configuration Table

Byte Offset	Name	Default Value	Description	Configured Through BOOTMODE Pins?
0	configselect msw	-	Mask for the configuration registers (below in the table) to be set. 1 = enables the register setting. Each register is represented by one bit	NO
4	configselect slsw	-	Mask for the configuration registers	NO
8	configselect lsw	-	Mask for the configuration registers	NO
12	pllprediv	-	DDR PLL pre divider value (must be the exact value, not value-1)	NO
16	pllMult	-	DDR PLL multiplier value (must be the exact value, not value-1)	NO
20	pllPostDiv	-	DDR PLL post divider value (must be the exact value, not value-1)	NO
24	sdRamConfig	-	SDRAM Config register	NO
28	sdRamConfig2	-	SDRAM Config register	NO
32	sdRamRefreshctl	-	SDRAM Refresh Control Register	NO
36	sdRamTiming1	-	SDRAM Timing 1 Register	NO
40	sdRamTiming2	-	SDRAM Timing 2 Register	NO
44	sdRamTiming3	-	SDRAM Timing 3 Register	NO
48	lpDfrNvmTiming	-	LP DDR2 NVM Timing Register	NO
52	powerMngCtl	-	Power management Control Register	NO

Table 4-31. DDR_EMIF Configuration Table (continued)

Byte Offset	Name	Default Value	Description	Configured Through BOOTMODE Pins?
56	iODFTTestLogic	-	IODFT Test Logic Global Control Register	NO
60	performcountCfg	-	Performance Counter Config Register	NO
64	performCountMstRegSel	-	Performance Counter Master Region Select Register	NO
68	readIdleCtl	-	Read IDLE counter Register	NO
72	sysVbusmIntEnSet	-	System Interrupt Enable Set Register	NO
76	sdRamOutImpdedCalcfg	-	SDRAM Output Impedance Calibration Config Register	NO
80	tempAlertCfg	-	Temperature Alert Configuration Register	NO
84	ddrPhyCtl1	-	DDR PHY Control Register 1	NO
88	ddrPhyCtl2	-	DDR PHY Control Register 1	NO
92	proClassSvceMap	-	Priority to Class of Service mapping Register	NO
96	mstId2ClsSvce1Map	-	Master ID to Class of Service Mapping 1 Register	NO
100	mstId2ClsSvce2Map	-	Master ID to Class of Service Mapping 2 Register	NO
104	eccCtl	-	ECC Control Register	NO
108	eccRange1	-	ECC Address Range1 Register	NO
112	eccRange2	-	ECC Address Range2 Register	NO
116	rdWrtExcThresh	-	Read Write Execution Threshold Register	NO
120 - 376	Chip Config	-	Chip Specific PHY configuration	NO

4.3.4 Boot Configuration Format

The BootROM uses a set of tables to carry out the boot process. The basic types of tables used by the BootROM are:

- Boot parameter table
- Boot image table

4.3.4.1 Boot Parameter Table

The boot parameter table is the most common format the BootROM employs to determine the boot flow. Boot parameter tables have a first few parameters in the table common across all the boot modes while the rest of the table format is dependant on the boot mode selected. See Section 4.3.3, *Boot Parameter Table* for the boot parameter table format for different boot modes. The BootROM copies a default boot parameter table for each boot mode into the reserved L2 section of master boot Core0 and modifies the default values based on the boot configuration selected through the BOOTMODE pins. This table forms the maps for the BootROM to execute the boot process.

4.3.4.2 Boot Image Table

The image to be loaded into the device is converted to a format recognizable by the BootROM.

4.3.4.2.1 GP Header Boot Image Format

The GP (General Purpose) header image format is blocks of data, each preceded with an 8-byte header. The 8-byte header consists of a 4-byte length and a 4-byte base address. A block of data follows each header. The data block must contain the number of bytes specified in the block length field. Each GP header image must contain at least one data block, but may contain more. Additional blocks are appended to the previous blocks. The image ends with a trailing 4-byte length field containing 0. This image format is usually used for boot modes in which the image is read from non-volatile memory. The general format of a GP header image is shown in Table 4-32.

Table 4-32. GP Header Boot Image Format

Header Image Format Parts	Subpart	Size	Description
BLOCK 0 GP HEADER	BLOCK 0 LENGTH	4 BYTES	Number of DATA bytes BLOCK 0
	BLOCK 0 BASE	4 BYTES	Address
BLOCK 0 DATA		= BLOCK 0 LENGTH	DATA
BLOCK 1 GP HEADER	BLOCK 1 LENGTH	4 BYTES	Number of DATA bytes BLOCK 1
	BLOCK 1 BASE	4 BYTES	Address
BLOCK 1 DATA		= BLOCK 1 LENGTH	DATA
...	...	...	...
BLOCK N GP HEADER	BLOCK N LENGTH	4 BYTES	Number of DATA bytes BLOCK N
	BLOCK N BASE	4 BYTES	Address
BLOCK N DATA		= BLOCK N LENGTH	DATA
TERMINATING 0		4 BYTES	Zero Length Indicator

The data in a GP header formatted image is processed as it is read from the boot source. The length field in each header tells the boot data processor how many bytes to expect in that block and the base address tells it where to store the data. Data processing continues until a terminating length of 0 is encountered. After all blocks have been read, the BootROM branches to the last base address specified in the image to begin execution of the image.

4.3.4.2.2 Blob Boot Image Format

The blob image format is simply a binary load image with the entry point being the first byte of the image. This image format is generally used for boot modes where the image is received from a host.

In boot modes in which the host does not have direct access to device memory, the image will be stored at a pre-determined address, typically the base address of MSMC SRAM. In these cases, the image must be linked to execute from that address. After the complete image has been received and stored, the BootROM branches to the image storage address for execution.

In boot modes in which the host has direct access to device memory, the host may store the image anywhere in device memory. After writing the image into device memory, the host must write the load address at the BOOT_MAGIC_ADDRESS location. When the BootROM sees a non-zero value as the BOOT_MAGIC_ADDRESS data, it branches to the address written.

4.3.4.3 Utilities Used to Generate Different Tables

4.3.4.3.1 Utilities for ARMSS Boot

Utilities used to generate different table formats for ARMSS boot are listed below:

- BYTESWAPCCS — byteswaps files in ccs format
- BYTESWAPBIN — creates a byte swapped copy of a binary file
- CATCCS — concatenates ccs format files
- CCS2BIN — converts ccs format to binary
- B2CCS — converts a hex b file into a ccs data file
- CCSADDGPHDR — adds a general-purpose header to a ccs format file
- CCSADDGPTLR — adds the 8-byte General Purpose Trailer
- CCSPAD — pad a ccs data file to a certain length. The length is in number of lines.

4.3.5 Boot Modes

4.3.5.1 Sleep Boot

The sleep boot mode does not result in any image being loaded for execution. In the ARMSS master sleep boot mode, the ARMSS simply performs ARM PLL and MAIN PLL initialization if specified by the BOOTMODE pins and then executes a polling loop of its BOOT_MAGIC_ADDRESS. This mode is typically used in debug and development environments in which code images are loaded using an emulator.

4.3.5.2 I²C Bootloader Operation

4.3.5.2.1 I²C BootROM Initialization Process

In the I²C boot mode, the BootROM configures the I²C peripheral.

I²C can operate either in master mode or in slave mode. In the master mode, the boot master drives the I²C slave device where the image is stored. In the slave mode, the data transfer is driven by an external I²C master device. The master device is usually another SoC device or a FPGA. The boot configuration options in the DEVSTAT register are used to select the master/slave mode.

The user is capable of loading the image at any block, provides the offset of the block, the speed of the I²C bus through the BOOTMODE pins values. A detailed summary of the I²C boot parameter table and the BOOTMODE pins definitions are listed in Section 4.3.3.3, *I2C/I2C Slave/I2C Master Write Boot Parameter Table*, Section 4.3.2.1, *Sleep/I2C Slave Boot Device Configuration*, and Section 4.3.2.3, *I2C Master Boot Device Configuration*.

4.3.5.2.1.1 Block Size

With ARMSS master boot the boot code will read 0x800 bytes before processing the data. The last block must be padded to the next 0x800 byte boundary and it must be padded with zeros.

The boot code does not support byte address to bus address wrapping. So the maximum image size that can be access is 64k bytes. What this means is that if the read address is 0xFF00, the read size is 0x200 and the I²C bus address is 0x50, then 0x100 bytes will be read from 0xFF00 at bus address 0x50, followed by 0x100 bytes from 0x0000 at bus address 0x50. The bus address does not increment when the address rolls over.

4.3.5.2.2 I²C BootROM Loading Process

With ARMSS as boot master, only GP header data format is supported. The boot code will read 0x800 bytes before processing the data. The last block must be padded to the next 0x800 byte boundary and it must be padded with zeros. Also, the data is passed as a byte stream to the GP header processing function. The GP header processing function creates the information header by forming the 32-bit words for size and address of each block in big endian format. This is followed by data that is copied into place as a byte stream.

4.3.5.2.2.1 Loading a Boot Image From EEPROM

In this mode, the I²C peripheral is configured as I²C master.

When ARMSS is the boot master, the BootROM will start reading from the I²C EEPROM at the specified I²C bus address on the specified I²C port. This read will be done beginning at the specified base address offset. Data will be read in 2-Kbyte chunks. The data will be stored at the address specified in the GP header. It will continue reading GP header formatted data from the EEPROM and storing it as directed by the headers until a complete image has been read. When the complete image has been read, the BootROM will branch to the base address of the last GP data block in the image.

4.3.5.2.2.2 Master Transmit

In master transmit mode data read from an I²C EEPROM is resent out the device as a simple write. An initial 6 byte field is transmitted, 4 bytes of block header, 2 bytes of data, which is received by the slave and loaded into the slave options field.

Then every block of data read from the I2C EEPROM is retransmitted back out the I2C bus. There must be at least one slave on the bus to acknowledge the requests or the boot will fail.

4.3.5.2.2.3 Loading Image In Slave Mode

The BootROM also provides the option to set the I²C to slave mode and connect to external master device, which can then send the image to boot from. The default slave address is set to the value specified in the boot configuration in Section 4.3.3.3, *I2C/I2C Slave/I2C Master Write Boot Parameter Table*.

With ARMSS as boot master, the BootROM will initialize the I²C hardware and then start polling the specified I²C port for data. As data is read from the I²C port, it will be stored at the address specified in the GP header. The BootROM will continue polling for and reading data until a complete image has been read. Data will be read and processed in 2-Kbyte chunks. The I²C master should pad the image with zeroes to be a multiple of this size. When the complete image has been read, the BootROM will branch to the base address of the last GP data block in the image.

4.3.5.3 SPI Bootloader Operation

4.3.5.3.1 SPI BootROM Initialization Process

In the SPI boot mode, the BootROM initializes only the SPI peripheral. Similar to the I²C boot mode, in the SPI boot mode, the interrupt subsystem and the EDMA are not enabled. The BootROM also bypasses the PLL and runs the boot master at the reference clock frequency. The image is read from the NOR flash connected to the corresponding SPI port and chip-select. The the starting offset to be read can be provided by the user through the BOOTMODE pins (Section 4.3.2.4, *SPI without PLL Boot Device Configuration*). A detailed summary of the SPI boot parameter table and the boot configuration definitions are listed in Section 4.3.3.4, *SPI Boot Parameter Table*.

4.3.5.3.2 SPI BootROM Loading Process

The SPI boot loading process is similar to the loading process explained in Section 4.3.5.2.2.1, *Loading a Boot Image From EEPROM*. except that the image is read from a NOR flash. The data formats supported are also the same as in Section 4.3.5.2.2, *I2C BootROM Loading Process*. All the other loading processes that are detailed for I²C master mode work for the SPI boot mode, too.

4.3.5.4 QSPI Bootloader Operation

4.3.5.4.1 QSPI BootROM Initialization Process

In the QSPI 48-MHz and QSPI 96-MHz boot modes, the BootROM initializes the QSPI peripheral and also configures the interrupt subsystem and the EDMA, then executes the IDLE instruction. The image is read from the QSPI flash connected to the selected chip-select. The starting offset to be read can be provided by the user through the BOOTMODE pins (Section 4.3.2.6, *QSPI Boot Device Configuration*).

A detailed summary of the QSPI boot parameter table and the boot configuration definitions are listed in Section 4.3.3.5, *QSPI Boot Parameter Table*.

NOTE: BootROM does not support QSPI boot in XIP mode. The whole image is first read, and then, executed.

4.3.5.4.2 QSPI BootROM Loading Process

The QSPI boot loading process is similar to the loading process explained in Section 4.3.5.2.2.1, *Loading a Boot Image From EEPROM* except that the image is read from a QSPI flash. All the other loading processes that are detailed for I^2C master mode work for the QSPI 48 and QSPI 96 boot modes, too.

The quad read QSPI protocol can optionally include mode bits, as seen in Table 4-33.

Table 4-33. QSPI Protocol

IO0-IO3	<command>	<address>	<mode>	<dummy>	<data>

In most QSPI flash devices the number of dummy cycles is configurable in a non-volatile register, and the value must be selected to be larger than a minimum which is based on the operation speed. The BootROM has chosen default values to use based on the QSPI device speed, and flash devices must meet the boot settings to start in quad mode.

Some flash devices use the mode bits. The mode bits are used to allow sending multiple read command without actually sending the command byte again. The boot code does not use this protocol. However, because some devices do use the mode bytes, the boot code enables the mode bits and will send a 0 value. The reason for this is that on devices which do use mode bits the bus could wind up in a floating state if the QSPI controller floats the IO lines when the flash device is expecting the data.

If a flash device does not use mode bits, then the value in the dummy byte count register should be set to 2 + the number of dummy bytes that the BootROM sends. Essentially the boot ROM sends 2 cycles of mode bits but the flash device sees them as 2 dummy cycles. The BootROM also supports independent selection of the number of IO lines driven for command, address, and data portions of the read cycle.

4.3.5.5 PCI Express (PCIE) Bootloader Operation

4.3.5.5.1 PCIE BootROM Initialization Process

In the PCIE boot mode, the BootROM configures the BAR registers, the number of windows, and their sizes to provide memory access to the host. The different BAR configurations that are initialized are shown in Table 4-4. The BootROM also configures the SerDes from information it obtains from the boot parameter table, see Section 4.3.3.2, *PCIE Boot Parameter Table*. Furthermore, the BootROM configures the interrupt subsystem by configuring the chip-level interrupt controller. Then the BootROM executes the IDLE instruction (MSI or legacy interrupts can be used). For PCIe interrupts, refer to Section 11.14, *PCIe Subsystem*.

If the timeout value is non-zero then a TIMER module is setup with the specified timeout, and the boot will fail if the boot is not completed before the timer expires.

4.3.5.5.2 PCIE BootROM Loading Process

Once the host initiates the link with the boot device, it should load all sections of the boot image. The host is also responsible for updating the BOOT_MAGIC_ADDRESS of ARMSS Core0.

4.3.5.5.3 PCIE BootROM Hand-Over Process

In this boot mode, ARMSS Core0 is executing an IDLE until the PCIE interrupt wakes it. The interrupt can be an MSI interrupt or a legacy interrupt. Once Core0 comes out of IDLE, the BootROM reads the BOOT_MAGIC_ADDRESS and modifies the program counter to start executing the boot image.

4.3.5.6 GPMC Bootloader Operation

4.3.5.6.1 NOR Bootloader Operation

4.3.5.6.1.1 NOR BootROM Initialization Process

In this mode, the BootROM performs minimum operations. The BootROM just configures the GPMC interface based on the configuration parameters specified in the boot parameter table for the XIP boot mode, see Section 4.3.3.6, *XIP Boot Parameter Table*. The boot parameter structure definition for the XIP boot mode and the parts of this table that can be configured by the BOOTMODE pins are detailed in Section 4.3.2.7, *XIP Boot Device Configuration*.

4.3.5.6.1.2 NOR BootROM Loading Process

During this process, the BootROM just sets the program counter to the address and XIP Chip-Select that are specified in the XIP Boot Parameter Table. No return is expected. The BootROM also does not reserve any memory in L2. The image remains stored in the flash and so no image transfer occurs during this boot mode.

4.3.5.7 Ethernet Bootloader Operation

4.3.5.7.1 Ethernet BootROM Initialization Process

When the device is set to boot through the Ethernet mode, the BootROM configures PHY, NAVSS, and the EMAC. These initial configurations, as well as the interface mode (RMII, G/MII, RGMII), are determined by querying the BOOTMODE pins in the DEVSTAT register, see Section 4.3.2.8, *Ethernet Boot Device Configuration*, and the boot parameter table for the Ethernet boot, see Section 4.3.3.7, *Ethernet Boot Parameter Table*.

The queue manager is setup to route the packets to queues polled by the BootROM.

4.3.5.7.2 Ethernet BootROM Loading Process

With ARMSS as the boot master, the image must be in blob format. After device configuration, the bootloader performs a standard BOOTP/TFTP boot. The device sends a BOOTP request with its MAC address to a host TFTP server to be assigned an IP from a pool of addresses. After this connection is established, the device is able to receive image data encapsulated in Ethernet packets, see Section 4.3.5.7.2.1. Data received is stripped of its network headers and the boot data is stored in the MSMC. After transfer is complete, the BootROM will begin executing the image at the base of MSMC.

4.3.5.7.2.1 Ethernet-Ready Announcement Format

Ethernet packets are accepted only in DIX (Ethernet II) frames with IPv4 and UDP headers. Frames not matching this criteria are discarded and subsequent frames are processed.

The DIX frame contains:

- Destination MAC address = H-MAC addr (from Section 4.3.3.7, normally FF:FF:FF:FF:FF:FF)
- Source MAC address = From Section 4.3.3.7, *Ethernet Boot Parameter Table*, default comes from e-fuse
- Type = IPv4 (0x800)

The IPv4 header contains:

- Version = 4
- Header length = 0
- TOS = 0
- Len = Computed during operation
- ID = 0x001
- Flags + Fragment offset = 0

- TTL = 0x10
- Protocol = UDP (17)
- Header checksum = Computed during operation
- SRC IP = 0.0.0.0
- DEST IP = 255.255.255.255

The UDP header contains:

- Source port = BOOTP client (68 dec)
- Destination port = BOOTP server (67 dec)
- Length = Computed during operation
- Checksum = Computed during operation

The BOOTP payload contains:

- Opcode = Request (1)
- HW Type = Ethernet (1)
- HW Address Length = 6
- Hop Count = 0
- Transaction ID = 1
- Number of seconds = 0
- Client IP = 0.0.0.0
- Your IP = 0.0.0.0
- Server IP = 0.0.0.0
- Gateway IP = 0.0.0.0
- Client HW Address = Device MAC address, read from e-fuse
- Server Hostname = NULL
- Filename = NULL
- Option 60, vendor ID string, from boot parameter table
- Option 61, client ID string, from boot parameter table.

TFTP is configured as dictated by the. The received data image is placed into the memory location specified by the same table. This value can be re-assigned in a multi-stage boot to allow direct loading to DDR_EMIF. A maximum data size of 0x1C_0000 bytes is supported.

4.3.5.7.3 Ethernet BootROM Hand Over Process

When boot is complete the NSS firmware is disabled, the CDMA is disabled and the queue manager is cleaned. The PHY is left enabled.

4.3.5.8 USB Bootloader Operation

The USB boot mode is used to read the boot image from external USB host.The USB subsystem is driven from the NSS/IEP PLL configured to 1000 MHz.

The image read in this boot mode must be in GP header format.

The BOOTMODE pins provide a way to specify:

- PHY frequency selection
- Reference clock source
- D_MODE – The boot code implements the USB DFU protocol
- USB port number

See Section 4.3.2.9, *USB Boot Device Configuration* and Section 4.3.3.8, *USB Boot Parameter Table*.

More information about USB DFU protocol can be found at http://www.usb.org/developers/docs/devclass_docs/DFU_1.1.pdf.

After entering the enumeration phase, the BootROM will continue reading GP header formatted data from the host and storing it as directed by the headers until a complete image has been read. When the complete image has been read, the BootROM will branch to the base address of the last GP data block in the image.

4.3.5.9 MMCSD Bootloader Operation

Only single data rate with backward compatible interface timing is supported.

The BOOTMODE pins provide a way to specify:

- Card type
- Boot acknowledge
- How to complete the boot – using maximum available data lines, or in 1-bit mode
- MMCSD port number.

See Section 4.3.2.10, *MMCSD Boot Device Configuration* and Section 4.3.3.9, *MMCSD Boot Parameter Table*

If the card type is set to autodetect, then an initial discovery phase is performed:

1. Send CMD 0. (GO IDLE, both for MMC and SD)
2. Send CMD 55. If the card responds then the card type is assumed to be SD and the boot attempt fails (the device do not support SD boot). On timeout the code sends CMD 1. If there is a response then the card type is MMC.

The BootROM will start reading from the MMCSD memory boot sector if specified by the BOOTMODE pins. The data will be stored at the address specified in the GP header. It will continue reading GP Header formatted data from the memory and storing it as directed by the headers until a complete image has been read. When the complete image has been read, the BootROM will branch to the base address of the last GP data block in the image.

4.3.5.10 UART Bootloader Operation

4.3.5.10.1 BootROM Initialization Process

In the UART boot mode, the selected UART module (port) is the only peripheral configured. The baud rate, data, parity, and stop bits are configured based on the information in the UART boot parameter table. The boot parameter table definitions and the boot configuration values that can be set are in Section 4.3.2.11 and Section 4.3.3.10.

Once the BootROM configures the UART, it sends the UART pings for few seconds, which can be seen in the host. The pings consist of an ASCII capital *C* character. The UART boot mode supports only the CRC mode of XMODEM and does not support CHECKSUM mode.

4.3.5.10.2 UART BootROM Loading Process

The boot image to be loadedmust be in blob format. Before the ping from the device stops, load the boot table from the host using the XMODEM protocol.

4.3.5.10.3 UART BootROM Hand-Over Process

The bootloader stores data at the base of MSMC. Once the complete boot image is transferred and stored in the MSMC, the bootloader will begin execution from the base of MSMC.

4.4 Boot RAM Memory Maps

Table 4-34 shows addresses assigned for boot by the ARMSS Core0.

Table 4-34. ARMSS Boot RAM Memory Map

Start Address	Size	Description
0x0C00 0000	0xE 0000	Free (Blob base)
0x0C0E 0000	0x8000	FAT cluster buffer
0x0C0E 8000	0x7E80	Ethernet Packet Memory
0x0C0E FE80	0x80	PCIe Config Block
0x0C0E FF00	0x100	Reserved
0x0C0F 0000	0x4	ARMSS BOOT_MAGIC_ADDRESS
0x0C0F 0010	0x4	Magic Echo Address
0x0C0F 0020	0xDE0	Context
0x0C0F 0E00	0x200	DDR_EMIF Config
0x0C0F 1000	0x2C00	Boot Data
0x0C0F 3C00	0x180	ARMSS BootROM Version string
0x0C0F 3D80	0x80	Boot Status Stack
0x0C0F 3E00	0x100	Boot Stats
0x0C0F 3F00	0x100	Boot Log
0x0C0F 4000	0x100	RAM call Table
0x0C0F 4100	0x100	Boot Parameter Tables
0x0C0F 4200	0x1DE0	Boot ROM data
0x0C0F 5FE0	0x1010	Boot Trace Data
0x0C0F 6FF0	0x10	Free
0x0C0F 7000	0xC00	Supervisor Stack
0x0C0F 7C00	0x400	Undefined Mode Stack

4.5 BootROM Function Calls

Table 4-35 shows BootROM functions available when ARMSS is the boot master.

Table 4-35. Calls to ARMSS BootROM Functions

Memory Address	Function Name	Description
0x00001000	_romtMonitorFunction(void (*fcn)(), …)	This function switches to monitor mode and branches to the passed function.
0x00001004	_romtBootReentry()	BootROM re-entry function. Typically used in a two-stage boot.
0x00001008	SINT16 _romtEnableModule(UINT32 modnum)	Power up a power domain. See Section 5.2.2, *Power Sleep Controller (PSC)*.
0x0000100C	SINT16 _romtDisableModule(UINT32 modnum)	Power down a power domain. See Section 5.2.2, *Power Sleep Controller (PSC)*.
0x00001018	SINT16 _romtConfigPll(UINT32 pllNum, UINT32 prediv, UINT32 mult, UINT32 postdiv)	Configure PLL. See Section 5.4.5.3, *PLL Controller*. pllNum: 0 = MAIN PLL, 1 = NSS PLL, 2 = DDR PLL, 4 = ARM PLL, 5 = UART PLL
0x0000101C	_romtDelay(UINT32 ncycles)	A simple loop delay
0x00001020	UINT16 _romtDeviceFreqMhz()	Return device frequency specificed by device variant (e-fuse), in units of MHz
0x00001024	UINT32 _romtArmNum()	Return the ARMSS number. Must be run from the supervisor mode.
0x00001028	SINT32 _romtTetrisPsc(UINT32 core, UINT32 state)	Transition the ARMSS PSC. Must be run from the supervisor mode. 0 = STATE_ON 1 = STATE_ON_WIR 2 = STATE_OFF_DYNUP 3 = STATE_OFF
0x0000102C	_romtCacheClean()	Perform a cache clean
0x00001030	SINT16 _romtPscSetState(UINT32 modnum, UINT32 state)	Set the PSC next state of a module domain. See Section 5.2.2, *Power Sleep Controller (PSC)*.
0x00001034	_romtMainEmif4Cfg()	DDR_EMIF configuration based on ROM configuration table – Section 4.3.3.11
0x00001054	BOOL _romtPcieBlockProcessWrap(pcieBlockActivate_t * pba)	Configure the specified PCIe port as an endpoint
0x00001058	_romtCacheCleanByAddr(UINT32 addr, UINT32 sizeBytes)	Clean cache by address
0x0000105C	_romtInvalidateCache(UINT32 addr, UINT32 sizeBytes)	Invalidate cache by address

4.5.1 pcieBlockActivate_t Structure

Below is the listing of the pcieBlockActivate_t structure used in romtPcieBlockProcessWrap function.

```
typedef struct pcieConfig_s  {
    UINT32 port;
    UINT32 serdesCfgReg;
    UINT32 serdesLaneCfg[2];
    UINT32 addrWidth;              /* only 32 and 64 are valid */
    UINT32 windowSize[5];          /* Window sizes in Mb. If 64 bit address are used only 1st two
apply */
    UINT32 classCodeRevId;         /* Class code rev ID */
    UINT32 devId_VendorId;         /* Device ID (MSW), vendor ID (LSW) */
    BOOL   lockDetected;           /* Returns TRUE if a lock was detected */
    BOOL   linkUpDetected;         /* Returns TRUE if a link up was detected */
```

```
} pcieConfig_t;

#define PCIE_BLOCK_MAGIC              0xcece1110

typedef struct pcieBlock_s
{

    UINT32       pcieBlockMagic;
    UINT32       rsvd0[3];

    pcieConfig_t pcieConfig;
    UINT32       rsvd1[2];

    sbCfg_t      sbCfg;     /* First entry - multiple entries are supported */


} pcieBlock_t;


typedef struct pcieBlockActivate_s
{
    pcieBlock_t  *blockp;
    UINT32        echoInit;
    UINT32        echoFinal;
    UINT32        resultMap;

} pcieBlockActivate_t;
```

Device Configuration

Topic **Page**

5.1 Control Module (BOOT_CFG)

The following sections describe the BOOT_CFG module.

5.1.1 BOOT_CFG Overview

The BOOT_CFG is a module on the device that controls top-level device behavior in various states. It contains registers for configuration, reset modes, bootstrap signals, and IO pad multiplexing controls. In functional level, the BOOT_CFG module also supports top level scratchpad and inter-processor communication registers, as well as generating interrupts to other processor cores.

The BOOT_CFG module has registers associated with:

- Device pad configuration
- Kick protection (register write protection)
- BOOT_CFG module interrupts
- TeraNet priority
- Inter-processor communication
- Some device external signals
- Event mux control
- Reset mux control
- Timer Pin Manager
- Scratchpad
- MLB IOs Control
- Device reset status
- DSP boot address
- eCAP/ePWM/eQEP control and status
- NSS MAC address
- ARMSS endian configuration
- ARMSS and DEBUG_SS trace buffers
- PLL control
- Clock muxing, enabling and division
- Control for some device LDOs
- USB PHY control
- Modules IDs
- Device power state
- DSP NMI event generation
- Oscilator control
- DCAN RAM Initialization
- Ethernet Configuration

5.1.2 BOOT_CFG Integration

This section describes the module integration in the device including information about clocks, resets, and hardware requests.

Figure 5-1 shows the integration of the BOOT_CFG module in the device.

Figure 5-1. BOOT_CFG Integration

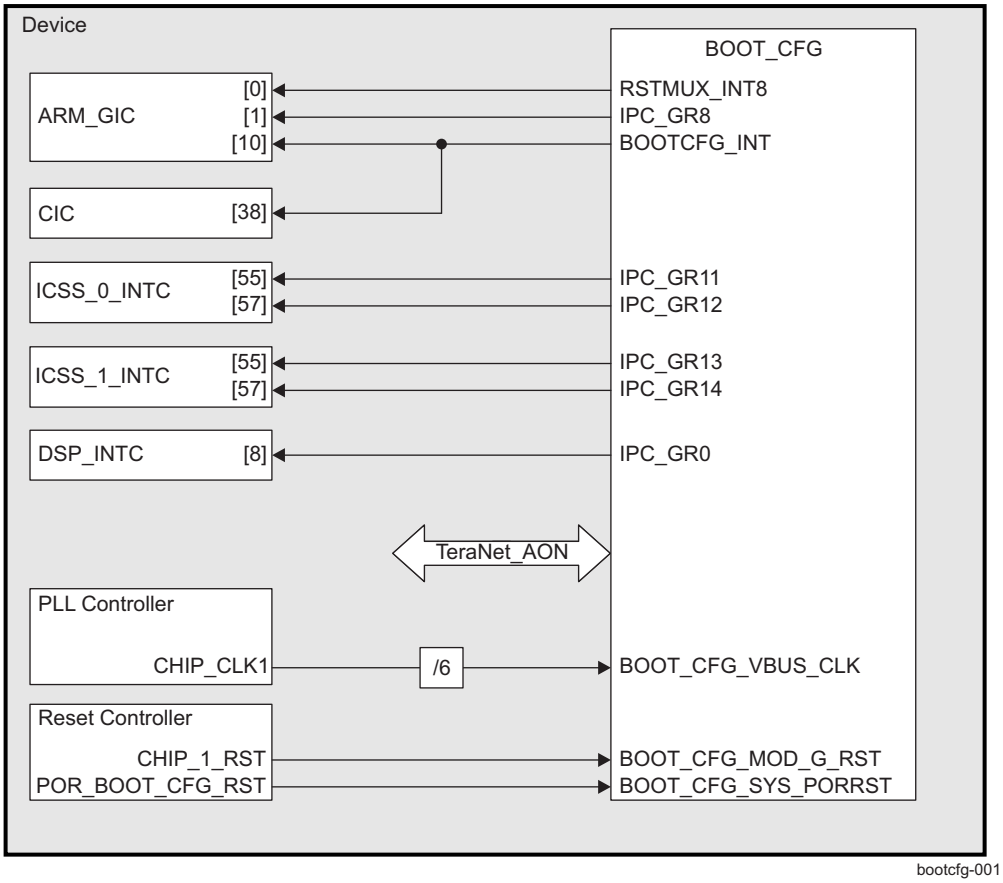


Table 5-1 through Table 5-3 summarize the integration of the BOOT_CFG module in the device.

Table 5-1. BOOT_CFG Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
BOOT_CFG	PD0	LPSC0	TeraNet_AON

Table 5-2. BOOT_CFG Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
BOOT_CFG	BOOT_CFG_VBUS_CLK	CHIP_CLK1/6	PLL Controller	Functional and interface clock for the BOOT_CFG module with frequency equal to CHIP_CLK1 divided by 6.
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
BOOT_CFG	BOOT_CFG_SYS_PORRST	POR_BOOT_CFG_RST	Reset Controller	Power On Reset
	BOOT_CFG_MOD_G_RST	CHIP_1_RST	Reset Controller	BOOT_CFG module reset

Table 5-3. BOOT_CFG Hardware Requests

Interrupt Requests							
Module Instance	**Event Name**	**Mapped To Input Event [Number]**					**Description**
		ARM GIC	**CIC**	**ICSS0 INTC**	**ICSS1 INTC**	**DSP INTC**	
BOOT_CFG	BOOTCFG_INT	[10]	[38]	-	-	-	Interrupt indicating addressing violation or protection violation error.
	IPC_GR0	-	-	-	-	[8]	Interrupt generated by writing of 1h to BOOTCFG_IPCGR0[0].
	IPC_GR8	[1]	-	-	-	-	Interrupt generated by writing of 1h to BOOTCFG_IPCGR8[0].
	IPC_GR11	-	-	[55]	-	-	Interrupt generated by writing of 1h to BOOTCFG_IPCGR11[0].
	IPC_GR12	-	-	[57]	-	-	Interrupt generated by writing of 1h to BOOTCFG_IPCGR12[0].
	IPC_GR13	-	-	-	[55]	-	Interrupt generated by writing of 1h to BOOTCFG_IPCGR13[0].
	IPC_GR14	-	-	-	[57]	-	Interrupt generated by writing of 1h to BOOTCFG_IPCGR14[0].
	RSTMUX_INT8	[0]	-	-	-	-	Interrupt generated by TIMER_5 when used as WD Timer for ARMSS.

NOTE: For more information about BOOTCFG_INT, see Section 5.1.3.1.3.

For more information about IPC_GR**x**, see Section 5.1.3.1.5.

For more information about RSTMUX_INT8, see Section 5.1.3.1.8.

5.1.3 BOOT_CFG Functional Description

5.1.3.1 Description for BOOT_CFG Register Types

The following sub-sections provide summarization and description for most of the registers which are part of the BOOT_CFG module memory space. The rest of the registers are described in the corresponding chapters where the functionality associated with particular BOOT_CFG register is covered more in detail.

5.1.3.1.1 Pad Configuration Registers

The pad configuration registers are used to configure most of the device pads. Each pad configuration register (PADCONFIGx) is assotiated only with one pad and has bits as described in Table 5-4.

Table 5-4. Description Of The Pad Configuration Register Bits

Bits	Field Name	Description	Type
31-21	RESERVED		R
20-19	BUFFERCLASS	Buffer class selection	R/W
		For 3.3V I/Os:	
		0h and 1h: 50B — 50 ohm nominal impedance for the buffer, class B driver 100MHz in 50 ohm mode	
		2h and 3h: 40D — 40 ohm nominal impedance for the buffer, class D driver 200MHz in 40 ohm mode	
		For 1.8V I/Os:	
		0h: 50B — 50 ohm nominal impedance for the buffer, class B driver 100MHz in 50 ohm mode	
		1h: 40C — 40 ohm nominal impedance for the buffer, class C driver 150MHz in 40 ohm mode	
		2h: 40D — 40 ohm nominal impedance for the buffer, class D driver 200MHz in 40 ohm mode	
		3h: 40E — 40 ohm nominal impedance for the buffer, class E driver 250MHz in 40 ohm mode	
18	RESERVED	This bit must not be modified.	R/W
17	PULLTYPESEL	Selects pull-up or pull-down type resistor for a given pad	R/W
		0h: Pull-down is selected	
		1h: Pull-up is selected	
16	PULLUDEN	Enables pull-up or pull-down resistor for a given pad	R/W
		0h: Enables weak pull-up/down	
		1h: Disables weak pull-up/down	
15-4	RESERVED		R
3-0	MUXMODE	Selects the desired multiplexing mode for a given pad	R/W
		0h: Primary function	
		1h: Secondary function	
		2h: Tertiary function	
		3h: Quaternary function	
		4h: Quinary function	
		5h: Senary function	

Many of the device pads support pad multiplexing. This means that their function can be independently chosen from two or more options. The selection of functions available on each pad is enumerated in table "*Pin Multiplexing*" of the device Data Manual. The desired function is selected via the MUXMODE field of the associated pad configuration register.

The BUFFERCLASS field is used to select the impedance and maximum switching frequency of an I/O. That choice is mainly driven by the speed of the corresponding interface the I/O is associated with. Choosing the correct buffer class is determined by identifying the switching frequency of an interface and choosing the class that is closest to and greater than that frequency. It has to be taken into account that some of the device interfaces do not have buffer class selection bits. Only interfaces available on I/Os having pad configuration registers have the buffer class selection available. For example, USB and EMIF do not have such bits as they use PHYs instead of I/Os with pad configuration registers. Table "*Multiplexing Characteristics*" of the device Data Manual can be used to check if certain signal have or not a PADCONFIG register associated with it.

Table 5-5 shows the reset values of all pad configuration registers (BOOTCFG_PADCONFIG0 to BOOTCFG_PADCONFIG259). If a register index is not presnt in that table this means the corresponding register is not used and should be considered as Reserved. Write accesses should not be performed to these locations.

Table 5-5. BOOTCFG_PADCONFIG0 to BOOTCFG_PADCONFIG259 Reset Values

Register Index	Reset Value	Register Index	Reset Value	Register Index	Reset Value	Register Index	Reset Value	Register Index	Reset Value	Register Index	Reset Value	Register Index	Reset Value
0	00140003h	40	00150003h	80	00140003h	120	00060003h	160	00140003h	200	00140003h	243	00050000h
1	00140003h	41	00150003h	81	00140003h	121	00060003h	161	00140003h	201	00140003h	244	00050000h
2	00140003h	42	00150003h	82	00140003h	122	00060003h	162	00140003h	202	00140003h	258	00040000h
3	00140003h	43	00150003h	83	00140003h	123	00060003h	163	00140003h	203	00140003h	259	00040000h
4	00140003h	44	00150003h	84	00140003h	124	00060003h	164	00140003h	204	00140003h		
5	00140003h	45	00150003h	85	00140003h	125	00060003h	165	00140003h	205	00140003h		
6	00140003h	46	00150003h	86	00140003h	126	00060003h	166	00140003h	206	00140003h		
7	00140003h	47	00150003h	87	00140003h	127	00060003h	167	00140003h	207	00140003h		
8	00140003h	48	00140003h	88	00140003h	128	00060003h	168	00140003h	208	00140003h		
9	00140003h	49	00140003h	89	00140003h	129	00140003h	169	00140003h	209	00140003h		
10	00140003h	50	00140003h	90	00140003h	130	00140003h	170	00140003h	210	00140003h		
11	00140003h	51	00150003h	91	00140003h	131	00140003h	171	00140003h	211	00140003h		
12	00140003h	52	00140003h	92	00140003h	132	00140003h	172	00140003h	212	00140003h		
13	00140003h	53	00140003h	93	00140003h	133	00140003h	173	00140003h	213	00140003h		
14	00140003h	54	00140003h	94	00140003h	134	00140003h	174	00140003h	214	00140003h		
15	00140003h	55	00140003h	95	00140003h	135	00160003h	175	00140003h	215	00140003h		
16	00140003h	56	00140003h	96	00140003h	136	00160003h	176	00140003h	216	00140003h		
17	00160003h	57	00140003h	97	00140000h	137	00160003h	177	00140003h	217	00140003h		
18	00160003h	58	00140003h	98	00060003h	138	00160003h	178	00140003h	218	00140003h		
19	00160003h	59	00160003h	99	00040003h	139	00140003h	179	00160003h	219	00140003h		
20	00160003h	60	00160003h	100	00060000h	140	00140003h	180	00140003h	220	00140003h		
21	00160003h	61	00160003h	101	00060000h	141	00140003h	181	00140003h	221	00160003h		
22	00160003h	62	00160003h	102	00040000h	142	00140003h	182	00140003h	222	00140003h		
23	00160003h	63	00160003h	103	00040000h	143	00140003h	183	00140003h	223	00060000h		
24	00160003h	64	00160003h	104	00040000h	144	00140003h	184	00140003h	224	00060000h		
25	00160003h	65	00160003h	105	00060000h	145	00140003h	185	00140003h	225	00060000h		
26	00160003h	66	00160003h	106	00060000h	146	00140003h	186	00140003h	226	00060000h		
27	00160003h	67	00160003h	107	00040000h	147	00140003h	187	00140003h	227	00060000h		

Table 5-5. BOOTCFG_PADCONFIG0 to BOOTCFG_PADCONFIG259 Reset Values (continued)

Register Index	Reset Value	Register Index	Reset Value	Register Index	Reset Value	Register Index	Reset Value	Register Index	Reset Value	Register Index	Reset Value	Register Index	Reset Value
28	00160003h	68	00160003h	108	00040000h	148	00140003h	188	00140003h	228	00060000h		
29	00160003h	69	00140003h	109	00040000h	149	00140003h	189	00140003h	229	00060000h		
30	00150003h	70	00140003h	110	00060003h	150	00140003h	190	00140003h	230	00060000h		
31	00150003h	71	00140003h	111	00060003h	151	00140003h	191	00140003h	231	00060000h		
32	00150003h	72	00140003h	112	00040003h	152	00140003h	192	00140003h	235	00050000h		
33	00150003h	73	00140003h	113	00040003h	153	00140003h	193	00140003h	236	00040000h		
34	00150003h	74	00140003h	114	00040003h	154	00140003h	194	00140003h	237	00040000h		
35	00150003h	75	00140003h	115	00060000h	155	00140003h	195	00140003h	238	00040000h		
36	00150003h	76	00140003h	116	00060000h	156	00140003h	196	00140003h	239	00050000h		
37	00150003h	77	00140003h	117	00060000h	157	00140003h	197	00140003h	240	00050000h		
38	00150003h	78	00140003h	118	00060000h	158	00140003h	198	00140003h	241	00050000h		
39	00150003h	79	00140003h	119	00060003h	159	00140003h	199	00140003h	242	00050000h		

5.1.3.1.2 Kick Protection Registers

The BOOT_CFG module supports a kicker mechanism which prevents spurious writes from changing the values of any of the BOOT_CFG registers. When the kicker is locked (which is its initial state after device reset) none of the BOOT_CFG registers are writable. They can only be read. A write is required to the BOOTCFG_KICK0 and BOOTCFG_KICK1 registers with exact data values before the kicker lock mechanism is un-locked. Once released then all the BOOT_CFG registers having write permissions are writable. The read only registers are still read only. First BOOTCFG_KICK0 has to be written with 83E7 0B13h. Then BOOTCFG_KICK1 has to be written with 95A4 F1E0h. After these two steps a write access to the BOOT_CFG registers is allowed. Writing any other data value to either of these two registers locks the kicker mechanism and blocks any writes to the BOOT_CFG registers.

NOTE: In order to ensure that all BOOT_CFG registers are write protected, software must always re-lock the kicker mechanism after completing the register writes.

In order to reduce the access latency some of the latency critical BOOT_CFG registers are non kick-protected. All other registers inside the BOOT_CFG module except the *Inter-processor Communication Registers* (for details, see Section 5.1.3.1.5) are kick-protected.

5.1.3.1.3 BOOT_CFG Module Interrupts

The BOOT_CFG module has one interrupt request, the BOOTCFG_INT, which is associated with the following registers:

- BOOTCFG_INTR_RAW_STAT_SET — interrupt raw status register
- BOOTCFG_INTR_ENABLED_STAT_CLR — interrupt status register
- BOOTCFG_INTR_ENABLE — interrupt enable register
- BOOTCFG_INTR_ENABLE_CLR — interrupt disable register

For the interrupt behavior of the BOOT_CFG module the following applies:

- The BOOT_CFG module asserts its interrupt line only if the interrupts are enabled by setting to 1h the corresponding bits in the BOOTCFG_INTR_ENABLE register. These interrupts can be disabled by setting to 1h the corresponding bits in the BOOTCFG_INTR_ENABLE_CLR register.
- After an interrupt has been serviced, software must clear the corresponding status flag. This is done by setting to 1h the corresponding bit in the BOOTCFG_INTR_ENABLED_STAT_CLR register which also

clears the corresponding bit in the BOOTCFG_INTR_RAW_STAT_SET register. The status flags in the BOOTCFG_INTR_RAW_STAT_SET register are set even if the corresponding interrupt is disabled unlike those in the BOOTCFG_INTR_ENABLED_STAT_CLR register, which are set only if the corresponding interrupt is enabled.

- An interrupt is also generated by the BOOT_CFG, if certain bit in the BOOTCFG_INTR_RAW_STAT_SET register is set to 1h and the corresponding interrupt is enabled through the BOOTCFG_INTR_ENABLE register. This feature is useful during user software debugging. In addition, even if interrupts are not enabled the corresponding raw flag in the BOOTCFG_INTR_RAW_STAT_SET register is set to 1h when an IRQ condition occurs.
- Even if interrupts are not enabled, a status bit in the BOOTCFG_INTR_RAW_STAT_SET register can also be cleared by setting to 1h the corresponding bit in the BOOTCFG_INTR_ENABLED_STAT_CLR register.

Table 5-6 lists the interrupt events which can assert the BOOTCFG_INT interrupt line.

Table 5-6. BOOT_CFG Events

Event Flag	Event Mask	Description
BOOTCFG_INTR_RAW_STAT_SET[1] ADDR_ERR BOOTCFG_INTR_ENABLED_STAT_CLR[1] ENABLED_ADDR_ERR	BOOTCFG_INTR_ENABLE[1] ADDR_ERR_EN BOOTCFG_INTR_ENABLE_CLR[1] ADDR_ERR_EN_CLR	Addressing violation interrupt. Occurs when accessing an illegal address inside the BOOT_CFG module.
BOOTCFG_INTR_RAW_STAT_SET[0] PROT_ERR BOOTCFG_INTR_ENABLED_STAT_CLR[0] ENABLED_PROT_ERR	BOOTCFG_INTR_ENABLE[0] PROT_ERR_EN BOOTCFG_INTR_ENABLE_CLR[0] PROT_ERR_EN_CLR	Protection violation interrupt. Occurs when a protected register is accessed without the required secure/privilege level permissions.

When addressing violation or protection violation occurs, the error associated details are captured in the BOOTCFG_FAULT_ADDR and BOOTCFG_FAULT_STAT registers. BOOTCFG_FAULT_ADDR contains the address of the first fault access and BOOTCFG_FAULT_STAT contains status attributes associated with the first fault access. To clear the contents of these two registers the BOOTCFG_FAULT_CLR[0] FAULT_CLR bit must be set to 1h.

5.1.3.1.4 TeraNet Priority Registers

The TeraNet priority registers are used for controlling the transaction priority of certain masters on the TeraNet. Each 3-bit field in these registers is associated only with one master. Setting this bit field to 0h means that the transaction performed by this master has highest priority over the other transactions. A value of 7h is for lowest priority. The TeraNet priority registers are:

- BOOTCFG_INITIATOR_PRIORITY0
- BOOTCFG_INITIATOR_PRIORITY1

5.1.3.1.5 Inter-processor Communication Registers

The inter-processor communication registers are used for generating interrupts to the device cores. Any master having access to the BOOT_CFG registers can generate an interrupt by writing 1h to the IPCGR**x**[0] bits. For example, writing 1h to BOOTCFG_IPCGR0[0] generates and interrupt to the DSP. These registers also provide a *Source ID* facility through the IPCGR**x**[31-4] SRC and IPCAR**x**[31-4] SRC_CLR bit fields by which up to 28 different sources of interrupts can be identified. Allocation of the source bits to source processor and meaning is entirely based on software convention. Virtually, anything can be a source for these fields as this is completely controlled by software. Writing 1h to an SRC bit sets to 1h both the SRC and corresponding SRC_CLR bit. Writing 1h to a SRC_CLR bit sets to 0h (clears) both the SRC_CLR and corresponding SRC bit.Table 5-7 summarizes the inter-processor communication (IPC) registers.

Table 5-7. Summary of the IPC Registers

IPC Register	Writing 1h to bit 0 results in:	Writing 1h to one of the bits [31-4] results in:
BOOTCFG_IPCGR0	interrupt generation to the DSP_INTC	setting that bit and the corresponding SRC_CLR bit to 1h
BOOTCFG_IPCGR8	interrupt generation to the ARM_GIC	setting that bit and the corresponding SRC_CLR bit to 1h
BOOTCFG_IPCGR11	interrupt generation to the ICSS0_INTC	setting that bit and the corresponding SRC_CLR bit to 1h
BOOTCFG_IPCGR12	interrupt generation to the ICSS0_INTC	setting that bit and the corresponding SRC_CLR bit to 1h
BOOTCFG_IPCGR13	interrupt generation to the ICSS1_INTC	setting that bit and the corresponding SRC_CLR bit to 1h
BOOTCFG_IPCGR14	interrupt generation to the ICSS1_INTC	setting that bit and the corresponding SRC_CLR bit to 1h
BOOTCFG_IPCGRH	interrupt generation to an external host through the HOUT device pad	setting that bit and the corresponding SRC_CLR bit to 1h
BOOTCFG_IPCAR0	N/A	setting that bit and the corresponding SRC bit to 0h
BOOTCFG_IPCAR8	N/A	setting that bit and the corresponding SRC bit to 0h
BOOTCFG_IPCAR11	N/A	setting that bit and the corresponding SRC bit to 0h
BOOTCFG_IPCAR12	N/A	setting that bit and the corresponding SRC bit to 0h
BOOTCFG_IPCAR13	N/A	setting that bit and the corresponding SRC bit to 0h
BOOTCFG_IPCAR14	N/A	setting that bit and the corresponding SRC bit to 0h
BOOTCFG_IPCARH	N/A	setting that bit and the corresponding SRC bit to 0h

When writing 1h to the BOOTCFG_IPCGRH[0] bit an interrupt pulse on the HOUT device pad is created. This pulse remains asserted for four BOOT_CFG_VBUS_CLK cycles followed by deassertion for four BOOT_CFG_VBUS_CLK cycles. During this period equal to eight BOOT_CFG_VBUS_CLK cycles another write to BOOTCFG_IPCGRH[0] with value of 1h cannot cause another interrupt pulse. To create another interrupt pulse on the HOUT pad, the BOOTCFG_IPCGRH[0] bit must be set to 1h after this eight BOOT_CFG_VBUS_CLK cycles window has already been elapsed.

NOTE: For latency reasons the IPC registers are non kick-protected which means that they can be written to without a need for performing unlocking procedure as described in Section 5.1.3.1.2.

5.1.3.1.6 Registers Associated With Some Device External Signals

Figure 5-2 shows the device external signals and the BOOT_CFG registers associated with these signals. Table 5-8 summarizes these registers.

Figure 5-2. External Signals and Registers Associated with Them

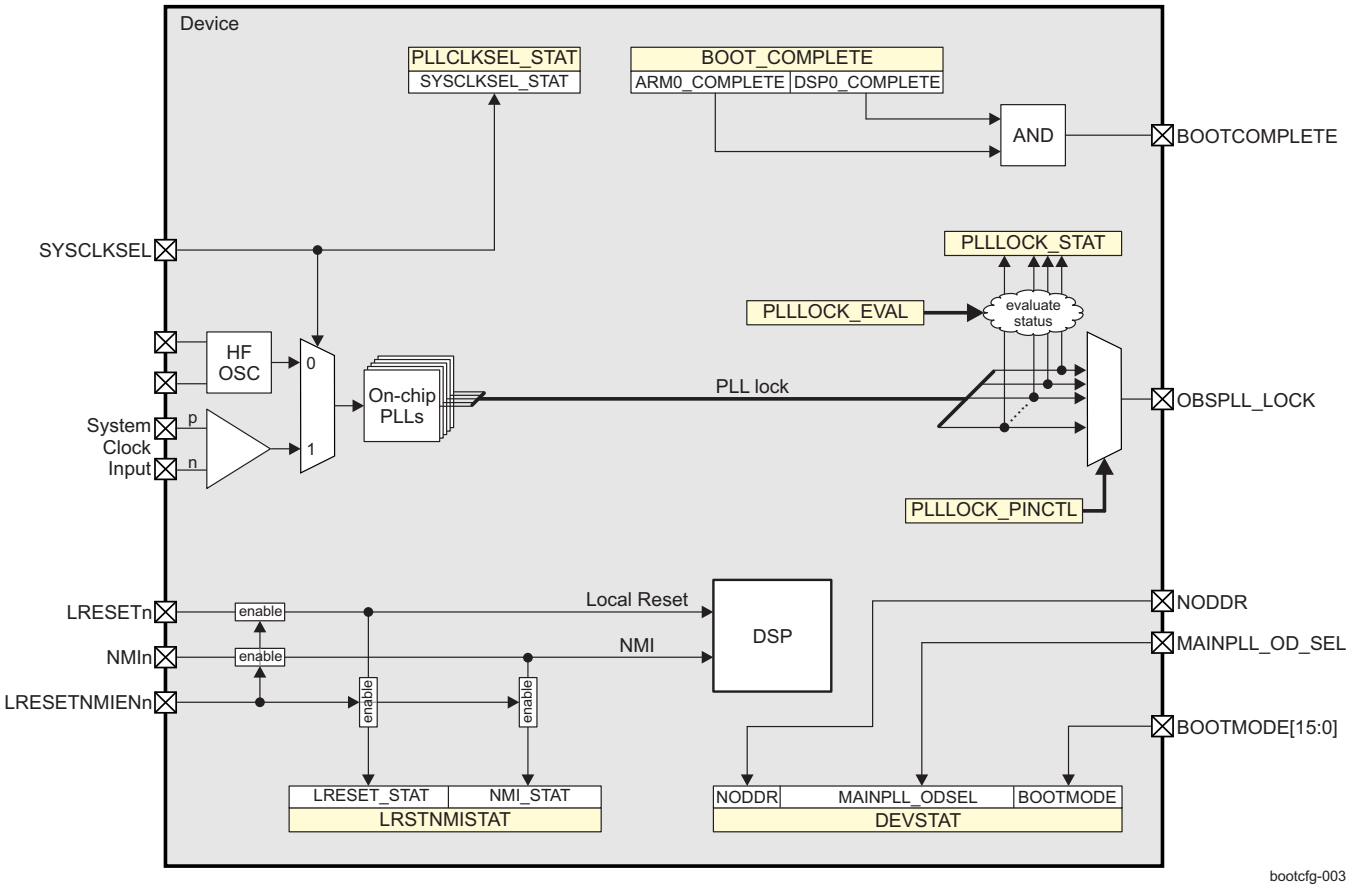


Table 5-8. Summary of the Registers Associated With the Device External Signals

BOOT_CFG Register	Associated External Device Signals	Reference
BOOTCFG_DEVSTAT	BOOTMODE[15-0], MAINPLL_OD_SEL, NODDR	see Section 5.1.3.1.6.1
BOOTCFG_BOOT_COMPLETE	BOOTCOMPLETE	see Section 5.1.3.1.6.2
BOOTCFG_LRSTNMISTAT	LRESETn, NMIn, LRESETNMIENn	see Section 5.1.3.1.6.3
BOOTCFG_LRSTNMISTAT_CLR	Used for clearing the bits in BOOTCFG_LRSTNMISTAT	
BOOTCFG_PLLLOCK_PINCTL	OBSPLL_LOCK	see Section 5.1.3.1.6.4
BOOTCFG_PLLLOCK_STAT		
BOOTCFG_PLLLOCK_EVAL		
BOOTCFG_PLLCLKSEL_STAT	SYSCLKSEL	see Section 5.1.3.1.6.5

5.1.3.1.6.1 BOOTCFG_DEVSTAT Register

The bits in the BOOTCFG_DEVSTAT register reflect the state of some device pads related to boot mode and device configuration. The boot process performed by the boot master is determined by the BOOTCFG_DEVSTAT[16-1] BOOTMODE field which reflects the values present on the BOOTMODE[15-0] pads. For more information about the booting procedure, see Section 4.3.2 in Chapter 4 *Initialization*.

The BOOTCFG_DEVSTAT[19] MAINPLL_ODSEL bit reflects the state of the MAINPLL_OD_SEL pad.

The BOOTCFG_DEVSTAT[20] NODDR bit reflects the state of the NODDR pad.

5.1.3.1.6.2 BOOTCFG_BOOT_COMPLETE Register

The BOOTCOMPLETE device pad indicates the ROM boot process completion to the external system. A value of '1' on the BOOTCOMPLETE pad is present when the ROM code sets to 1h both bits in the BOOTCFG_BOOT_COMPLETE register. One bit is associated with the DSP and the other one with the ARMSS. After device reset a value of '0' is present on the BOOTCOMPLETE pad.

5.1.3.1.6.3 LRSTNMISTAT And BOOTCFG_LRSTNMISTAT_CLR Registers

The LRESETn signal can be used by an external device to generate local reset to the DSP. The NMIn signal can be used by an external device to generate non-maskable interrupt (NMI) to the DSP. The the rising edge of the LRESETNMIENn signal is used to latch the status of the LRESETn and NMIn signals in the LRSTNMISTAT register. Writing a value of 1h to a bit in the BOOTCFG_LRSTNMISTAT_CLR register clears the corresponding bit in the LRSTNMISTAT register.

Table 5-9 shows the LRESETn and NMIn decoding based on the LRESETNMIENn values.

Table 5-9. LRESETn and NMIn Decoding

LRESETn	NMIn	LRESETNMIENn	Action
x	x	1	Neither local reset nor NMI assertion
0	x	0	Assert local reset to the DSP
1	0	0	Assert NMI to the DSP
1	1	0	Local reset and NMI de-assertion

5.1.3.1.6.4 BOOTCFG_PLLLOCK_PINCTL, BOOTCFG_PLLLOCK_STAT And BOOTCFG_PLLLOCK_EVAL Registers

The OBSPLL_LOCK pad is used to indicate when one of the device PLLs is locked. The **x**_LOCK_SEL bits in the BOOTCFG_PLLLOCK_PINCTL register select which PLL lock signal to be routed to the OBSPLL_LOCK device pad. Only one lock signal can be selected at any time. Writing 1h to a **x**_LOCK_SEL bit makes the corresponding PLL lock signal available on the OBSPLL_LOCK pad.

The BOOTCFG_PLLLOCK_STAT register has bits which show the status of each PLL lock signal. This status indicates either PLL lock or loss of PLL lock. The loss of PLL lock is captured in the **x**_LOCK_STAT bits in a sticky fashion. This means that when loss of PLL lock occurs the corresponding **x**_LOCK_STAT bit is set to 0h and it is never set again to 1h even if the corresponding PLL is locked. To reflect the current state of the corresponding PLL lock signal the associated **x**_LOCK_EVAL bit of the BOOTCFG_PLLLOCK_EVAL register must be set to 1h. In case of PLL lock the corresponding **x**_LOCK_STAT is set to 1h. If PLL is not locked a value of 0h is read. In other words, the **x**_LOCK_EVAL bit must be set 1h to update the **x**_LOCK_STAT bit with the actual value of the PLL lock signal of the corresponding on-chip PLL.

Once a PLL is programmed, it is expected that software should set to 1h the **x**_LOCK_EVAL bit to get the current status of a PLL lock signal. In anormal operation all **x**_LOCK_STAT bits show the status as '1' indicating a PLL lock.

5.1.3.1.6.5 BOOTCFG_PLLCLKSEL_STAT Register

Through the BOOTCFG_PLLCLKSEL_STAT[0] SYSCLKSEL_STAT bit the value present on the SYSCLKSEL device pad can be read. This pad is used to select the input clock for the on-chip PLLs. For more information, see Section 5.4, *Clock Management*.

5.1.3.1.7 Event Mux Control Registers

There are event multiplexers in the BOOT_CFG which collect events from the GPIO**x**. By using the BOOT_CFG event mux dedicated registers these events can be mapped to the following DMA and interrupt controllers:

- ARM_GIC
- CIC
- DSP_INTC
- PMMC_INTC
- EDMA_0
- EDMA_1

These events can also be used as a trigger source for the eCAP_**x** modules.

Table 5-10 shows all event mux associated registers and where the events can be mapped to. Each EVTSEL**x** field is associated with one event multiplexer and is an 8-bit field which can select one of 256 input events. Writing of 0h selects input event 0, writing of FFh selects input event 255 to be mapped to the output of the corresponding event multiplexer. Then this event is mapped to module listed in column "**Events Mapped to**". For exact mapping of each event mux output to each interrupt and DMA line, see Table 6-2, *AINTC Interrupt Sources*, Table 6-37, *C66x DSP Interrupt Sources*, Table 9-53, *CIC Input Events Mapping*, Table 10-7, *EDMACC_0 Synchronization Events*, and Table 10-8, *EDMACC_1 Synchronization Events*.

Figure 5-3 shows the event multiplexers. They are also referred to as GPIOMUX.

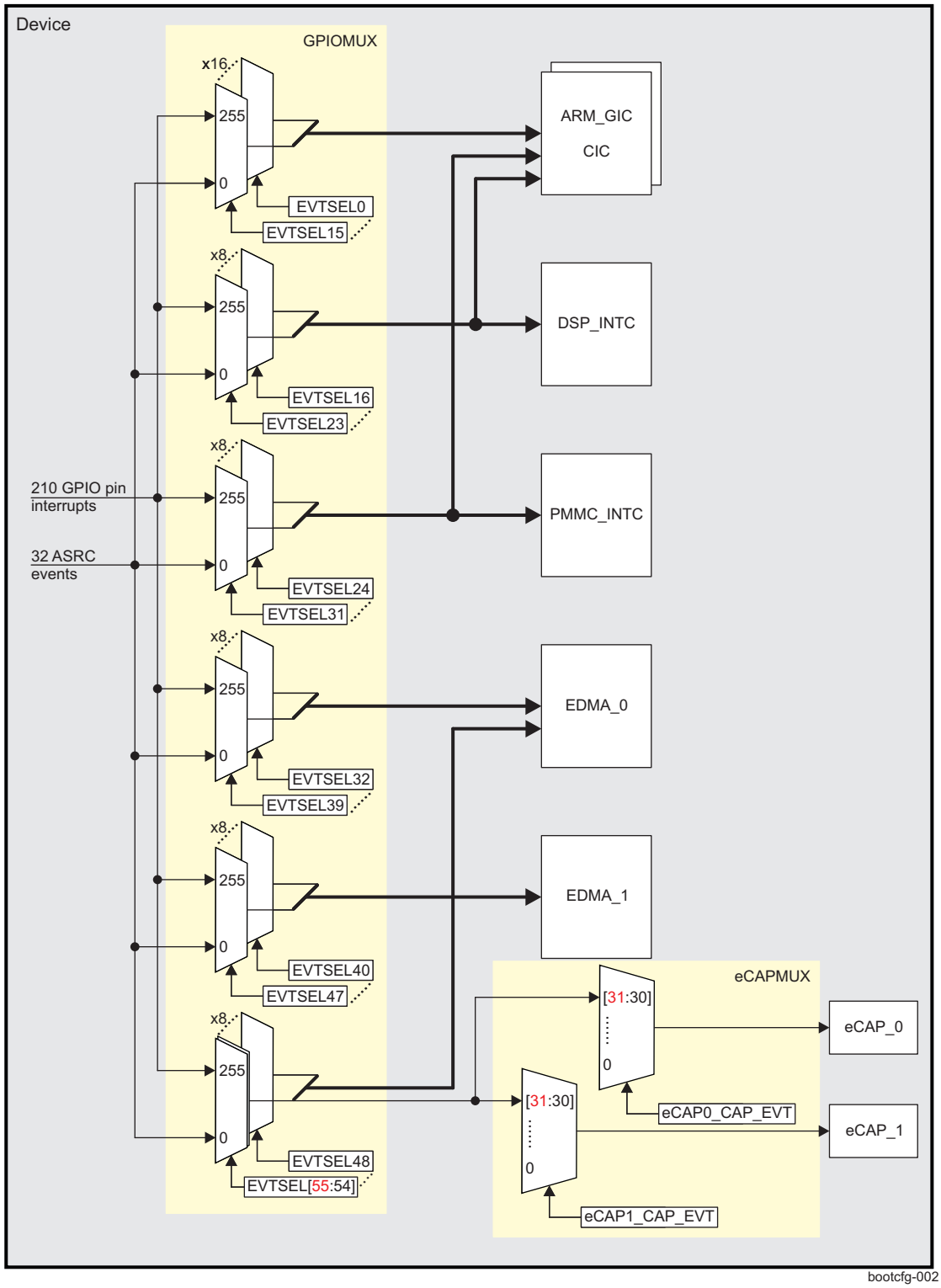
Table 5-10. Summary of the Event Mux Registers

Register	Bit Field	Events Mapped to
BOOTCFG_EVENT_MUXCTL0	EVTSEL0	ARM_GIC; CIC
	EVTSEL1	
	EVTSEL2	
	EVTSEL3	
BOOTCFG_EVENT_MUXCTL1	EVTSEL4	ARM_GIC; CIC
	EVTSEL5	
	EVTSEL6	
	EVTSEL7	
BOOTCFG_EVENT_MUXCTL2	EVTSEL8	ARM_GIC; CIC
	EVTSEL9	
	EVTSEL10	
	EVTSEL11	
BOOTCFG_EVENT_MUXCTL3	EVTSEL12	ARM_GIC; CIC
	EVTSEL13	
	EVTSEL14	
	EVTSEL15	
BOOTCFG_EVENT_MUXCTL4	EVTSEL16	ARM_GIC; CIC; DSP_INTC
	EVTSEL17	
	EVTSEL18	
	EVTSEL19	
BOOTCFG_EVENT_MUXCTL5	EVTSEL20	ARM_GIC; CIC; DSP_INTC
	EVTSEL21	
	EVTSEL22	
	EVTSEL23	
BOOTCFG_EVENT_MUXCTL6	EVTSEL24	ARM_GIC; CIC; PMMC_INTC
	EVTSEL25	
	EVTSEL26	
	EVTSEL27	

Table 5-10. Summary of the Event Mux Registers (continued)

Register	Bit Field	Events Mapped to
BOOTCFG_EVENT_MUXCTL7	EVTSEL28	ARM_GIC; CIC; PMMC_INTC
	EVTSEL29	
	EVTSEL30	
	EVTSEL31	
BOOTCFG_EVENT_MUXCTL8	EVTSEL32	EDMA_0
	EVTSEL33	
	EVTSEL34	
	EVTSEL35	
BOOTCFG_EVENT_MUXCTL9	EVTSEL36	EDMA_0
	EVTSEL37	
	EVTSEL38	
	EVTSEL39	
BOOTCFG_EVENT_MUXCTL10	EVTSEL40	EDMA_1
	EVTSEL41	
	EVTSEL42	
	EVTSEL43	
BOOTCFG_EVENT_MUXCTL11	EVTSEL44	EDMA_1
	EVTSEL45	
	EVTSEL46	
	EVTSEL47	
BOOTCFG_EVENT_MUXCTL12	EVTSEL48	EDMA_0
	EVTSEL49	
	EVTSEL50	
	EVTSEL51	
BOOTCFG_EVENT_MUXCTL13	EVTSEL52	EDMA_0
	EVTSEL53	
	EVTSEL54	EDMA_0; eCAPMUX
	EVTSEL55	

Figure 5-3. Event Multiplexers



5.1.3.1.8 Reset Mux Control Registers

The BOOTCFG_RSTMUX0 and BOOTCFG_RSTMUX8 registers are used to control the two reset muxes in the device. The BOOTCFG_RSTMUX0 register controls the reset mux associated with the DSP and the BOOTCFG_RSTMUX8 register controls the reset mux associated with the ARMSS.

TIMER_0 is dedicated to DSP and TIMER_5 is dedicated to ARMSS. These timers can be used as either general purpose or watch dog (WD) timers. When used as WD timers the events they generate are routed to the two reset muxes. Depending on the BOOTCFG_RSTMUX0[3-1] OMODE0 bit field the TIMER_0 event causes one of the following:

- Local reset to the DSP
- Non-maskable interrupt (NMI) to the DSP
- NMI to the DSP followed by local reset to the DSP
- Device reset

Depending on the BOOTCFG_RSTMUX8[3-1] OMODE8 bit field the TIMER5 event causes one of the following:

- Device reset
- An interrupt to the ARM_GIC (RSTMUX_INT8)
- An interrupt to the ARM_GIC (RSTMUX_INT8) followed by a device reset

The delay between the NMI and the local reset to the DSP is configured by the BOOTCFG_RSTMUX0[7-5] DELAY0 bit field.

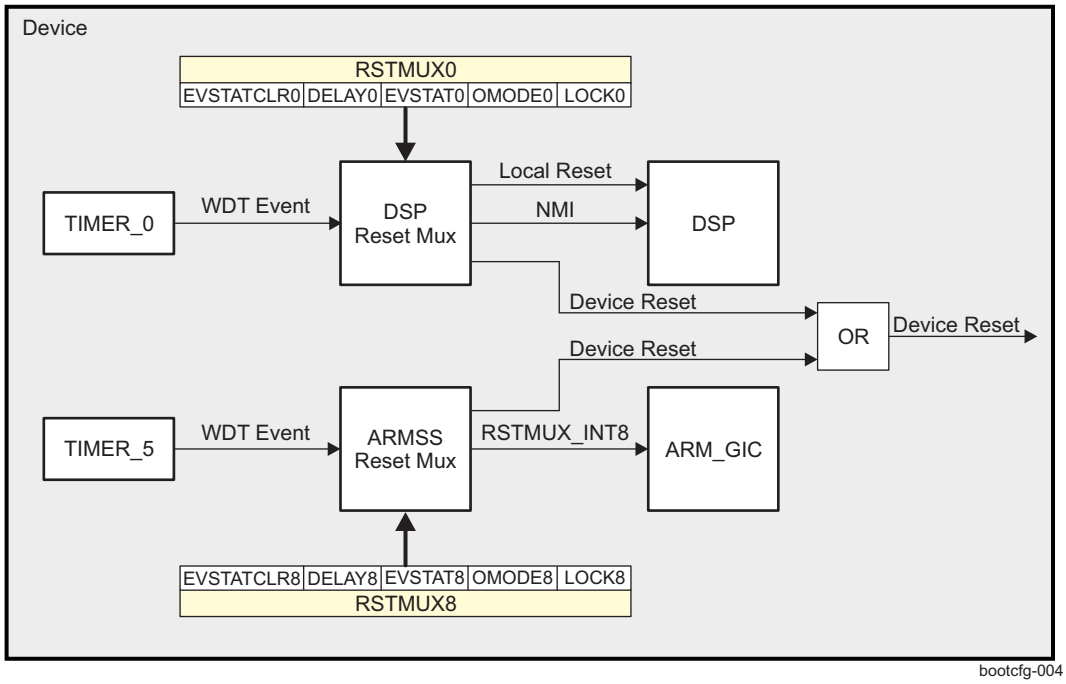
The delay between the interrupt to the ARM_GIC and the device reset is configured by the BOOTCFG_RSTMUX8[7-5] DELAY8 bit field.

When a WD timer event is received by the reset mux this is indicated in the RSTMUX**x**[4] EVSTAT**x** bits. This status is cleared by writing to 1h the EVSTATCLR**x** bit.

NOTE: To prevent any accidental modification of the RSTMUX**x** registers it is recommended to set to 1h the LOCK**x** bits after WD timers are configured.

Figure 5-4 shows the reset mux sheme.

Figure 5-4. Reset Mux Sheme



5.1.3.1.9 Timer Pin Manager Control Registers

Not all timer inputs and outputs are pinned out of the device. Through the Timer Pin Manager the inputs and outputs of the six timers (TIMER_0 through TIMER_5) can be made available on the TIMI[1:0] and TIMO[1:0] device pads.

Each timer input can be configured to be driven by one of the TIMI[1:0] pads through the corresponding bits in the BOOTCFG_TINPSEL0 and BOOTCFG_TINPSEL1 registers. Each of the TIMO[1:0] pads can be configured to be driven by any of the timer outputs through the corresponding bits in the BOOTCFG_TOUTPSEL0 register.

For more information about each timer, see Section 11.17 *Timers*.

5.1.3.1.10 Scrachpad Registers

There are several general purpose read-write registers in the BOOT_CFG memory space intended to be used as a scratchpad. Table 5-11 shows these registers.

Table 5-11. Summary of the Scratchpad Registers

Register Name	Register Name	Register Name	Register Name
BOOTCFG_SCRATCH0	BOOTCFG_SCRATCH4	BOOTCFG_SCRATCH8	BOOTCFG_SCRATCH12
BOOTCFG_SCRATCH1	BOOTCFG_SCRATCH5	BOOTCFG_SCRATCH9	BOOTCFG_SCRATCH13
BOOTCFG_SCRATCH2	BOOTCFG_SCRATCH6	BOOTCFG_SCRATCH10	BOOTCFG_SCRATCH14
BOOTCFG_SCRATCH3	BOOTCFG_SCRATCH7	BOOTCFG_SCRATCH11	BOOTCFG_SCRATCH15

5.1.3.1.11 Registers for Control of the MLB IOs

There are I/O cells specially intended for the MLB 6-pin interface. They are low-voltage differential signaling (LVDS) I/O cells with controls which reside in several registers of the BOOT_CFG module. There is also a bandgap cell intended to provide stable reference voltage to these LVDS IOs across voltage and temperature variations. Without this reference voltage these differential receivers and transmitters cannot function properly. The same bandgap cell also provides a reference voltage to the NSS LVDS buffer associated with signals CPTS_REFCLK_P and CPTS_REFCLK_N. Figure 5-5 shows the registers for controlling the MLB LVDS IOs and bandgap cell. Table 5-12 summarizes and describes these registers.

Figure 5-5. MLB I/O Cells And Their Controls

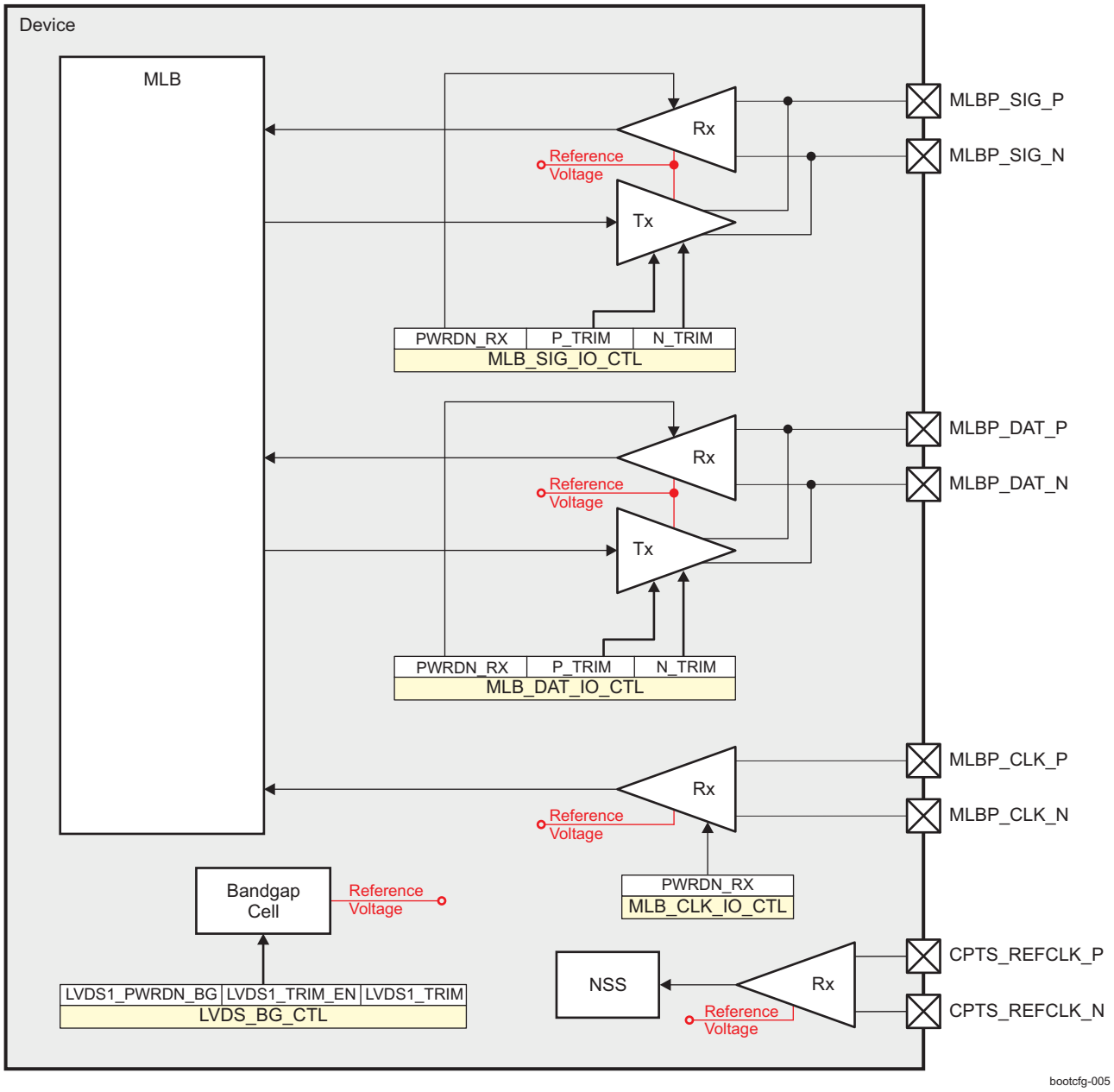


Table 5-12. Summary of the MLB IOs Control Registers

Associated Signals	Register	Bit Field	Description
MLBP_SIG_P MLBP_SIG_N	BOOTCFG_MLB_SIG_IO_CTL	N_TRIM	Trims the driver's NMOS side output impedance.
		P_TRIM	Trims the driver's PMOS side output impedance.
		PWRDN_RX	Turns ON or OFF the MLB LVDS SIG receiver.

Table 5-12. Summary of the MLB IOs Control Registers (continued)

Associated Signals	Register	Bit Field	Description
MLBP_DAT_P MLBP_DAT_N	BOOTCFG_MLB_DAT_IO_CTL	N_TRIM	Trims the driver's NMOS side output impedance.
		P_TRIM	Trims the driver's PMOS side output impedance.
		PWRDN_RX	Turns ON or OFF the MLB LVDS DAT receiver.
MLBP_CLK_P MLBP_CLK_N	BOOTCFG_MLB_CLK_IO_CTL	PWRDN_RX	Turns ON or OFF the MLB LVDS CLK receiver.
N/A	BOOTCFG_LVDS_BG_CTL	LVDS1_PWRDN_BG	Turns ON or OFF the MLB LVDS and NSS LVDS associated bandgap cell.
		LVDS1_TRIM_EN	Trim enable.
		LVDS1_TRIM	Trim value for the reference voltage generated by the bandgap cell.

NOTE: The MLB differential input/output buffers do not contain internal termination resistors, so the appropriate external termination resistors are required when using MLB in 6-pin mode.

NOTE: The BOOTCFG_LVDS_BG_CTL register has also bits for controlling another bandgap cell which provides reference voltage to the LVDS buffers associated with the following signals:

- DDR_CLK_P and DDR_CLK_N
- SYSCLK_P and SYSCLK_N

5.1.3.1.12 Device Reset Status Registers

There are two reset associated registers in the BOOT_CFG memory space. Table 5-13 shows these registers.

Table 5-13. Summary of the Device Reset Status Registers

Register Name	Bit Field	Description
BOOTCFG_RESET_STAT	GRST_STAT	Device global reset indication
	LRST_STAT0	DSP local reset indication
BOOTCFG_RESET_STAT_CLR	GRST_STAT_CLR	Writing 1h clears the BOOTCFG_RESET_STAT[31] GRST_STAT bit
	LRST_STAT_0_CLR	Writing 1h clears the BOOTCFG_RESET_STAT[0] LRST_STAT0 bit

5.1.3.1.13 DSP Boot Address Registers

There are two registers in the BOOT_CFG memory space associated with the DSP boot address. Table 5-14 shows these registers.

Table 5-14. Summary of the DSP Boot Address Associated Registers

Register Name	Description
BOOTCFG_DSP_BOOT_ADDR0	Boot address for the DSP used on secure devices
BOOTCFG_DSP_BOOT_ADDR0_NS	Boot address for the DSP used on non-secure devices

5.1.3.1.14 eCAP/ePWM/eQEP Control and Status Registers

The BOOT_CFG registers which provide control and status information for the eCAP/ePWM/eQEP modules are shown in Table 5-15.

Table 5-15. Summary of the eCAP/ePWM/eQEP Control and Status Registers

Register Name	Description	Associated Functionality Described in:
BOOTCFG_EPWM_CTL	Time base clock and other controls for the ePWM**x** module	Section 11.5.3.4 *ePWM Modules Time Base Clock Gating*, Section 11.5.3.3 *ADC start of conversion signals (EPWM_SOCA and EPWM_SOCB)* and Section 11.5.3.2 *Daisy-Chain Connectivity between ePWM Modules* in Section 11.5 *Enhanced PWM (ePWM) Module*
BOOTCFG_ECAP_CAPEVT_CTL	Selects capture event for the eCAP**x** module	Section 11.4.4.1 *eCAP Input Capture Events* in Section 11.4 *Enhanced Capture (eCAP) Module*
BOOTCFG_EQEP_STAT	Provides eQEP**x** phase error status information	Section 11.6.3.1 *Device Specific eQEP Features* in Section 11.6 *Enhanced Quadrature Encoder Pulse (eQEP) Module*

5.1.3.1.15 NSS MAC Address Registers

Each device has unique 48-bit MAC address which is stored in the BOOT_CFG registers shown in Table 5-16.

Table 5-16. Summary of the NSS MAC Address Registers

Bit Field	Description
BOOTCFG_MACID0[31-0] MACID	Contains the first, second, third and fourth octets of the NSS MAC address
BOOTCFG_MACID1[15-0] MACID	Contains the fifth and sixth octets of the NSS MAC address

5.1.3.1.16 ARMSS Endian Configuration Registers

NOTE: ARMSS supports only little-endian mode of operation. Big-endian mode is not supported on ARMSS.

The BOOT_CFG module has registers intended to provide the ARMSS with a never changing ednian view of the registers of the device peripherals in case of 32-bit accesses. Table 5-17 shows these registers and Table 5-18 lists their reset values.

Table 5-17. Summary of the ARMSS Endian Configuration Registers

Register Name	Register Name	Register Name	ARMSS Region
BOOTCFG_ARMENDIAN_CFGx_0	BOOTCFG_ARMENDIAN_CFGx_1	BOOTCFG_ARMENDIAN_CFGx_2	x (x = 0 to 7)

Table 5-18. BOOTCFG_ARMENDIAN_CFGx_0 to BOOTCFG_ARMENDIAN_CFGx_2 Reset Values

Register Name	Reset Value	Register Name	Reset Value
BOOTCFG_ARMENDIAN_CFG0_0	1C000h	BOOTCFG_ARMENDIAN_CFG4_0	23A00h
BOOTCFG_ARMENDIAN_CFG0_1	6h	BOOTCFG_ARMENDIAN_CFG4_1	5h
BOOTCFG_ARMENDIAN_CFG0_2	1h	BOOTCFG_ARMENDIAN_CFG4_2	1h
BOOTCFG_ARMENDIAN_CFG1_0	20000h	BOOTCFG_ARMENDIAN_CFG5_0	240000h
BOOTCFG_ARMENDIAN_CFG1_1	9h	BOOTCFG_ARMENDIAN_CFG5_1	6h
BOOTCFG_ARMENDIAN_CFG1_2	1h	BOOTCFG_ARMENDIAN_CFG5_2	1h
BOOTCFG_ARMENDIAN_CFG2_0	BC000h	BOOTCFG_ARMENDIAN_CFG6_0	1000000h
BOOTCFG_ARMENDIAN_CFG2_1	4h	BOOTCFG_ARMENDIAN_CFG6_1	0h
BOOTCFG_ARMENDIAN_CFG2_2	1h	BOOTCFG_ARMENDIAN_CFG6_2	1h
BOOTCFG_ARMENDIAN_CFG3_0	210000h	BOOTCFG_ARMENDIAN_CFG7_0	FFFFFF00h
BOOTCFG_ARMENDIAN_CFG3_1	8h	BOOTCFG_ARMENDIAN_CFG7_1	0h
BOOTCFG_ARMENDIAN_CFG3_2	1h	BOOTCFG_ARMENDIAN_CFG7_2	1h

5.1.3.1.17 Registers Associated with the ARMSS and DEBUG_SS Trace Buffers

There are registers in the BOOT_CFG module memory space which are intended to hold USB Transfer Request Blocks (TRBs) used in case draining of the ARMSS and Debug subsystem trace buffers is needed. Table 5-19 shows these registers. Corresponding shadow registers are also present. Table 5-20 lists the reset values of these registers.

Table 5-19. Summary of the ARMSS and DEBUG_SS Trace Buffer Associated Registers

ARMSS Trace Buffer Registers	ARMSS Trace Buffer Shadow Registers	Debug Subsystem Trace Buffer Registers	Debug Subsystem Trace Buffer Shadow Registers	TRBx Word (x = 0 to 2)
BOOTCFG_ARMTBR_TRBx_W0	BOOTCFG_ARMTBR_SHDW_TRBx_W0	BOOTCFG_DBGTBR_TRBx_W0	BOOTCFG_DBGTBR_SHDW_TRBx_W0	0
BOOTCFG_ARMTBR_TRBx_W1	BOOTCFG_ARMTBR_SHDW_TRBx_W1	BOOTCFG_DBGTBR_TRBx_W1	BOOTCFG_DBGTBR_SHDW_TRBx_W1	1
BOOTCFG_ARMTBR_TRBx_W2	BOOTCFG_ARMTBR_SHDW_TRBx_W2	BOOTCFG_DBGTBR_TRBx_W2	BOOTCFG_DBGTBR_SHDW_TRBx_W2	2
BOOTCFG_ARMTBR_TRBx_W3	BOOTCFG_ARMTBR_SHDW_TRBx_W3	BOOTCFG_DBGTBR_TRBx_W3	BOOTCFG_DBGTBR_SHDW_TRBx_W3	3

Table 5-20. Reset Values of the ARMSS and DEBUG_SS Trace Buffer Associated Registers

Register Name	Reset Value	Register Name	Reset Value
BOOTCFG_ARMTBR_TRB0_W0	20700000h	BOOTCFG_DBGTBR_TRB0_W0	20600000h
BOOTCFG_ARMTBR_TRB0_W1	0h	BOOTCFG_DBGTBR_TRB0_W1	0h
BOOTCFG_ARMTBR_TRB0_W2	3C00h	BOOTCFG_DBGTBR_TRB0_W2	7C00h
BOOTCFG_ARMTBR_TRB0_W3	11h	BOOTCFG_DBGTBR_TRB0_W3	11h
BOOTCFG_ARMTBR_TRB1_W0	20700000h	BOOTCFG_DBGTBR_TRB1_W0	20600000h
BOOTCFG_ARMTBR_TRB1_W1	0h	BOOTCFG_DBGTBR_TRB1_W1	0h
BOOTCFG_ARMTBR_TRB1_W2	3C00h	BOOTCFG_DBGTBR_TRB1_W2	7C00h
BOOTCFG_ARMTBR_TRB1_W3	11h	BOOTCFG_DBGTBR_TRB1_W3	11h
BOOTCFG_ARMTBR_TRB2_W0	26204C0h	BOOTCFG_DBGTBR_TRB2_W0	2620540h
BOOTCFG_ARMTBR_TRB2_W1	0h	BOOTCFG_DBGTBR_TRB2_W1	0h
BOOTCFG_ARMTBR_TRB2_W2	0h	BOOTCFG_DBGTBR_TRB2_W2	0h
BOOTCFG_ARMTBR_TRB2_W3	81h	BOOTCFG_DBGTBR_TRB2_W3	81h

Table 5-20. Reset Values of the ARMSS and DEBUG_SS Trace Buffer Associated Registers (continued)

Register Name	Reset Value	Register Name	Reset Value
BOOTCFG_ARMTBR_SHDW_TRB0_W0	20700000h	BOOTCFG_DBGTBR_SHDW_TRB0_W0	20600000h
BOOTCFG_ARMTBR_SHDW_TRB0_W1	0h	BOOTCFG_DBGTBR_SHDW_TRB0_W1	0h
BOOTCFG_ARMTBR_SHDW_TRB0_W2	3C00h	BOOTCFG_DBGTBR_SHDW_TRB0_W2	7C00h
BOOTCFG_ARMTBR_SHDW_TRB0_W3	11h	BOOTCFG_DBGTBR_SHDW_TRB0_W3	11h
BOOTCFG_ARMTBR_SHDW_TRB1_W0	20700000h	BOOTCFG_DBGTBR_SHDW_TRB1_W0	20600000h
BOOTCFG_ARMTBR_SHDW_TRB1_W1	0h	BOOTCFG_DBGTBR_SHDW_TRB1_W1	0h
BOOTCFG_ARMTBR_SHDW_TRB1_W2	3C00h	BOOTCFG_DBGTBR_SHDW_TRB1_W2	7C00h
BOOTCFG_ARMTBR_SHDW_TRB1_W3	11h	BOOTCFG_DBGTBR_SHDW_TRB1_W3	11h
BOOTCFG_ARMTBR_SHDW_TRB2_W0	26204C0h	BOOTCFG_DBGTBR_SHDW_TRB2_W0	2620540h
BOOTCFG_ARMTBR_SHDW_TRB2_W1	0h	BOOTCFG_DBGTBR_SHDW_TRB2_W1	0h
BOOTCFG_ARMTBR_SHDW_TRB2_W2	0h	BOOTCFG_DBGTBR_SHDW_TRB2_W2	0h
BOOTCFG_ARMTBR_SHDW_TRB2_W3	81h	BOOTCFG_DBGTBR_SHDW_TRB2_W3	81h

5.1.3.1.18 PLL Control Registers

The registers for controlling the device PLLs reside in the BOOT_CFG module memory space. Table 5-21 summarizes these registers. More information can be found in Section 5.4.5 *PLLs*.

Table 5-21. Summary of the PLL Control Registers

Register Name	Register Name
BOOTCFG_MAIN_PLL_CTL0	BOOTCFG_MAIN_PLL_CTL1
BOOTCFG_NSS_PLL_CTL0	BOOTCFG_NSS_PLL_CTL1
BOOTCFG_DDR3A_PLL_CTL0	BOOTCFG_DDR3A_PLL_CTL1
BOOTCFG_ARM_PLL_CTL0	BOOTCFG_ARM_PLL_CTL1
BOOTCFG_DSS_PLL_CTL0	BOOTCFG_DSS_PLL_CTL1
BOOTCFG_ICSS_PLL_CTL0	BOOTCFG_ICSS_PLL_CTL1
BOOTCFG_UART_PLL_CTL0	BOOTCFG_UART_PLL_CTL1

5.1.3.1.19 Clock Muxing, Enabling and Division Registers

There are several registers within the BOOT_CFG module address space dedicated to clock muxing, clock enabling/disabling and clock division for some device modules. Table 5-22 summarizes these registers. Related information can also be found in Section 5.4 *Clock Management*.

Table 5-22. Summary of the Clock Muxing, Enabling and Division Registers

Register Name	Register Name
BOOTCFG_DDR_CLKCTL	BOOTCFG_ICSS_CLKCTL
BOOTCFG_USB0_CLKCTL	BOOTCFG_USB1_CLKCTL
BOOTCFG_ETHERNET_CLKCTL	BOOTCFG_PCIE_CLKCTL
BOOTCFG_SERIALPORT_CLKCTL	

5.1.3.1.20 Registers for Control of Some Device LDOs

The BOOTCFG_LDO_USB_CTL and BOOTCFG_LDO_PCIE_CTL registers have bits to program the USB and PCIe LDOs.

5.1.3.1.21 USB PHY Control Registers

The registers for controlling the USB0 and USB1 PHYs are shown in Table 5-23.

Table 5-23. Summary of the USB0 and USB1 PHY Control Registers

Register Name	Register Name
BOOTCFG_USB0_PHY_CTL0	BOOTCFG_USB1_PHY_CTL0
BOOTCFG_USB0_PHY_CTL1	BOOTCFG_USB1_PHY_CTL1
BOOTCFG_USB0_PHY_CTL2	BOOTCFG_USB1_PHY_CTL2
BOOTCFG_USB0_PHY_CTL4	BOOTCFG_USB1_PHY_CTL4

5.1.3.1.22 ID Registers

The following ID registers are present in the BOOT_CFG memory space:

- BOOTCFG_REVISION
- BOOTCFG_JTAGID
- BOOTCFG_PCIEVENDORID

5.1.4 BOOT_CFG Registers

Table 5-25 lists the memory-mapped registers for the BOOT_CFG. All register offset addresses not listed in Table 5-25 should be considered as reserved locations and the register contents should not be modified.

Table 5-24. BOOT_CFG Instances

Instance	Physical Address
BOOT_CFG	0262 0000h

Table 5-25. BOOT_CFG Registers

Offset	Acronym	Register Name	BOOT_CFG Physical Address	Section
0h	BOOTCFG_REVISION	BOOT_CFG Module Revision ID Register	0262 0000h	Section 5.1.4.1
18h	BOOTCFG_JTAGID	JTAG/DEVICE ID Register	0262 0018h	Section 5.1.4.2
20h	BOOTCFG_DEVSTAT	Device Status Register	0262 0020h	Section 5.1.4.3
38h	BOOTCFG_KICK0	Kick Register 0	0262 0038h	Section 5.1.4.4
3Ch	BOOTCFG_KICK1	Kick Register 1	0262 003Ch	Section 5.1.4.5
40h	BOOTCFG_DSP_BOOT_ADDR0	DSP Boot Address Register	0262 0040h	Section 5.1.4.6
E0h	BOOTCFG_INTR_RAW_STAT_SET	BOOT_CFG Module Interrupt Raw Status/Set Register	0262 00E0h	Section 5.1.4.7
E4h	BOOTCFG_INTR_ENABLED_STAT_CLR	BOOT_CFG Module Interrupt Enabled Status/Clear Register	0262 00E4h	Section 5.1.4.8
E8h	BOOTCFG_INTR_ENABLE	BOOT_CFG Module Interrupt Enable Register	0262 00E8h	Section 5.1.4.9
ECh	BOOTCFG_INTR_ENABLE_CLR	BOOT_CFG Module Interrupt Enable Clear Register	0262 00ECh	Section 5.1.4.10
F0h	BOOTCFG_EOI	BOOT_CFG Module EOI Register	0262 00F0h	Section 5.1.4.11
F4h	BOOTCFG_FAULT_ADDR	BOOT_CFG Module Fault Address Register	0262 00F4h	Section 5.1.4.12
F8h	BOOTCFG_FAULT_STAT	BOOT_CFG Module Fault Status Register	0262 00F8h	Section 5.1.4.13
FCh	BOOTCFG_FAULT_CLR	BOOT_CFG Module Fault Clear Register	0262 00FCh	Section 5.1.4.14
110h	BOOTCFG_MACID0	MAC ID0 Register	0262 0110h	Section 5.1.4.15
114h	BOOTCFG_MACID1	MAC ID1 Register	0262 0114h	Section 5.1.4.16
128h	BOOTCFG_PCIEVENDORID	PCIE Vendor ID Register	0262 0128h	Section 5.1.4.17
130h	BOOTCFG_LRSTNMISTAT_CLR	LRESETNMI Pin Status Clear Register	0262 0130h	Section 5.1.4.18
134h	BOOTCFG_RESET_STAT_CLR	Reset Status Clear Register	0262 0134h	Section 5.1.4.19
13Ch	BOOTCFG_BOOT_COMPLETE	Boot Complete Register	0262 013Ch	Section 5.1.4.20
144h	BOOTCFG_RESET_STAT	Reset Status Register	0262 0144h	Section 5.1.4.21
148h	BOOTCFG_LRSTNMISTAT	LRESETNMI Pin Status Register	0262 0148h	Section 5.1.4.22
14Ch	BOOTCFG_DEVCFG	Device Configuration Register	0262 014Ch	Section 5.1.4.23
150h	BOOTCFG_PWR_STATE	Power State Control Register	0262 0150h	Section 5.1.4.24
154h	BOOTCFG_INITIATOR_PRIORITY0	Initiator Priority Control 0 Register	0262 0154h	Section 5.1.4.25
158h	BOOTCFG_INITIATOR_PRIORITY1	Initiator Priority Control 1 Register	0262 0158h	Section 5.1.4.26
200h	BOOTCFG_NMIGR0	NMI Generation Register	0262 0200h	Section 5.1.4.27
240h	BOOTCFG_IPCGR0	Inter-Processor Communication Register	0262 0240h	Section 5.1.4.28
260h	BOOTCFG_IPCGR8	Inter-Processor Communication Register	0262 0260h	Section 5.1.4.29

Table 5-25. BOOT_CFG Registers (continued)

Offset	Acronym	Register Name	BOOT_CFG Physical Address	Section
26Ch	BOOTCFG_IPCGR11	Inter-Processor Communication Register	0262 026Ch	Section 5.1.4.30
270h	BOOTCFG_IPCGR12	Inter-Processor Communication Register	0262 0270h	Section 5.1.4.31
274h	BOOTCFG_IPCGR13	Inter-Processor Communication Register	0262 0274h	Section 5.1.4.32
278h	BOOTCFG_IPCGR14	Inter-Processor Communication Register	0262 0278h	Section 5.1.4.33
27Ch	BOOTCFG_IPCGRH	Inter-Processor Communication Register	0262 027Ch	Section 5.1.4.34
280h	BOOTCFG_IPCAR0	Inter-Processor Communication Register	0262 0280h	Section 5.1.4.35
2A0h	BOOTCFG_IPCAR8	Inter-Processor Communication Register	0262 02A0h	Section 5.1.4.36
2ACh	BOOTCFG_IPCAR11	Inter-Processor Communication Register	0262 02ACh	Section 5.1.4.37
2B0h	BOOTCFG_IPCAR12	Inter-Processor Communication Register	0262 02B0h	Section 5.1.4.38
2B4h	BOOTCFG_IPCAR13	Inter-Processor Communication Register	0262 02B4h	Section 5.1.4.39
2B8h	BOOTCFG_IPCAR14	Inter-Processor Communication Register	0262 02B8h	Section 5.1.4.40
2BCh	BOOTCFG_IPCARH	Inter-Processor Communication Register	0262 02BCh	Section 5.1.4.41
2D8h	BOOTCFG_TINPSEL0	Timer Input Selection Register 0	0262 02D8h	Section 5.1.4.42
2DCh	BOOTCFG_TINPSEL1	Timer Input Selection Register 1	0262 02DCh	Section 5.1.4.43
2F8h	BOOTCFG_TOUTPSEL0	Timer Output Selection Register	0262 02F8h	Section 5.1.4.44
308h	BOOTCFG_RSTMUX0	Reset Mux Register 0	0262 0308h	Section 5.1.4.45
328h	BOOTCFG_RSTMUX8	Reset Mux Register 8	0262 0328h	Section 5.1.4.46
350h	BOOTCFG_MAIN_PLL_CTL0	MAIN PLL Control Register 0	0262 0350h	Section 5.1.4.47
354h	BOOTCFG_MAIN_PLL_CTL1	MAIN PLL Control Register 1	0262 0354h	Section 5.1.4.48
358h	BOOTCFG_NSS_PLL_CTL0	NSS PLL Control Register 0	0262 0358h	Section 5.1.4.49
35Ch	BOOTCFG_NSS_PLL_CTL1	NSS PLL Control Register 1	0262 035Ch	Section 5.1.4.50
360h	BOOTCFG_DDR3A_PLL_CTL0	DDR3A PLL Control Register 0	0262 0360h	Section 5.1.4.51
364h	BOOTCFG_DDR3A_PLL_CTL1	DDR3A PLL Control Register 1	0262 0364h	Section 5.1.4.52
370h	BOOTCFG_ARM_PLL_CTL0	ARM PLL Control Register 0	0262 0370h	Section 5.1.4.53
374h	BOOTCFG_ARM_PLL_CTL1	ARM PLL Control Register 1	0262 0374h	Section 5.1.4.54
380h	BOOTCFG_DSS_PLL_CTL0	DSS PLL Control Register 0	0262 0380h	Section 5.1.4.55
384h	BOOTCFG_DSS_PLL_CTL1	DSS PLL Control Register 1	0262 0384h	Section 5.1.4.56
388h	BOOTCFG_ICSS_PLL_CTL0	ICSS PLL Control Register 0	0262 0388h	Section 5.1.4.57
38Ch	BOOTCFG_ICSS_PLL_CTL1	ICSS PLL Control Register 1	0262 038Ch	Section 5.1.4.58
390h	BOOTCFG_UART_PLL_CTL0	UART PLL Control Register 0	0262 0390h	Section 5.1.4.59
394h	BOOTCFG_UART_PLL_CTL1	UART PLL Control Register 1	0262 0394h	Section 5.1.4.60
400h + (x*10h)	BOOTCFG_ARMENDIAN_CFGx_0 (x = 0 to 7)	ARMSS Endian Configuration Register 0 for Region x	0262 0400h + (x*10h)	Section 5.1.4.61
404h + (x*10h)	BOOTCFG_ARMENDIAN_CFGx_1 (x = 0 to 7)	ARMSS Endian Configuration Register 1 for Region x	0262 0404h + (x*10h)	Section 5.1.4.62
408h + (x*10h)	BOOTCFG_ARMENDIAN_CFGx_2 (x = 0 to 7)	ARMSS Endian Configuration Register 2 for Region x	0262 0408h + (x*10h)	Section 5.1.4.63
480h + (x*10h)	BOOTCFG_ARMTBR_TRBx_W0 (x = 0 to 2)	ARMSS Trace Buffer TRBx Register 0	0262 0480h + (x*10h)	Section 5.1.4.64

Table 5-25. BOOT_CFG Registers (continued)

Offset	Acronym	Register Name	BOOT_CFG Physical Address	Section
484h + (x*10h)	BOOTCFG_ARMTBR_TRBx_W1 (x = 0 to 2)	ARMSS Trace Buffer TRBx Register 1	0262 0484h + (x*10h)	Section 5.1.4.65
488h + (x*10h)	BOOTCFG_ARMTBR_TRBx_W2 (x = 0 to 2)	ARMSS Trace Buffer TRBx Register 2	0262 0488h + (x*10h)	Section 5.1.4.66
48Ch + (x*10h)	BOOTCFG_ARMTBR_TRBx_W3 (x = 0 to 2)	ARMSS Trace Buffer TRBx Register 3	0262 048Ch + (x*10h)	Section 5.1.4.67
4C0h + (x*10h)	BOOTCFG_ARMTBR_SHDW_TRBx_W0 (x = 0 to 2)	ARMSS Trace Buffer TRBx Shadow Register 0	0262 04C0h + (x*10h)	Section 5.1.4.68
4C4h + (x*10h)	BOOTCFG_ARMTBR_SHDW_TRBx_W1 (x = 0 to 2)	ARMSS Trace Buffer TRBx Shadow Register 1	0262 04C4h + (x*10h)	Section 5.1.4.69
4C8h + (x*10h)	BOOTCFG_ARMTBR_SHDW_TRBx_W2 (x = 0 to 2)	ARMSS Trace Buffer TRBx Shadow Register 2	0262 04C8h + (x*10h)	Section 5.1.4.70
4CCh + (x*10h)	BOOTCFG_ARMTBR_SHDW_TRBx_W3 (x = 0 to 2)	ARMSS Trace Buffer TRBx Shadow Register 3	0262 04CCh + (x*10h)	Section 5.1.4.71
500h + (x*10h)	BOOTCFG_DBGTBR_TRBx_W0 (x = 0 to 2)	Debug Subsystem Trace Buffer TRBx Register 0	0262 0500h + (x*10h)	Section 5.1.4.72
504h + (x*10h)	BOOTCFG_DBGTBR_TRBx_W1 (x = 0 to 2)	Debug Subsystem Trace Buffer TRBx Register 1	0262 0504h + (x*10h)	Section 5.1.4.73
508h + (x*10h)	BOOTCFG_DBGTBR_TRBx_W2 (x = 0 to 2)	Debug Subsystem Trace Buffer TRBx Register 2	0262 0508h + (x*10h)	Section 5.1.4.74
50Ch + (x*10h)	BOOTCFG_DBGTBR_TRBx_W3 (x = 0 to 2)	Debug Subsystem Trace Buffer TRBx Register 3	0262 050Ch + (x*10h)	Section 5.1.4.75
540h + (x*10h)	BOOTCFG_DBGTBR_SHDW_TRBx_W0 (x = 0 to 2)	Debug Subsystem Trace Buffer TRBx Shadow Register 0	0262 0540h + (x*10h)	Section 5.1.4.76
544h + (x*10h)	BOOTCFG_DBGTBR_SHDW_TRBx_W1 (x = 0 to 2)	Debug Subsystem Trace Buffer TRBx Shadow Register 1	0262 0544h + (x*10h)	Section 5.1.4.77
548h + (x*10h)	BOOTCFG_DBGTBR_SHDW_TRBx_W2 (x = 0 to 2)	Debug Subsystem Trace Buffer TRBx Shadow Register 2	0262 0548h + (x*10h)	Section 5.1.4.78
54Ch + (x*10h)	BOOTCFG_DBGTBR_SHDW_TRBx_W3 (x = 0 to 2)	Debug Subsystem Trace Buffer TRBx Shadow Register 3	0262 054Ch + (x*10h)	Section 5.1.4.79
654h	BOOTCFG_SPARE1	DSS Divider Control Register	0262 0654h	Section 5.1.4.80
690h	BOOTCFG_DDR_CLKCTL	DDR Clock Control Register	0262 0690h	Section 5.1.4.81
694h	BOOTCFG_ICSS_CLKCTL	ICSS Clock Control Register	0262 0694h	Section 5.1.4.82
698h	BOOTCFG_ETHERNET_CLKCTL	Ethernet Clock Control Register	0262 0698h	Section 5.1.4.83
69Ch	BOOTCFG_USB0_CLKCTL	USB0 Clock Control Register	0262 069Ch	Section 5.1.4.84
6A0h	BOOTCFG_USB1_CLKCTL	USB1 Clock Control Register	0262 06A0h	Section 5.1.4.85
6A4h	BOOTCFG_SERIALPORT_CLKCTL	Serial Port Clock Control Register	0262 06A4h	Section 5.1.4.86
6A8h	BOOTCFG_OSC_CTL	Oscillator Control Register	0262 06A8h	Section 5.1.4.87
6ACh	BOOTCFG_PCIE_CLKCTL	PCIE Clock Control Register	0262 06ACh	Section 5.1.4.88
700h	BOOTCFG_CHIP_MISC_CTL0	Chip Miscellaneous Control Register	0262 0700h	Section 5.1.4.89
710h	BOOTCFG_SYSENDSTAT	System Endian Status Register	0262 0710h	Section 5.1.4.90
714h	BOOTCFG_PLLLOCK_PINCTL	PLL Lock Pin Control Register	0262 0714h	Section 5.1.4.91
718h	BOOTCFG_PLLLOCK_STAT	PLL Lock Pin Status Register	0262 0718h	Section 5.1.4.92
71Ch	BOOTCFG_PLLLOCK_EVAL	PLL Lock Pin Eval Register	0262 071Ch	Section 5.1.4.93
720h	BOOTCFG_PLLCLKSEL_STAT	PLL Input Clock Selection Status Register	0262 0720h	Section 5.1.4.94
738h	BOOTCFG_USB0_PHY_CTL0	USB0 PHY Control Register 0	0262 0738h	Section 5.1.4.95
73Ch	BOOTCFG_USB0_PHY_CTL1	USB0 PHY Control Register 1	0262 073Ch	Section 5.1.4.96
740h	BOOTCFG_USB0_PHY_CTL2	USB0 PHY Control Register 2	0262 0740h	Section 5.1.4.97
748h	BOOTCFG_USB0_PHY_CTL4	USB0 PHY Control Register 4	0262 0748h	Section 5.1.4.98
750h	BOOTCFG_USB1_PHY_CTL0	USB1 PHY Control Register 0	0262 0750h	Section 5.1.4.99

Table 5-25. BOOT_CFG Registers (continued)

Offset	Acronym	Register Name	BOOT_CFG Physical Address	Section
754h	BOOTCFG_USB1_PHY_CTL1	USB1 PHY Control Register 1	0262 0754h	Section 5.1.4.100
758h	BOOTCFG_USB1_PHY_CTL2	USB1 PHY Control Register 2	0262 0758h	Section 5.1.4.101
760h	BOOTCFG_USB1_PHY_CTL4	USB1 PHY Control Register 4	0262 0760h	Section 5.1.4.102
768h	BOOTCFG_USB0_EBC_IN_CTL	USB0 External Buffer Control Register	0262 0768h	Section 5.1.4.103
76Ch	BOOTCFG_USB1_EBC_IN_CTL	USB1 External Buffer Control Register	0262 076Ch	Section 5.1.4.104
780h	BOOTCFG_SCRATCH0	Scratchpad Register 0	0262 0780h	Section 5.1.4.105
784h	BOOTCFG_SCRATCH1	Scratchpad Register 1	0262 0784h	Section 5.1.4.106
788h	BOOTCFG_SCRATCH2	Scratchpad Register 2	0262 0788h	Section 5.1.4.107
78Ch	BOOTCFG_SCRATCH3	Scratchpad Register 3	0262 078Ch	Section 5.1.4.108
790h	BOOTCFG_SCRATCH4	Scratchpad Register 4	0262 0790h	Section 5.1.4.109
794h	BOOTCFG_SCRATCH5	Scratchpad Register 5	0262 0794h	Section 5.1.4.110
798h	BOOTCFG_SCRATCH6	Scratchpad Register 6	0262 0798h	Section 5.1.4.111
79Ch	BOOTCFG_SCRATCH7	Scratchpad Register 7	0262 079Ch	Section 5.1.4.112
7A0h	BOOTCFG_SCRATCH8	Scratchpad Register 8	0262 07A0h	Section 5.1.4.113
7A4h	BOOTCFG_SCRATCH9	Scratchpad Register 9	0262 07A4h	Section 5.1.4.114
7A8h	BOOTCFG_SCRATCH10	Scratchpad Register 10	0262 07A8h	Section 5.1.4.115
7ACh	BOOTCFG_SCRATCH11	Scratchpad Register 11	0262 07ACh	Section 5.1.4.116
7B0h	BOOTCFG_SCRATCH12	Scratchpad Register 12	0262 07B0h	Section 5.1.4.117
7B4h	BOOTCFG_SCRATCH13	Scratchpad Register 13	0262 07B4h	Section 5.1.4.118
7B8h	BOOTCFG_SCRATCH14	Scratchpad Register 14	0262 07B8h	Section 5.1.4.119
7BCh	BOOTCFG_SCRATCH15	Scratchpad Register 15	0262 07BCh	Section 5.1.4.120
844h	BOOTCFG_DSP_BOOT_ADDR0_NS	DSP Non-secure Boot Address Register	0262 0844h	Section 5.1.4.121
C80h	BOOTCFG_OBSCLKCTL	Observation Clock Control Register	0262 0C80h	Section 5.1.4.122
C90h	BOOTCFG_EFUSE_BOOTROM	EFUSE BOOTROM Register	0262 0C90h	Section 5.1.4.123
D00h	BOOTCFG_EVENT_MUXCTL0	Event Mux Control Register 0	0262 0D00h	Section 5.1.4.124
D04h	BOOTCFG_EVENT_MUXCTL1	Event Mux Control Register 1	0262 0D04h	Section 5.1.4.125
D08h	BOOTCFG_EVENT_MUXCTL2	Event Mux Control Register 2	0262 0D08h	Section 5.1.4.126
D0Ch	BOOTCFG_EVENT_MUXCTL3	Event Mux Control Register 3	0262 0D0Ch	Section 5.1.4.127
D10h	BOOTCFG_EVENT_MUXCTL4	Event Mux Control Register 4	0262 0D10h	Section 5.1.4.128
D14h	BOOTCFG_EVENT_MUXCTL5	Event Mux Control Register 5	0262 0D14h	Section 5.1.4.129
D18h	BOOTCFG_EVENT_MUXCTL6	Event Mux Control Register 6	0262 0D18h	Section 5.1.4.130
D1Ch	BOOTCFG_EVENT_MUXCTL7	Event Mux Control Register 7	0262 0D1Ch	Section 5.1.4.131
D20h	BOOTCFG_EVENT_MUXCTL8	Event Mux Control Register 8	0262 0D20h	Section 5.1.4.132
D24h	BOOTCFG_EVENT_MUXCTL9	Event Mux Control Register 9	0262 0D24h	Section 5.1.4.133
D28h	BOOTCFG_EVENT_MUXCTL10	Event Mux Control Register 10	0262 0D28h	Section 5.1.4.134
D2Ch	BOOTCFG_EVENT_MUXCTL11	Event Mux Control Register 11	0262 0D2Ch	Section 5.1.4.135
D30h	BOOTCFG_EVENT_MUXCTL12	Event Mux Control Register 12	0262 0D30h	Section 5.1.4.136
D34h	BOOTCFG_EVENT_MUXCTL13	Event Mux Control Register 13	0262 0D34h	Section 5.1.4.137
E10h	BOOTCFG_DCAN_RAMINIT	DCAN RAM Initialization Register	0262 0E10h	Section 5.1.4.138
E20h	BOOTCFG_ETHERNET_CFG	Ethernet Configuration Register	0262 0E20h	Section 5.1.4.139
E30h	BOOTCFG_MLB_SIG_IO_CTL	MLB Signal IO Control Register	0262 0E30h	Section 5.1.4.140
E34h	BOOTCFG_MLB_DAT_IO_CTL	MLB Data IO Control Register	0262 0E34h	Section 5.1.4.141
E38h	BOOTCFG_MLB_CLK_IO_CTL	MLB Clock IO Control Register	0262 0E38h	Section 5.1.4.142
E40h	BOOTCFG_EPWM_CTL	ePWM Control Register	0262 0E40h	Section 5.1.4.143

Table 5-25. BOOT_CFG Registers (continued)

Offset	Acronym	Register Name	BOOT_CFG Physical Address	Section
E50h	BOOTCFG_ECAP_CAPEVT_CTL	eCAP Event Control Register	0262 0E50h	Section 5.1.4.144
E60h	BOOTCFG_EQEP_STAT	eQEP Status Register	0262 0E60h	Section 5.1.4.145
E70h	BOOTCFG_LVDS_BG_CTL	LVDS Bandgap Control Register	0262 0E70h	Section 5.1.4.146
E80h	BOOTCFG_LDO_USB_CTL	USB PHY LDO control register	0262 0E80h	Section 5.1.4.147
E84h	BOOTCFG_LDO_PCIE_CTL	PCIE PHY LDO control register	0262 0E84h	Section 5.1.4.148
1000h to 140Ch	BOOTCFG_PADCONFIG0 to BOOTCFG_PADCONFIG259	Pad Configuration Register 0 to 259	0262 1000h to 0262 140Ch	Section 5.1.4.149

5.1.4.1 BOOTCFG_REVISION Register (Offset = 0h) [reset = 4E842901h]

BOOTCFG_REVISION is shown in Figure 5-6 and described in Table 5-27.

BOOT_CFG Revision ID register.

Table 5-26. BOOTCFG_REVISION Instances

Instance	Physical Address
BOOT_CFG	0262 0000h

Figure 5-6. BOOTCFG_REVISION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOTCFG_REV																															
R-4E842901h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 5-27. BOOTCFG_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BOOTCFG_REV	R	4E842901h	TI internal data. Identifies revision of peripheral.

Table 5-28. Register Call Summary for BOOTCFG_REVISION

BOOT_CFG Functional Description • ID Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_REVISION Register (Offset = 0h) [reset = 4E842901h]: [0]

5.1.4.2 BOOTCFG_JTAGID Register (Offset = 18h) [reset = 8BB0602Fh]

BOOTCFG_JTAGID is shown in Figure 5-7 and described in Table 5-30.

Table 5-29. BOOTCFG_JTAGID Instances

Instance	Physical Address
BOOT_CFG	0262 0018h

Figure 5-7. BOOTCFG_JTAGID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VARIANT				PARTNUMBER											
R-8h				R-BB06h											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PARTNUMBER				MANUFACTURER											LSB
R-BB06h				R-17h											R-1h

LEGEND: R = Read Only; *-n* = value after reset

Table 5-30. BOOTCFG_JTAGID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	VARIANT	R	8h	Variant value 8h for 66AK2G1x devices
27-12	PARTNUMBER	R	BB06h	Part number for boundary scan
11-1	MANUFACTURER	R	17h	Indicates manufacture
0	LSB	R	1h	

Table 5-31. Register Call Summary for BOOTCFG_JTAGID

BOOT_CFG Functional Description • ID Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_JTAGID Register (Offset = 18h) [reset = -BB0602Fh]: [0]

5.1.4.3 BOOTCFG_DEVSTAT Register (Offset = 20h) [reset = 1h]

BOOTCFG_DEVSTAT is shown in Figure 5-8 and described in Table 5-33.

Indicates device bootstrap selection upon a power-on reset by PORn or RESETFULLn. The default value of this register is determined by the bootstrap pins. Once set, these bits remain set until a power-on reset.

Table 5-32. BOOTCFG_DEVSTAT Instances

Instance	Physical Address
BOOT_CFG	0262 0020h

Figure 5-8. BOOTCFG_DEVSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED			NODDR	MAINPLL_ODSEL	AVSIFSEL		BOOTMODE
R-			R-0h	R/W-0h	R/W-0h		R/W-0h
15	14	13	12	11	10	9	8
BOOTMODE							
R/W-0h							
7	6	5	4	3	2	1	0
BOOTMODE							LENDIAN
R/W-0h							R-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-33. BOOTCFG_DEVSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-21	RESERVED	R		
20	NODDR	R	0h	MSMC to EMIF control 0 - EMIF enabled, MSMC sends external traffic to DDR(normal mode) 1 - EMIF disabled, MSMC will re-direct DDR traffic toMSMC internal null end point
19	MAINPLL_ODSEL	R/W	0h	Main PLL OUTPUT_DIVIDE (OD) selection for the Boot ROM code. Boot ROM code programs the SECCTL register (Main PLL secondary control inside the PLLCTL IP) based on this bootstrap pin 0 - OUTPUT_DIVIDE (OD) is set to 1 (i.e. divide-by-2) (Default) 1 - OUTPUT_DIVIDE (OD) is set to 0 (i.e. divide-by-1)
18-17	AVSIFSEL	R/W	0h	Reserved
16-1	BOOTMODE	R/W	0h	Boot mode selection For more information about the booting procedure, see Section 4.3.2 in Chapter 4 *Initialization*.
0	LENDIAN	R	1h	Device Endian Mode. Applies to the entire device. 1 - Little Endian, permanently tied high

Table 5-34. Register Call Summary for BOOTCFG_DEVSTAT

BOOT_CFG Functional Description
- Registers Associated With Some Device External Signals: [0]
- BOOTCFG_DEVSTAT Register: [0][1][3][4]

Table 5-34. Register Call Summary for BOOTCFG_DEVSTAT (continued)

BOOT_CFG Registers
• BOOT_CFG Registers: [0] • BOOTCFG_DEVSTAT Register (Offset = 20h) [reset = 1h]: [0]

5.1.4.4 BOOTCFG_KICK0 Register (Offset = 38h) [reset = 0h]

BOOTCFG_KICK0 is shown in Figure 5-9 and described in Table 5-36.

The BOOT_CFG module contains a kicker mechanism to prevent spurious writes from changing any of the BOOT_CFG MMR values. When the kicker is locked (which it is initially after power-on reset), none of the BOOT_CFG MMRs are writable (they are only readable). This mechanism requires an MMR write to each of the BOOTCFG_KICK0 and BOOTCFG_KICK1 registers with exact data values before the kicker lock mechanism is unlocked. Once released, all the BOOT_CFG MMRs having write permissions are writable (the read only MMRs are still read only). The BOOTCFG_KICK0 data is 83E70B13h. The BOOTCFG_KICK1 data is 95A4F1E0h. Writing any other data value to either of these kick MMRs locks the kicker mechanism and blocks writes to BOOT_CFG MMRs. To ensure protection to all BOOT_CFG MMRs, software must always re-lock the kicker mechanism after completing the MMR writes.

Table 5-35. BOOTCFG_KICK0 Instances

Instance	Physical Address
BOOT_CFG	0262 0038h

Figure 5-9. BOOTCFG_KICK0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KICK0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-36. BOOTCFG_KICK0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KICK0	R/W	0h	

Table 5-37. Register Call Summary for BOOTCFG_KICK0

BOOT_CFG Functional Description • Kick Protection Registers: [0][1]
BOOT_CFG Registers • BOOTCFG_KICK0 Register (Offset = 38h) [reset = 0h]: [0][1][2] • BOOT_CFG Registers: [0]

5.1.4.5 BOOTCFG_KICK1 Register (Offset = 3Ch) [reset = 0h]

BOOTCFG_KICK1 is shown in Figure 5-10 and described in Table 5-39.

BOOT_CFG BOOTCFG_KICK1 lock register.

Table 5-38. BOOTCFG_KICK1 Instances

Instance	Physical Address
BOOT_CFG	0262 003Ch

Figure 5-10. BOOTCFG_KICK1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KICK1																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-39. BOOTCFG_KICK1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KICK1	R/W	0h	

Table 5-40. Register Call Summary for BOOTCFG_KICK1

BOOT_CFG Functional Description • Kick Protection Registers: [0][1]
BOOT_CFG Registers • BOOTCFG_KICK0 Register (Offset = 38h) [reset = 0h]: [0][1] • BOOT_CFG Registers: [0] • BOOTCFG_KICK1 Register (Offset = 3Ch) [reset = 0h]: [0][1]

5.1.4.6 BOOTCFG_DSP_BOOT_ADDR0 Register (Offset = 40h) [reset = 20B00000h]

BOOTCFG_DSP_BOOT_ADDR0 is shown in Figure 5-11 and described in Table 5-42.

Boot address register for C66x core. Bit 0 of this register is a don't care and must be masked by software.

Table 5-41. BOOTCFG_DSP_BOOT_ADDR0 Instances

Instance	Physical Address
BOOT_CFG	0262 0040h

Figure 5-11. BOOTCFG_DSP_BOOT_ADDR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ISTP_RST_VAL																						RESERVED									
R/W-082C00h																						R-									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-42. BOOTCFG_DSP_BOOT_ADDR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	ISTP_RST_VAL	R/W	082C00h	DSP istp_rst_val[21:0] port value. On secure device this field is read-only by Secure Master. Emulator cannot access this in Secure mode.
9-0	RESERVED	R		Reserved

Table 5-43. Register Call Summary for BOOTCFG_DSP_BOOT_ADDR0

BOOT_CFG Functional Description • DSP Boot Address Registers: [0]
BOOT_CFG Registers • BOOTCFG_DSP_BOOT_ADDR0 Register (Offset = 40h) [reset = 20B00000h]: [0] • BOOT_CFG Registers: [0]

5.1.4.7 BOOTCFG_INTR_RAW_STAT_SET Register (Offset = E0h) [reset = -h]

BOOTCFG_INTR_RAW_STAT_SET is shown in Figure 5-12 and described in Table 5-45.

The Interrupt Raw Status/Set register shows the interrupt status (before enabling) and allows setting the interrupt status.

Table 5-44. BOOTCFG_INTR_RAW_STAT_SET Instances

Instance	Physical Address
BOOT_CFG	0262 00E0h

Figure 5-12. BOOTCFG_INTR_RAW_STAT_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED						ADDR_ERR	PROT_ERR
R-						W1toS-0h	W1toS-0h

LEGEND: R = Read Only; W1toS = Write 1 to Set Bit; -*n* = value after reset

Table 5-45. BOOTCFG_INTR_RAW_STAT_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R		Reserved
1	ADDR_ERR	W1toS	0h	Addressing violation error. Raw status is read. Write a 1 to set the status. Writing a 0 has no effect.
0	PROT_ERR	W1toS	0h	Protection violation error. Raw status is read. Write a 1 to set the status. Writing a 0 has no effect.

Table 5-46. Register Call Summary for BOOTCFG_INTR_RAW_STAT_SET

BOOT_CFG Functional Description • BOOT_CFG Module Interrupts: [0][1][2][3][4][5][6][7]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_INTR_RAW_STAT_SET Register (Offset = E0h) [reset = -h]: [0]

5.1.4.8 BOOTCFG_INTR_ENABLED_STAT_CLR Register (Offset = E4h) [reset = -h]

BOOTCFG_INTR_ENABLED_STAT_CLR is shown in Figure 5-13 and described in Table 5-48.

The Interrupt Enabled Status/Clear register shows the interrupt enabled status and allows clearing of the interruptstatus.

Table 5-47. BOOTCFG_INTR_ENABLED_STAT_CLR Instances

Instance	Physical Address
BOOT_CFG	0262 00E4h

Figure 5-13. BOOTCFG_INTR_ENABLED_STAT_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED						ENABLED_ADDR_ERR	ENABLED_PROT_ERR
R-						W1toCl-0h	W1toCl-0h

LEGEND: R = Read Only; W1toCl = Write 1 to Clear Bit; -*n* = value after reset

Table 5-48. BOOTCFG_INTR_ENABLED_STAT_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R		Reserved
1	ENABLED_ADDR_ERR	W1toCl	0h	Addressing violation error. Enabled status is read. Write a 1 to clear the status. Writing a 0 has no effect.
0	ENABLED_PROT_ERR	W1toCl	0h	Protection violation error. Enabled status is read. Write a 1 to clear the status. Writing a 0 has no effect.

Table 5-49. Register Call Summary for BOOTCFG_INTR_ENABLED_STAT_CLR

BOOT_CFG Functional Description • BOOT_CFG Module Interrupts: [0][1][2][3][4][5]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_INTR_ENABLED_STAT_CLR Register (Offset = E4h) [reset = -h]: [0]

5.1.4.9 BOOTCFG_INTR_ENABLE Register (Offset = E8h) [reset = -h]

BOOTCFG_INTR_ENABLE is shown in Figure 5-14 and described in Table 5-51.

The Interrupt Enable register shows the interrupt enable value and allows setting the enable.

Table 5-50. BOOTCFG_INTR_ENABLE Instances

Instance	Physical Address
BOOT_CFG	0262 00E8h

Figure 5-14. BOOTCFG_INTR_ENABLE Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED						ADDR_ERR_EN	PROT_ERR_EN
R-						W1toS-0h	W1toS-0h

LEGEND: R = Read Only; W1toS = Write 1 to Set Bit; -*n* = value after reset

Table 5-51. BOOTCFG_INTR_ENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R		Reserved
1	ADDR_ERR_EN	W1toS	0h	Addressing violation error enable. Write a 1 to set the enable. Writing a 0 has no effect.
0	PROT_ERR_EN	W1toS	0h	Protection violation error enable. Write a 1 to set the enable. Writing a 0 has no effect.

Table 5-52. Register Call Summary for BOOTCFG_INTR_ENABLE

BOOT_CFG Functional Description • BOOT_CFG Module Interrupts: [0][1][2][3][4]
BOOT_CFG Registers • BOOTCFG_INTR_ENABLE Register (Offset = E8h) [reset = -h]: [0] • BOOT_CFG Registers: [0]

5.1.4.10 BOOTCFG_INTR_ENABLE_CLR Register (Offset = ECh) [reset = -h]

BOOTCFG_INTR_ENABLE_CLR is shown in Figure 5-15 and described in Table 5-54.

The Interrupt Enable Clear register shows the interrupt enable and allows clearing of the interrupt enable.

Table 5-53. BOOTCFG_INTR_ENABLE_CLR Instances

Instance	Physical Address
BOOT_CFG	0262 00ECh

Figure 5-15. BOOTCFG_INTR_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED						ADDR_ERR_EN_CLR	PROT_ERR_EN_CLR
R-						W1toCl-0h	W1toCl-0h

LEGEND: R = Read Only; W1toCl = Write 1 to Clear Bit; -*n* = value after reset

Table 5-54. BOOTCFG_INTR_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R		Reserved
1	ADDR_ERR_EN_CLR	W1toCl	0h	Addressing violation error enable. Write a 1 to clear the enable. Writing a 0 has no effect.
0	PROT_ERR_EN_CLR	W1toCl	0h	Protection violation error enable. Write a 1 to clear the enable. Writing a 0 has no effect.

Table 5-55. Register Call Summary for BOOTCFG_INTR_ENABLE_CLR

BOOT_CFG Functional Description • BOOT_CFG Module Interrupts: [0][1][2][3]
BOOT_CFG Registers • BOOTCFG_INTR_ENABLE_CLR Register (Offset = ECh) [reset = -h]: [0] • BOOT_CFG Registers: [0]

5.1.4.11 BOOTCFG_EOI Register (Offset = F0h) [reset = -h]

BOOTCFG_EOI is shown in Figure 5-16 and described in Table 5-57.

The EOI register allows software to indicate when the end of interrupt service is complete. The EOI vector value is dependent on the interrupt handling.

Table 5-56. BOOTCFG_EOI Instances

Instance	Physical Address
BOOT_CFG	0262 00F0h

Figure 5-16. BOOTCFG_EOI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								EOI_VECTOR							
R-																								R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-57. BOOTCFG_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R		Reserved
7-0	EOI_VECTOR	R/W	0h	EOI vector value.

Table 5-58. Register Call Summary for BOOTCFG_EOI

BOOT_CFG Registers
- BOOT_CFG Registers: [0]
- BOOTCFG_EOI Register (Offset = F0h) [reset = -h]: [0]

5.1.4.12 BOOTCFG_FAULT_ADDR Register (Offset = F4h) [reset = -h]

BOOTCFG_FAULT_ADDR is shown in Figure 5-17 and described in Table 5-60.

This register holds the address of the first fault transfer.

Table 5-59. BOOTCFG_FAULT_ADDR Instances

Instance	Physical Address
BOOT_CFG	0262 00F4h

Figure 5-17. BOOTCFG_FAULT_ADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FAULT_ADDR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 5-60. BOOTCFG_FAULT_ADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FAULT_ADDR	R	0h	Fault address.

Table 5-61. Register Call Summary for BOOTCFG_FAULT_ADDR

BOOT_CFG Functional Description • BOOT_CFG Module Interrupts: [0][1]
BOOT_CFG Registers • BOOTCFG_FAULT_ADDR Register (Offset = F4h) [reset = -h]: [0] • BOOT_CFG Registers: [0]

5.1.4.13 BOOTCFG_FAULT_STAT Register (Offset = F8h) [reset = -h]

BOOTCFG_FAULT_STAT is shown in Figure 5-18 and described in Table 5-63.

This register holds the status and attributes of the first fault transfer.

Table 5-62. BOOTCFG_FAULT_STAT Instances

Instance	Physical Address
BOOT_CFG	0262 00F8h

Figure 5-18. BOOTCFG_FAULT_STAT Register

31	30	29	28	27	26	25	24
BOOTCFG_TXNID							
R-0h							
23	22	21	20	19	18	17	16
BOOTCFG_MSTID							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			BOOTCFG_PRIVID				RESERVED
R-			R-0h				R-
7	6	5	4	3	2	1	0
RESERVED			FAULT_TYPE				
R-			R-0h				

LEGEND: R = Read Only; *-n* = value after reset

Table 5-63. BOOTCFG_FAULT_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	BOOTCFG_TXNID	R	0h	Transaction ID associated with the fault access
23-16	BOOTCFG_MSTID	R	0h	Master ID associated with the fault access
15-13	RESERVED	R		Reserved
12-9	BOOTCFG_PRIVID	R	0h	Privilege ID associated with the fault access
8-6	RESERVED	R		Reserved
5-0	FAULT_TYPE	R	0h	Fault type. 000000b = no fault 000001b = user execute fault 000010b = user write fault 000100b = user read fault 001000b = supervisor execute fault 010000b = supervisor write fault 100000b = supervisor read fault

Table 5-64. Register Call Summary for BOOTCFG_FAULT_STAT

BOOT_CFG Functional Description • BOOT_CFG Module Interrupts: [0][1]
BOOT_CFG Registers • BOOTCFG_FAULT_STAT Register (Offset = F8h) [reset = -h]: [0] • BOOT_CFG Registers: [0]

5.1.4.14 BOOTCFG_FAULT_CLR Register (Offset = FCh) [reset = -h]

BOOTCFG_FAULT_CLR is shown in Figure 5-19 and described in Table 5-66.

This register allows the software to clear the current fault so that another can be captured when this register is written.

Table 5-65. BOOTCFG_FAULT_CLR Instances

Instance	Physical Address
BOOT_CFG	0262 00FCh

Figure 5-19. BOOTCFG_FAULT_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED							FAULT_CLR
R-							W1toCl-0h

LEGEND: R = Read Only; W1toCl = Write 1 to Clear Bit; -*n* = value after reset

Table 5-66. BOOTCFG_FAULT_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R		Reserved
0	FAULT_CLR	W1toCl	0h	Fault clear. Writing a 1 clears the current fault. Writing a 0 has no effect.

Table 5-67. Register Call Summary for BOOTCFG_FAULT_CLR

BOOT_CFG Functional Description • BOOT_CFG Module Interrupts: [0]
BOOT_CFG Registers • BOOTCFG_FAULT_CLR Register (Offset = FCh) [reset = -h]: [0] • BOOT_CFG Registers: [0]

5.1.4.15 BOOTCFG_MACID0 Register (Offset = 110h) [reset = -h]

BOOTCFG_MACID0 is shown in Figure 5-20 and described in Table 5-69.

Each individual device has a 48-bit MAC address and consumes only one unique MAC address out of the range. There are two registers to hold these values, MACI_D0[31-0] (32 bits) and MAC_ID1[15-0] (16 bits).

Table 5-68. BOOTCFG_MACID0 Instances

Instance	Physical Address
BOOT_CFG	0262 0110h

Figure 5-20. BOOTCFG_MACID0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MACID																															
R-																															

LEGEND: R = Read Only; *-n* = value after reset

Table 5-69. BOOTCFG_MACID0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MACID	R		MAC ID, lower 32 bits

Table 5-70. Register Call Summary for BOOTCFG_MACID0

BOOT_CFG Functional Description • NSS MAC Address Registers: [0]
BOOT_CFG Registers • BOOTCFG_MACID0 Register (Offset = 110h) [reset = -h]: [0] • BOOT_CFG Registers: [0]

5.1.4.16 BOOTCFG_MACID1 Register (Offset = 114h) [reset = -h]

BOOTCFG_MACID1 is shown in Figure 5-21 and described in Table 5-72.

Each individual device has a 48-bit MAC address and consumes only one unique MAC address out of the range. There are two registers to hold these values, MAC_ID0[31-0] (32 bits) and MAC_ID1[15-0] (16 bits).

Table 5-71. BOOTCFG_MACID1 Instances

Instance	Physical Address
BOOT_CFG	0262 0114h

Figure 5-21. BOOTCFG_MACID1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						FLOW	BCAST
R-0h						R-0h	R-0h
15	14	13	12	11	10	9	8
MACID							
R-							
7	6	5	4	3	2	1	0
MACID							
R-							

LEGEND: R = Read Only; *-n* = value after reset

Table 5-72. BOOTCFG_MACID1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17	FLOW	R	0h	MAC Flow Control 0 - Off 1 - On
16	BCAST	R	0h	Default m/b-cast reception 0 - Broadcast 1 - Disabled
15-0	MACID	R		MAC ID, upper 16 bits

Table 5-73. Register Call Summary for BOOTCFG_MACID1

BOOT_CFG Functional Description • NSS MAC Address Registers: [0]
BOOT_CFG Registers • BOOTCFG_MACID1 Register (Offset = 114h) [reset = -h]: [0] • BOOT_CFG Registers: [0]

5.1.4.17 BOOTCFG_PCIEVENDORID Register (Offset = 128h) [reset = B00B104Ch]

BOOTCFG_PCIEVENDORID is shown in Figure 5-22 and described in Table 5-75.

PCIe device ID and vendor ID register.

Table 5-74. BOOTCFG_PCIEVENDORID Instances

Instance	Physical Address
BOOT_CFG	0262 0128h

Figure 5-22. BOOTCFG_PCIEVENDORID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCIEDEVICEID																PCIEVENDORID															
R/W-B00Bh																R/W-104Ch															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-75. BOOTCFG_PCIEVENDORID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	PCIEDEVICEID	R/W	B00Bh	PCIe Device ID
15-0	PCIEVENDORID	R/W	104Ch	TI Vendor ID

Table 5-76. Register Call Summary for BOOTCFG_PCIEVENDORID

BOOT_CFG Functional Description • ID Registers: [0]
BOOT_CFG Registers • BOOTCFG_PCIEVENDORID Register (Offset = 128h) [reset = B00B104Ch]: [0] • BOOT_CFG Registers: [0]

5.1.4.18 BOOTCFG_LRSTNMISTAT_CLR Register (Offset = 130h) [reset = 0h]

BOOTCFG_LRSTNMISTAT_CLR is shown in Figure 5-23 and described in Table 5-78.

Local reset and NMI pin status clear register for C66x core. The LRSTNMISTAT bits can be cleared by writing 1 to the corresponding bit in the BOOTCFG_LRSTNMISTAT_CLR register.

Table 5-77. BOOTCFG_LRSTNMISTAT_CLR Instances

Instance	Physical Address
BOOT_CFG	0262 0130h

Figure 5-23. BOOTCFG_LRSTNMISTAT_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							NMI_STAT_0_CLR
R-							R/WtoPH-0h
7	6	5	4	3	2	1	0
RESERVED							LRESET_STAT_0_CLR
R-							R/WtoPH-0h

LEGEND: R = Read Only; R/WtoPH = Read/Write to Pulse High; *-n* = value after reset

Table 5-78. BOOTCFG_LRSTNMISTAT_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R		Reserved
8	NMI_STAT_0_CLR	R/WtoPH	0h	C66x core 0 NMI pin clear Writes: 0 - No effect 1 - Clears the NMI_STAT0 bit in the BOOTCFG_LRSTNMISTAT register
7-1	RESERVED	R		Reserved
0	LRESET_STAT_0_CLR	R/WtoPH	0h	C66x core 0 local reset pin clear Writes: 0 - No effect 1 - Clears the LRESET_STAT0 bit in theLRSTNMISTAT register

Table 5-79. Register Call Summary for BOOTCFG_LRSTNMISTAT_CLR

BOOT_CFG Functional Description • Registers Associated With Some Device External Signals: [0] • LRSTNMISTAT And BOOTCFG_LRSTNMISTAT_CLR Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_LRSTNMISTAT_CLR Register (Offset = 130h) [reset = 0h]: [0][1]

5.1.4.19 BOOTCFG_RESET_STAT_CLR Register (Offset = 134h) [reset = 0h]

BOOTCFG_RESET_STAT_CLR is shown in Figure 5-24 and described in Table 5-81.

The RESET_STAT bits can be cleared by writing 1 to the corresponding bit in the BOOTCFG_RESET_STAT_CLR register.

Table 5-80. BOOTCFG_RESET_STAT_CLR Instances

Instance	Physical Address
BOOT_CFG	0262 0134h

Figure 5-24. BOOTCFG_RESET_STAT_CLR Register

31	30	29	28	27	26	25	24
GRST_STAT_CLR	RESERVED						
R/WtoPH-0h	R-						
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED							LRST_STAT_0_CLR
R-							R/WtoPH-0h

LEGEND: R = Read Only; R/WtoPH = Read/Write to Pulse High; *-n* = value after reset

Table 5-81. BOOTCFG_RESET_STAT_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GRST_STAT_CLR	R/WtoPH	0h	Global reset clear Writes: 0 - No effect 1 - Clears the GRST_STAT bit in the RESET_STATregister
30-1	RESERVED	R		Reserved
0	LRST_STAT_0_CLR	R/WtoPH	0h	C66x core 0 local reset clear Writes: 0 - No effect 1 - Clears the LRST_STAT0 bit in the RESET_STATregister

Table 5-82. Register Call Summary for BOOTCFG_RESET_STAT_CLR

BOOT_CFG Functional Description • Device Reset Status Registers: [0]
BOOT_CFG Registers • BOOTCFG_RESET_STAT_CLR Register (Offset = 134h) [reset = 0h]: [0][1] • BOOT_CFG Registers: [0]

5.1.4.20 BOOTCFG_BOOT_COMPLETE Register (Offset = 13Ch) [reset = 0h]

BOOTCFG_BOOT_COMPLETE is shown in Figure 5-25 and described in Table 5-84.

Register to control BOOTCOMPLETE pin status to indicate completion of the ROM booting process. The BCx bit indicates the boot complete status of the corresponding C66x or Arm cores. All BCx bits are sticky bits - that is, they can be set only once by the software after device reset and they will be cleared to 0 on all device resets. Boot ROM code is implemented such that each C66x and Arm cores sets its corresponding BCx bit immediately before branching to the predefined location in memory. When all the BCx bits are set to 1, the BOOTCOMPLETE pin status will be high indicating that device ROM booting process is complete.

Table 5-83. BOOTCFG_BOOT_COMPLETE Instances

Instance	Physical Address
BOOT_CFG	0262 013Ch

Figure 5-25. BOOTCFG_BOOT_COMPLETE Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							ARM0_COMPLETE
R-							R/W1toS-0h
7	6	5	4	3	2	1	0
RESERVED							DSP0_COMPLETE
R-							R/W1toS-0h

LEGEND: R = Read Only; R/W1toS = Read/Write 1 to Set Bit; -*n* = value after reset

Table 5-84. BOOTCFG_BOOT_COMPLETE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R		Reserved
8	ARM0_COMPLETE	R/W1toS	0h	A15 core 0 ROM boot status 0 - ROM boot not complete 1 - ROM boot complete
7-1	RESERVED	R		Reserved
0	DSP0_COMPLETE	R/W1toS	0h	C66x core 0 ROM boot status 0 - ROM boot not complete 1 - ROM boot complete

Table 5-85. Register Call Summary for BOOTCFG_BOOT_COMPLETE

BOOT_CFG Functional Description • BOOTCFG_BOOT_COMPLETE Register: [0] • Registers Associated With Some Device External Signals: [0]
BOOT_CFG Registers • BOOTCFG_BOOT_COMPLETE Register (Offset = 13Ch) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.21 BOOTCFG_RESET_STAT Register (Offset = 144h) [reset = 0h]

BOOTCFG_RESET_STAT is shown in Figure 5-26 and described in Table 5-87.

The Reset Status Register captures the status of local reset (LRx) for each of the C66x cores and also the global device reset (GR). Software can use this information to take different device initialization steps.

- In case of local reset: The LRx bits are written as 1 and the GR bit is written as 0 only when the C66x core receives a local reset without receiving a global reset.

- In case of global reset: The LRx bits are written as 0 and the GR bit is written as 1 only when a global reset is asserted.

Table 5-86. BOOTCFG_RESET_STAT Instances

Instance	Physical Address
BOOT_CFG	0262 0144h

Figure 5-26. BOOTCFG_RESET_STAT Register

31	30	29	28	27	26	25	24
GRST_STAT	RESERVED						
R-0h	R-						
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED							LRST_STAT0
R-							R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 5-87. BOOTCFG_RESET_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GRST_STAT	R	0h	Global reset status 0 - Device has not received a global reset 1 - Device received a global reset
30-1	RESERVED	R		Reserved
0	LRST_STAT0	R	0h	C66x core 0 local reset status 0 - Core has not received a local reset 1 - Core received a local reset

Table 5-88. Register Call Summary for BOOTCFG_RESET_STAT

BOOT_CFG Functional Description • Device Reset Status Registers: [0][1][2]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_RESET_STAT Register (Offset = 144h) [reset = 0h]: [0]

5.1.4.22 BOOTCFG_LRSTNMISTAT Register (Offset = 148h) [reset = 0h]

BOOTCFG_LRSTNMISTAT is shown in Figure 5-27 and described in Table 5-90.

Local reset and NMI pin status register for C66x core. BOOTCFG_LRSTNMISTAT is used to latch the status of LRESETz and NMIz based on LRESETNMIENz pin. The pin status is latched on the rising edge of LRESETNMIENz. Note that LRESETz and NMIz are available only for the C66x core and not the Arm core.

Table 5-89. BOOTCFG_LRSTNMISTAT Instances

Instance	Physical Address
BOOT_CFG	0262 0148h

Figure 5-27. BOOTCFG_LRSTNMISTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							NMI_STAT
R-							R-0h
7	6	5	4	3	2	1	0
RESERVED							LRESET_STAT
R-							R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 5-90. BOOTCFG_LRSTNMISTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R		Reserved
8	NMI_STAT	R	0h	C66x core 0 NMI pin status 0 - NMIz pin not asserted 1 - NMIz pin asserted
7-1	RESERVED	R		Reserved
0	LRESET_STAT	R	0h	C66x core 0 local reset pin status 0 - LRESETz pin not asserted 1 - LRESETz pin asserted

Table 5-91. Register Call Summary for BOOTCFG_LRSTNMISTAT

BOOT_CFG Functional Description • Registers Associated With Some Device External Signals: [0][1]
BOOT_CFG Registers • BOOTCFG_LRSTNMISTAT Register (Offset = 148h) [reset = 0h]: [0][1] • BOOT_CFG Registers: [0] • BOOTCFG_LRSTNMISTAT_CLR Register (Offset = 130h) [reset = 0h]: [0]

5.1.4.23 BOOTCFG_DEVCFG Register (Offset = 14Ch) [reset = 1h]

BOOTCFG_DEVCFG is shown in Figure 5-28 and described in Table 5-93.

Device configuration register.

Table 5-92. BOOTCFG_DEVCFG Instances

Instance	Physical Address
BOOT_CFG	0262 014Ch

Figure 5-28. BOOTCFG_DEVCFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED					PCIE_DEV_TYPE		SYSCLKOUTEN
R-					R/W-0h		R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-93. BOOTCFG_DEVCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R		Reserved
2-1	PCIE_DEV_TYPE	R/W	0h	PCIE device type Field values (Others are reserved): 0h = Endpoint mode 1h = Legacy Endpoint mode 2h = Rootcomplex mode
0	SYSCLKOUTEN	R/W	1h	SYSCLKOUT enable 0 - No clock output 1 - Clock output enabled

Table 5-94. Register Call Summary for BOOTCFG_DEVCFG

BOOT_CFG Registers
- BOOTCFG_DEVCFG Register (Offset = 14Ch) [reset = 1h]: [0]
- BOOT_CFG Registers: [0]

5.1.4.24 BOOTCFG_PWR_STATE Register (Offset = 150h) [reset = 0h]

BOOTCFG_PWR_STATE is shown in Figure 5-29 and described in Table 5-96.

Register is controlled by the software to indicate the device power-saving mode. This register is only accessible by secure master or secure guest on a secure device, emulation access is not allowed.

Table 5-95. BOOTCFG_PWR_STATE Instances

Instance	Physical Address
BOOT_CFG	0262 0150h

Figure 5-29. BOOTCFG_PWR_STATE Register

31	30	29	28	27	26	25	24
PWR_STATE_GENERAL							
R/W-0h							
23	22	21	20	19	18	17	16
PWR_STATE_GENERAL							
R/W-0h							
15	14	13	12	11	10	9	8
PWR_STATE_GENERAL							
R/W-0h							
7	6	5	4	3	2	1	0
PWR_STATE_GENERAL				PMMC_FW_LOAD	PWR_MODE		
R/W-0h				R/W-0h	R/W-0h		

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-96. BOOTCFG_PWR_STATE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	PWR_STATE_GENERAL	R/W	0h	General purpose register, allocated for software usage
3	PMMC_FW_LOAD	R/W	0h	Indicates whether PMMC firmware is loaded by host processor (A15 or DSP) or not 0 - PMMC firmware not loaded 1 - PMMC firmware is loaded
2-0	PWR_MODE	R/W	0h	Indicates device power mode Field values (Others are reserved): 000b - Run mode

Table 5-97. Register Call Summary for BOOTCFG_PWR_STATE

BOOT_CFG Registers
- BOOTCFG_PWR_STATE Register (Offset = 150h) [reset = 0h]: [0]
- BOOT_CFG Registers: [0]

5.1.4.25 BOOTCFG_INITIATOR_PRIORITY0 Register (Offset = 154h) [reset = 77777777h]

BOOTCFG_INITIATOR_PRIORITY0 is shown in Figure 5-30 and described in Table 5-99.

This register configures the trasaction priority of the initiator (master port) to control arbitration within the interconnect. Valid settings are 7h (lowest priority) to 0h (urgent priority) for each bitfield.

Table 5-98. BOOTCFG_INITIATOR_PRIORITY0 Instances

Instance	Physical Address
BOOT_CFG	0262 0154h

Figure 5-30. BOOTCFG_INITIATOR_PRIORITY0 Register

31	30	29	28	27	26	25	24
RESERVED	ICSS1_PRU1_PRI			RESERVED	ICSS1_PRU0_PRI		
R-0h	R/W-7h			R-0h	R/W-7h		
23	22	21	20	19	18	17	16
RESERVED	ICSS0_PRU1_PRI			RESERVED	ICSS0_PRU0_PRI		
R-0h	R/W-7h			R-0h	R/W-7h		
15	14	13	12	11	10	9	8
RESERVED	MMC1_PRI			RESERVED	MMC0_PRI		
R-0h	R/W-7h			R-0h	R/W-7h		
7	6	5	4	3	2	1	0
RESERVED	MLB_PRI			RESERVED	PMMC_PRI		
R-0h	R/W-7h			R-0h	R/W-7h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-99. BOOTCFG_INITIATOR_PRIORITY0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30-28	ICSS1_PRU1_PRI	R/W	7h	ICSS1 PRU1 master port priority
27	RESERVED	R	0h	Reserved
26-24	ICSS1_PRU0_PRI	R/W	7h	ICSS1 PRU0 master port priority
23	RESERVED	R	0h	Reserved
22-20	ICSS0_PRU1_PRI	R/W	7h	ICSS0 PRU1 master port priority
19	RESERVED	R	0h	Reserved
18-16	ICSS0_PRU0_PRI	R/W	7h	ICSS0 PRU0 master port priority
15	RESERVED	R	0h	Reserved
14-12	MMC1_PRI	R/W	7h	MMC1 master port priority
11	RESERVED	R	0h	Reserved
10-8	MMC0_PRI	R/W	7h	MMC0 master port priority
7	RESERVED	R	0h	Reserved
6-4	MLB_PRI	R/W	7h	MLB master port priority
3	RESERVED	R	0h	Reserved
2-0	PMMC_PRI	R/W	7h	PMMC master port priority

Table 5-100. Register Call Summary for BOOTCFG_INITIATOR_PRIORITY0

BOOT_CFG Functional Description • TeraNet Priority Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_INITIATOR_PRIORITY0 Register (Offset = 154h) [reset = 77777777h]: [0]

5.1.4.26 BOOTCFG_INITIATOR_PRIORITY1 Register (Offset = 158h) [reset = 17h]

BOOTCFG_INITIATOR_PRIORITY1 is shown in Figure 5-31 and described in Table 5-102.

This register configures the trasaction priority of the initiator (master port) to control arbitration within the interconnect. Valid settings are 7h (lowest priority) to 0h (urgent priority) for each bitfield.

Table 5-101. BOOTCFG_INITIATOR_PRIORITY1 Instances

Instance	Physical Address
BOOT_CFG	0262 0158h

Figure 5-31. BOOTCFG_INITIATOR_PRIORITY1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	DSS_PRI_HI			RESERVED	DSS_PRI_LO		
R-0h	R/W-1h			R-0h	R/W-7h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-102. BOOTCFG_INITIATOR_PRIORITY1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6-4	DSS_PRI_HI	R/W	1h	DSS master port priority for high priority access (when MFLAG = 1)
3	RESERVED	R	0h	Reserved
2-0	DSS_PRI_LO	R/W	7h	DSS master port priority for low priority access (when MFLAG = 0)

Table 5-103. Register Call Summary for BOOTCFG_INITIATOR_PRIORITY1

BOOT_CFG Functional Description • TeraNet Priority Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_INITIATOR_PRIORITY1 Register (Offset = 158h) [reset = 17h]: [0]

5.1.4.27 BOOTCFG_NMIGR0 Register (Offset = 200h) [reset = 0h]

BOOTCFG_NMIGR0 is shown in Figure 5-32 and described in Table 5-105.

BOOTCFG_NMIGR0 register generates NMI event to C66x core. Writing a 1 to the NMIG field generates an NMI pulse. Writing a 0 has no effect and reads return 0 and have no other effect.

Table 5-104. BOOTCFG_NMIGR0 Instances

Instance	Physical Address
BOOT_CFG	0262 0200h

Figure 5-32. BOOTCFG_NMIGR0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED							NMIGR0_REG
R-							R/WtoPH-0h

LEGEND: R = Read Only; R/WtoPH = Read/Write to Pulse High; *-n* = value after reset

Table 5-105. BOOTCFG_NMIGR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R		Reserved
0	NMIGR0_REG	R/WtoPH	0h	Reads return 0 Writes: 0 - No effect 1 - Creates NMI pulse to C66x core 0

Table 5-106. Register Call Summary for BOOTCFG_NMIGR0

BOOT_CFG Registers
- BOOTCFG_NMIGR0 Register (Offset = 200h) [reset = 0h]: [0][1]
- BOOT_CFG Registers: [0]

5.1.4.28 BOOTCFG_IPCGR0 Register (Offset = 240h) [reset = 0h]

BOOTCFG_IPCGR0 is shown in Figure 5-33 and described in Table 5-108.

Register facilitate inter-core interrupt to C66x core 0. A write of 1 to the IPCG field of the BOOTCFG_IPCGR0 register generates an interrupt pulse to C66x core 0. The register also provides a Source ID facility identifying up to 28 different sources of interrupt. Allocation of source bits to source processor and meaning is entirely based on software convention. There can be numerous sources for the register as this is completely controlled by software. Any master that has access to BOOT_CFG module space can write to this register. Register is KICK un-locked, that means software need not go through KICK sequence in order to write to this register.

Table 5-107. BOOTCFG_IPCGR0 Instances

Instance	Physical Address
BOOT_CFG	0262 0240h

Figure 5-33. BOOTCFG_IPCGR0 Register

31	30	29	28	27	26	25	24
IPCGR0_SRC							
R/W1toS-0h							
23	22	21	20	19	18	17	16
IPCGR0_SRC							
R/W1toS-0h							
15	14	13	12	11	10	9	8
IPCGR0_SRC							
R/W1toS-0h							
7	6	5	4	3	2	1	0
IPCGR0_SRC				RESERVED			IPCGR0_REG
R/W1toS-0h				R-			R/WtoPH-0h

LEGEND: R = Read Only; R/W1toS = Read/Write 1 to Set Bit; R/WtoPH = Read/Write to Pulse High; -*n* = value after reset

Table 5-108. BOOTCFG_IPCGR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	IPCGR0_SRC	R/W1toS	0h	Reads return current value of internal register bit Writes: 0 - No effect 1 - Sets both SRCx and the corresponding SRC_CLRx bitin BOOTCFG_IPCAR0 register
3-1	RESERVED	R		Reserved
0	IPCGR0_REG	R/WtoPH	0h	Reads return 0 Writes: 0 - No effect 1 - Creates an inter-core interrupt

Table 5-109. Register Call Summary for BOOTCFG_IPCGR0

BOOT_CFG Functional Description • Inter-processor Communication Registers: [0][1]
Control Module (BOOT_CFG) • BOOT_CFG Integration: [0]

Table 5-109. Register Call Summary for BOOTCFG_IPCGR0 (continued)

BOOT_CFG Registers
• BOOTCFG_IPCGR0 Register (Offset = 240h) [reset = 0h]: [0][1] • BOOT_CFG Registers: [0] • BOOTCFG_IPCAR0 Register (Offset = 280h) [reset = 0h]: [0]

5.1.4.29 BOOTCFG_IPCGR8 Register (Offset = 260h) [reset = 0h]

BOOTCFG_IPCGR8 is shown in Figure 5-34 and described in Table 5-111.

Register facilitate inter-core interrupt to A15 core 0. A write of 1 to the IPCG field of the BOOTCFG_IPCGR8 register generates an interrupt pulse to the corresponding core. The register also provides a Source ID facility identifying up to 28 different sources of interrupt. Allocation of source bits to source processor and meaning is entirely based on software convention. There can be numerous sources for the register as this is completely controlled by software. Any master that has access to BOOT_CFG module space can write to this register. Register is KICK un-locked, that means software need not go through KICK sequence in order to write to this register.

Table 5-110. BOOTCFG_IPCGR8 Instances

Instance	Physical Address
BOOT_CFG	0262 0260h

Figure 5-34. BOOTCFG_IPCGR8 Register

31	30	29	28	27	26	25	24
IPCGR8_SRC							
R/W1toS-0h							
23	22	21	20	19	18	17	16
IPCGR8_SRC							
R/W1toS-0h							
15	14	13	12	11	10	9	8
IPCGR8_SRC							
R/W1toS-0h							
7	6	5	4	3	2	1	0
IPCGR8_SRC				RESERVED			IPCGR8_REG
R/W1toS-0h				R-			R/WtoPH-0h

LEGEND: R = Read Only; R/W1toS = Read/Write 1 to Set Bit; R/WtoPH = Read/Write to Pulse High; -*n* = value after reset

Table 5-111. BOOTCFG_IPCGR8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	IPCGR8_SRC	R/W1toS	0h	Reads return current value of internal register bit Writes: 0 - No effect 1 - Sets both SRCx and the corresponding SRC_CLRx bitin BOOTCFG_IPCAR8 register
3-1	RESERVED	R		Reserved
0	IPCGR8_REG	R/WtoPH	0h	Reads return 0 Writes: 0 - No effect 1 - Creates an inter-core interrupt

Table 5-112. Register Call Summary for BOOTCFG_IPCGR8

BOOT_CFG Functional Description • Inter-processor Communication Registers: [0]
Control Module (BOOT_CFG) • BOOT_CFG Integration: [0]

Table 5-112. Register Call Summary for BOOTCFG_IPCGR8 (continued)

BOOT_CFG Registers
• BOOTCFG_IPCGR8 Register (Offset = 260h) [reset = 0h]: [0][1] • BOOT_CFG Registers: [0] • BOOTCFG_IPCAR8 Register (Offset = 2A0h) [reset = 0h]: [0]

5.1.4.30 BOOTCFG_IPCGR11 Register (Offset = 26Ch) [reset = 0h]

BOOTCFG_IPCGR11 is shown in Figure 5-35 and described in Table 5-114.

Register facilitate inter-core interrupt to ICSS0. A write of 1 to the IPCG field of the BOOTCFG_IPCGR11 register generates an interrupt pulse to ICSS0. The register also provides a Source ID facility identifying up to 28 different sources of interrupt. Allocation of source bits to source processor and meaning is entirely based on software convention. There can be numerous sources for the register as this is completely controlled by software. Any master that has access to BOOT_CFG module space can write to this register. Register is KICK un-locked, that means software need not go through KICK sequence in order to write to this register.

Table 5-113. BOOTCFG_IPCGR11 Instances

Instance	Physical Address
BOOT_CFG	0262 026Ch

Figure 5-35. BOOTCFG_IPCGR11 Register

31	30	29	28	27	26	25	24
IPCGR11_SRC							
R/W1toS-0h							
23	22	21	20	19	18	17	16
IPCGR11_SRC							
R/W1toS-0h							
15	14	13	12	11	10	9	8
IPCGR11_SRC							
R/W1toS-0h							
7	6	5	4	3	2	1	0
IPCGR11_SRC				RESERVED			IPCGR11_REG
R/W1toS-0h				R-			R/WtoPH-0h

LEGEND: R = Read Only; R/W1toS = Read/Write 1 to Set Bit; R/WtoPH = Read/Write to Pulse High; -*n* = value after reset

Table 5-114. BOOTCFG_IPCGR11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	IPCGR11_SRC	R/W1toS	0h	Reads return current value of internal register bit Writes: 0 - No effect 1 - Sets both SRCx and the corresponding SRC_CLRx bit in BOOTCFG_IPCAR11 register
3-1	RESERVED	R		Reserved
0	IPCGR11_REG	R/WtoPH	0h	Reads return 0 Writes: 0 - No effect 1 - Creates an inter-core interrupt

Table 5-115. Register Call Summary for BOOTCFG_IPCGR11

BOOT_CFG Functional Description • Inter-processor Communication Registers: [0]
Control Module (BOOT_CFG) • BOOT_CFG Integration: [0]

Table 5-115. Register Call Summary for BOOTCFG_IPCGR11 (continued)

BOOT_CFG Registers
• BOOTCFG_IPCGR11 Register (Offset = 26Ch) [reset = 0h]: [0][1] • BOOT_CFG Registers: [0] • BOOTCFG_IPCAR11 Register (Offset = 2ACh) [reset = 0h]: [0]

5.1.4.31 BOOTCFG_IPCGR12 Register (Offset = 270h) [reset = 0h]

BOOTCFG_IPCGR12 is shown in Figure 5-36 and described in Table 5-117.

Register facilitate inter-core interrupt to ICSS0. A write of 1 to the IPCG field of the BOOTCFG_IPCGR12 register generates an interrupt pulse to ICSS0. The register also provides a Source ID facility identifying up to 28 different sources of interrupt. Allocation of source bits to source processor and meaning is entirely based on software convention. There can be numerous sources for the register as this is completely controlled by software. Any master that has access to BOOT_CFG module space can write to this register. Register is KICK un-locked, that means software need not go through KICK sequence in order to write to this register.

Table 5-116. BOOTCFG_IPCGR12 Instances

Instance	Physical Address
BOOT_CFG	0262 0270h

Figure 5-36. BOOTCFG_IPCGR12 Register

31	30	29	28	27	26	25	24
IPCGR12_SRC							
R/W1toS-0h							
23	22	21	20	19	18	17	16
IPCGR12_SRC							
R/W1toS-0h							
15	14	13	12	11	10	9	8
IPCGR12_SRC							
R/W1toS-0h							
7	6	5	4	3	2	1	0
IPCGR12_SRC				RESERVED			IPCGR12_REG
R/W1toS-0h				R-			R/WtoPH-0h

LEGEND: R = Read Only; R/W1toS = Read/Write 1 to Set Bit; R/WtoPH = Read/Write to Pulse High; -*n* = value after reset

Table 5-117. BOOTCFG_IPCGR12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	IPCGR12_SRC	R/W1toS	0h	Reads return current value of internal register bit Writes: 0 - No effect 1 - Sets both SRCx and the corresponding SRC_CLRxbit in BOOTCFG_IPCAR12 register
3-1	RESERVED	R		Reserved
0	IPCGR12_REG	R/WtoPH	0h	Reads return 0 Writes: 0 - No effect 1 - Creates an inter-core interrupt

Table 5-118. Register Call Summary for BOOTCFG_IPCGR12

BOOT_CFG Functional Description • Inter-processor Communication Registers: [0]
Control Module (BOOT_CFG) • BOOT_CFG Integration: [0]

Table 5-118. Register Call Summary for BOOTCFG_IPCGR12 (continued)

BOOT_CFG Registers
• BOOTCFG_IPCGR12 Register (Offset = 270h) [reset = 0h]: [0][1] • BOOT_CFG Registers: [0] • BOOTCFG_IPCAR12 Register (Offset = 2ACh) [reset = 0h]: [0]

5.1.4.32 BOOTCFG_IPCGR13 Register (Offset = 274h) [reset = 0h]

BOOTCFG_IPCGR13 is shown in Figure 5-37 and described in Table 5-120.

Register facilitate inter-core interrupt to ICSS1. A write of 1 to the IPCG field of the BOOTCFG_IPCGR13 register generates an interrupt pulse to ICSS1. The register also provides a Source ID facility identifying up to 28 different sources of interrupt. Allocation of source bits to source processor and meaning is entirely based on software convention. There can be numerous sources for the register as this is completely controlled by software. Any master that has access to BOOT_CFG module space can write to this register. Register is KICK un-locked, that means software need not go through KICK sequence in order to write to this register.

Table 5-119. BOOTCFG_IPCGR13 Instances

Instance	Physical Address
BOOT_CFG	0262 0274h

Figure 5-37. BOOTCFG_IPCGR13 Register

31	30	29	28	27	26	25	24
IPCGR13_SRC							
R/W1toS-0h							
23	22	21	20	19	18	17	16
IPCGR13_SRC							
R/W1toS-0h							
15	14	13	12	11	10	9	8
IPCGR13_SRC							
R/W1toS-0h							
7	6	5	4	3	2	1	0
IPCGR13_SRC				RESERVED			IPCGR13_REG
R/W1toS-0h				R-			R/WtoPH-0h

LEGEND: R = Read Only; R/W1toS = Read/Write 1 to Set Bit; R/WtoPH = Read/Write to Pulse High; -*n* = value after reset

Table 5-120. BOOTCFG_IPCGR13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	IPCGR13_SRC	R/W1toS	0h	Reads return current value of internal register bit Writes: 0 - No effect 1 - Sets both SRCx and the corresponding SRC_CLRx bit in BOOTCFG_IPCAR13 register
3-1	RESERVED	R		Reserved
0	IPCGR13_REG	R/WtoPH	0h	Reads return 0 Writes: 0 - No effect 1 - Creates an inter-core interrupt

Table 5-121. Register Call Summary for BOOTCFG_IPCGR13

BOOT_CFG Functional Description • Inter-processor Communication Registers: [0]
Control Module (BOOT_CFG) • BOOT_CFG Integration: [0]

Table 5-121. Register Call Summary for BOOTCFG_IPCGR13 (continued)

BOOT_CFG Registers
• BOOTCFG_IPCGR13 Register (Offset = 274h) [reset = 0h]: [0][1] • BOOT_CFG Registers: [0] • BOOTCFG_IPCAR13 Register (Offset = 2B4h) [reset = 0h]: [0]

5.1.4.33 BOOTCFG_IPCGR14 Register (Offset = 278h) [reset = 0h]

BOOTCFG_IPCGR14 is shown in Figure 5-38 and described in Table 5-123.

Register facilitate inter-core interrupt to ICSS1. A write of 1 to the IPCG field of the BOOTCFG_IPCGR14 register generates an interrupt pulse to ICSS1. The register also provides a Source ID facility identifying up to 28 different sources of interrupt. Allocation of source bits to source processor and meaning is entirely based on software convention. There can be numerous sources for the register as this is completely controlled by software. Any master that has access to BOOT_CFG module space can write to this register. Register is KICK un-locked, that means software need not go through KICK sequence in order to write to this register.

Table 5-122. BOOTCFG_IPCGR14 Instances

Instance	Physical Address
BOOT_CFG	0262 0278h

Figure 5-38. BOOTCFG_IPCGR14 Register

31	30	29	28	27	26	25	24
IPCGR14_SRC							
R/W1toS-0h							
23	22	21	20	19	18	17	16
IPCGR14_SRC							
R/W1toS-0h							
15	14	13	12	11	10	9	8
IPCGR14_SRC							
R/W1toS-0h							
7	6	5	4	3	2	1	0
IPCGR14_SRC				RESERVED			IPCGR14_REG
R/W1toS-0h				R-			R/WtoPH-0h

LEGEND: R = Read Only; R/W1toS = Read/Write 1 to Set Bit; R/WtoPH = Read/Write to Pulse High; -*n* = value after reset

Table 5-123. BOOTCFG_IPCGR14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	IPCGR14_SRC	R/W1toS	0h	Reads return current value of internal register bit Writes: 0 - No effect 1 - Sets both SRCx and the corresponding SRC_CLRx bit in BOOTCFG_IPCAR14 register
3-1	RESERVED	R		Reserved
0	IPCGR14_REG	R/WtoPH	0h	Reads return 0 Writes: 0 - No effect 1 - Creates an inter-core interrupt

Table 5-124. Register Call Summary for BOOTCFG_IPCGR14

BOOT_CFG Functional Description • Inter-processor Communication Registers: [0]
Control Module (BOOT_CFG) • BOOT_CFG Integration: [0]

Table 5-124. Register Call Summary for BOOTCFG_IPCGR14 (continued)

BOOT_CFG Registers
• BOOTCFG_IPCAR14 Register (Offset = 2B8h) [reset = 0h]: [0] • BOOT_CFG Registers: [0] • BOOTCFG_IPCGR14 Register (Offset = 278h) [reset = 0h]: [0][1]

5.1.4.34 BOOTCFG_IPCGRH Register (Offset = 27Ch) [reset = 0h]

BOOTCFG_IPCGRH is shown in Figure 5-39 and described in Table 5-126.

Register facilitate interrupt to external host. Operation and use of the BOOTCFG_IPCGRH register is the same as for other IPCGR registers. The interrupt output pulse created by the BOOTCFG_IPCGRH register appears on device pin HOUT. The host interrupt output pulse is stretched so that it is asserted for four bootcfg clock cycles (CLK/6) followed by a deassertion of four bootcfg clock cycles. Generating the pulse results in a pulse-blocking window that is eight CLK/6-cycles long. Back-to-back writes to the BOOTCFG_IPCGRH register with the IPCGRH_REG bit (bit 0) set, generates only one pulse if the back-to-back writes to BOOTCFG_IPCGRH are less than the eight CLK/6 cycle window - the pulse blocking window. To generate back-to-back pulses, the back-to-back writes to the BOOTCFG_IPCGRH register must be written after the eight CLK/6 cycle pulse-blocking window has elapsed. Register is KICK un-locked, that means software need not go through KICK sequence in order to write to this register.

Table 5-125. BOOTCFG_IPCGRH Instances

Instance	Physical Address
BOOT_CFG	0262 027Ch

Figure 5-39. BOOTCFG_IPCGRH Register

31	30	29	28	27	26	25	24
IPCGRH_SRC							
R/W1toS-0h							
23	22	21	20	19	18	17	16
IPCGRH_SRC							
R/W1toS-0h							
15	14	13	12	11	10	9	8
IPCGRH_SRC							
R/W1toS-0h							
7	6	5	4	3	2	1	0
IPCGRH_SRC				RESERVED			IPCGRH_REG
R/W1toS-0h				R-			R/WtoPH-0h

LEGEND: R = Read Only; R/W1toS = Read/Write 1 to Set Bit; R/WtoPH = Read/Write to Pulse High; -*n* = value after reset

Table 5-126. BOOTCFG_IPCGRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	IPCGRH_SRC	R/W1toS	0h	Reads return current value of internal register bit Writes: 0 - No effect 1 - Sets both SRCx and the corresponding SRC_CLRx bit in BOOTCFG_IPCARH register
3-1	RESERVED	R		Reserved
0	IPCGRH_REG	R/WtoPH	0h	Reads return 0 Writes: 0 - No effect 1 - Creates an interrupt pulse on device pin (host interrupt/event output in HOUT pin)

Table 5-127. Register Call Summary for BOOTCFG_IPCGRH

BOOT_CFG Functional Description • Inter-processor Communication Registers: [0][1][2][3]
BOOT_CFG Registers • BOOTCFG_IPCGRH Register (Offset = 27Ch) [reset = 0h]: [0][1][2][3][4][5] • BOOT_CFG Registers: [0] • BOOTCFG_IPCARH Register (Offset = 2BCh) [reset = 0h]: [0]

5.1.4.35 BOOTCFG_IPCAR0 Register (Offset = 280h) [reset = 0h]

BOOTCFG_IPCAR0 is shown in Figure 5-40 and described in Table 5-129.

Register facilitate inter-core interrupt acknowledgement for DSP core 0. The register also provides a Source ID facility by which up to 28 different sources of interrupt can be identified. Allocation of source bits to source processor and meaning is entirely based on software convention. Virtually anything can be a source for the register as this is completely controlled by software. Any master that has access to BOOT_CFG module space can write to this register. Register is KICK un-locked, that means software need not go through KICK sequence in order to write to this register.

Table 5-128. BOOTCFG_IPCAR0 Instances

Instance	Physical Address
BOOT_CFG	0262 0280h

Figure 5-40. BOOTCFG_IPCAR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPCGR0_SRC_CLR																												RESERVED			
R/W1toCl-0h																												R-			

LEGEND: R = Read Only; R/W1toCl = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 5-129. BOOTCFG_IPCAR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	IPCGR0_SRC_CLR	R/W1toCl	0h	Reads return current value of internal register bit Writes: 0 - No effect 1 - Clears both SRC_CLRx and the corresponding SRCx bit in BOOTCFG_IPCGR0 register
3-0	RESERVED	R		Reserved

Table 5-130. Register Call Summary for BOOTCFG_IPCAR0

BOOT_CFG Functional Description • Inter-processor Communication Registers: [0]
BOOT_CFG Registers • BOOTCFG_IPCGR0 Register (Offset = 240h) [reset = 0h]: [0] • BOOT_CFG Registers: [0] • BOOTCFG_IPCAR0 Register (Offset = 280h) [reset = 0h]: [0]

5.1.4.36 BOOTCFG_IPCAR8 Register (Offset = 2A0h) [reset = 0h]

BOOTCFG_IPCAR8 is shown in Figure 5-41 and described in Table 5-132.

Register facilitate inter-core interrupt acknowledgement for Arm core 0. The register also provide a Source ID facility by which up to 28 different sources of interrupt can be identified. Allocation of source bits to source processor and meaning is entirely based on software convention. Virtually anything can be a source for the register as this is completely controlled by software. Any master that has access to BOOT_CFG module space can write to this register. Register is KICK un-locked, that means software need not go through KICK sequence in order to write to this register.

Table 5-131. BOOTCFG_IPCAR8 Instances

Instance	Physical Address
BOOT_CFG	0262 02A0h

Figure 5-41. BOOTCFG_IPCAR8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPCGR8_SRC_CLR																												RESERVED			
R/W1toCl-0h																												R-			

LEGEND: R = Read Only; R/W1toCl = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 5-132. BOOTCFG_IPCAR8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	IPCGR8_SRC_CLR	R/W1toCl	0h	Reads return current value of internal register bit Writes: 0 - No effect 1 - Clears both SRC_CLRx and the corresponding SRCxbit in BOOTCFG_IPCGR8 register
3-0	RESERVED	R		Reserved

Table 5-133. Register Call Summary for BOOTCFG_IPCAR8

BOOT_CFG Functional Description • Inter-processor Communication Registers: [0]
BOOT_CFG Registers • BOOTCFG_IPCGR8 Register (Offset = 260h) [reset = 0h]: [0] • BOOT_CFG Registers: [0] • BOOTCFG_IPCAR8 Register (Offset = 2A0h) [reset = 0h]: [0]

5.1.4.37 BOOTCFG_IPCAR11 Register (Offset = 2ACh) [reset = 0h]

BOOTCFG_IPCAR11 is shown in Figure 5-42 and described in Table 5-135.

Register facilitate inter-core interrupt acknowledgement for ICSS0. The register also provides a Source ID facility by which up to 28 different sources of interrupt can be identified. Allocation of source bits to source processor and meaning is entirely based on software convention. Virtually anything can be a source for the register as this is completely controlled by software. Any master that has access to BOOT_CFG module space can write to this register. Register is KICK un-locked, that means software need not go through KICK sequence in order to write to this register.

Table 5-134. BOOTCFG_IPCAR11 Instances

Instance	Physical Address
BOOT_CFG	0262 02ACh

Figure 5-42. BOOTCFG_IPCAR11 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPCGR11_SRC_CLR																												RESERVED			
R/W1toC-0h																												R-			

LEGEND: R = Read Only; R/W1toC = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 5-135. BOOTCFG_IPCAR11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	IPCGR11_SRC_CLR	R/W1toC	0h	Reads return current value of internal register bit Writes: 0 - No effect 1 - Clears both SRC_CLRx and the corresponding SRCx bit in BOOTCFG_IPCGR11 register
3-0	RESERVED	R		Reserved

Table 5-136. Register Call Summary for BOOTCFG_IPCAR11

BOOT_CFG Functional Description • Inter-processor Communication Registers: [0]
BOOT_CFG Registers • BOOTCFG_IPCGR11 Register (Offset = 26Ch) [reset = 0h]: [0] • BOOT_CFG Registers: [0] • BOOTCFG_IPCAR11 Register (Offset = 2ACh) [reset = 0h]: [0]

5.1.4.38 BOOTCFG_IPCAR12 Register (Offset = 2ACh) [reset = 0h]

BOOTCFG_IPCAR12 is shown in Figure 5-43 and described in Table 5-138.

Register facilitate inter-core interrupt acknowledgement for ICSS0. The register also provides a Source ID facility by which up to 28 different sources of interrupt can be identified. Allocation of source bits to source processor and meaning is entirely based on software convention. Virtually anything can be a source for the register asthis is completely controlled by software. Any master that has access to BOOT_CFG module space can write to this register. Register is KICK un-locked, that means software need not go through KICK sequence in order to write to this register.

Table 5-137. BOOTCFG_IPCAR12 Instances

Instance	Physical Address
BOOT_CFG	0262 02ACh

Figure 5-43. BOOTCFG_IPCAR12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPCGR12_SRC_CLR																												RESERVED			
R/W1toC-0h																												R-			

LEGEND: R = Read Only; R/W1toC = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 5-138. BOOTCFG_IPCAR12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	IPCGR12_SRC_CLR	R/W1toC	0h	Reads return current value of internal register bit Writes: 0 - No effect 1 - Clears both SRC_CLRx and the corresponding SRCx bit in BOOTCFG_IPCGR12 register
3-0	RESERVED	R		Reserved

Table 5-139. Register Call Summary for BOOTCFG_IPCAR12

BOOT_CFG Functional Description • Inter-processor Communication Registers: [0]
BOOT_CFG Registers • BOOTCFG_IPCGR12 Register (Offset = 270h) [reset = 0h]: [0] • BOOT_CFG Registers: [0] • BOOTCFG_IPCAR12 Register (Offset = 2ACh) [reset = 0h]: [0]

5.1.4.39 BOOTCFG_IPCAR13 Register (Offset = 2B4h) [reset = 0h]

BOOTCFG_IPCAR13 is shown in Figure 5-44 and described in Table 5-141.

Register facilitate inter-core interrupt acknowledgement for ICSS1. The register also provides a Source ID facility by which up to 28 different sources of interrupt can be identified. Allocation of source bits to source processor and meaning is entirely based on software convention. Virtually anything can be a source for the register as this is completely controlled by software. Any master that has access to BOOT_CFG module space can write to this register. Register is KICK un-locked, that means software need not go through KICK sequence in order to write to this register.

Table 5-140. BOOTCFG_IPCAR13 Instances

Instance	Physical Address
BOOT_CFG	0262 02B4h

Figure 5-44. BOOTCFG_IPCAR13 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPCGR13_SRC_CLR																												RESERVED			
R/W1toC-0h																												R-			

LEGEND: R = Read Only; R/W1toC = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 5-141. BOOTCFG_IPCAR13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	IPCGR13_SRC_CLR	R/W1toC	0h	Reads return current value of internal register bit Writes: 0 - No effect 1 - Clears both SRC_CLRx and the corresponding SRCx bit in BOOTCFG_IPCGR13 register
3-0	RESERVED	R		Reserved

Table 5-142. Register Call Summary for BOOTCFG_IPCAR13

BOOT_CFG Functional Description • Inter-processor Communication Registers: [0]
BOOT_CFG Registers • BOOTCFG_IPCGR13 Register (Offset = 274h) [reset = 0h]: [0] • BOOT_CFG Registers: [0] • BOOTCFG_IPCAR13 Register (Offset = 2B4h) [reset = 0h]: [0]

5.1.4.40 BOOTCFG_IPCAR14 Register (Offset = 2B8h) [reset = 0h]

BOOTCFG_IPCAR14 is shown in Figure 5-45 and described in Table 5-144.

Register facilitate inter-core interrupt acknowledgement for ICSS1. The register also provides a Source ID facility by which up to 28 different sources of interrupt can be identified. Allocation of source bits to source processor and meaning is entirely based on software convention. Virtually anything can be a source for the register as this is completely controlled by software. Any master that has access to BOOT_CFG module space can write to this register. Register is KICK un-locked, that means software need not go through KICK sequence in order to write to this register.

Table 5-143. BOOTCFG_IPCAR14 Instances

Instance	Physical Address
BOOT_CFG	0262 02B8h

Figure 5-45. BOOTCFG_IPCAR14 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPCGR14_SRC_CLR																												RESERVED			
R/W1toC-0h																												R-			

LEGEND: R = Read Only; R/W1toC = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 5-144. BOOTCFG_IPCAR14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	IPCGR14_SRC_CLR	R/W1toC	0h	Reads return current value of internal register bit Writes: 0 - No effect 1 - Clears both SRC_CLRx and the corresponding SRCx bit in BOOTCFG_IPCGR14 register
3-0	RESERVED	R		Reserved

Table 5-145. Register Call Summary for BOOTCFG_IPCAR14

BOOT_CFG Functional Description • Inter-processor Communication Registers: [0]
BOOT_CFG Registers • BOOTCFG_IPCAR14 Register (Offset = 2B8h) [reset = 0h]: [0] • BOOT_CFG Registers: [0] • BOOTCFG_IPCGR14 Register (Offset = 278h) [reset = 0h]: [0]

5.1.4.41 BOOTCFG_IPCARH Register (Offset = 2BCh) [reset = 0h]

BOOTCFG_IPCARH is shown in Figure 5-46 and described in Table 5-147.

Register facilitate interrupt acknowledgement to external host. Operation and use of the BOOTCFG_IPCARH register is the same as for other IPCAR registers. Register is KICK un-locked, that means software need not go through KICK sequence in order to write to this register.

Table 5-146. BOOTCFG_IPCARH Instances

Instance	Physical Address
BOOT_CFG	0262 02BCh

Figure 5-46. BOOTCFG_IPCARH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPCGRH_SRC_CLR																												RESERVED			
R/W1toC-0h																												R-			

LEGEND: R = Read Only; R/W1toC = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 5-147. BOOTCFG_IPCARH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	IPCGRH_SRC_CLR	R/W1toC	0h	Reads return current value of internal register bit Writes: 0 - No effect 1 - Clears both SRC_CLRx and the corresponding SRCxbit in BOOTCFG_IPCGRH register
3-0	RESERVED	R		Reserved

Table 5-148. Register Call Summary for BOOTCFG_IPCARH

BOOT_CFG Functional Description • Inter-processor Communication Registers: [0]
BOOT_CFG Registers • BOOTCFG_IPCGRH Register (Offset = 27Ch) [reset = 0h]: [0] • BOOT_CFG Registers: [0] • BOOTCFG_IPCARH Register (Offset = 2BCh) [reset = 0h]: [0][1]

5.1.4.42 BOOTCFG_TINPSEL0 Register (Offset = 2D8h) [reset = 10101010h]

BOOTCFG_TINPSEL0 is shown in Figure 5-47 and described in Table 5-150.

Register selects timer pin inputs for Timer 0 to 3.

Table 5-149. BOOTCFG_TINPSEL0 Instances

Instance	Physical Address
BOOT_CFG	0262 02D8h

Figure 5-47. BOOTCFG_TINPSEL0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED	TINPHSEL3			RESERVED	TINPLSEL3			RESERVED	TINPHSEL2			RESERVED	TINPLSEL2		
R-	R/W-1h			R-	R/W-0h			R-	R/W-1h			R-	R/W-0h		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	TINPHSEL1			RESERVED	TINPLSEL1			RESERVED	TINPHSEL0			RESERVED	TINPLSEL0		
R-	R/W-1h			R-	R/W-0h			R-	R/W-1h			R-	R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-150. BOOTCFG_TINPSEL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R		Reserved
30-28	TINPHSEL3	R/W	1h	Input select for TIMER_3 high Field values (Others are reserved): 0h = TIMI0 1h = TIMI1
27	RESERVED	R		Reserved
26-24	TINPLSEL3	R/W	0h	Input select for TIMER_3 low Field values (Others are reserved): 0h = TIMI0 1h = TIMI1
23	RESERVED	R		Reserved
22-20	TINPHSEL2	R/W	1h	Input select for TIMER_2 high Field values (Others are reserved): 0h = TIMI0 1h = TIMI1
19	RESERVED	R		Reserved
18-16	TINPLSEL2	R/W	0h	Input select for TIMER_2 low Field values (Others are reserved): 0h = TIMI0 1h = TIMI1
15	RESERVED	R		Reserved
14-12	TINPHSEL1	R/W	1h	Input select for TIMER_1 high Field values (Others are reserved): 0h = TIMI0 1h = TIMI1
11	RESERVED	R		Reserved

Table 5-150. BOOTCFG_TINPSEL0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10-8	TINPLSEL1	R/W	0h	Input select for TIMER_1 low Field values (Others are reserved): 0h = TIMI0 1h = TIMI1
7	RESERVED	R		Reserved
6-4	TINPHSEL0	R/W	1h	Input select for TIMER_0 high Field values (Others are reserved): 0h = TIMI0 1h = TIMI1
3	RESERVED	R		Reserved
2-0	TINPLSEL0	R/W	0h	Input select for TIMER_0 low Field values (Others are reserved): 0h = TIMI0 1h = TIMI1

Table 5-151. Register Call Summary for BOOTCFG_TINPSEL0

BOOT_CFG Functional Description • Timer Pin Manager Control Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_TINPSEL0 Register (Offset = 2D8h) [reset = 10101010h]: [0]

5.1.4.43 BOOTCFG_TINPSEL1 Register (Offset = 2DCh) [reset = 1010h]

BOOTCFG_TINPSEL1 is shown in Figure 5-48 and described in Table 5-153.

Register selects timer pin inputs for TIMER_4 and TIMER_5

Table 5-152. BOOTCFG_TINPSEL1 Instances

Instance	Physical Address
BOOT_CFG	0262 02DCh

Figure 5-48. BOOTCFG_TINPSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	TINPHSEL5			RESERVED	TINPLSEL5			RESERVED	TINPHSEL4			RESERVED	TINPLSEL4		
R-	R/W-1h			R-	R/W-0h			R-	R/W-1h			R-	R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-153. BOOTCFG_TINPSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R		Reserved
14-12	TINPHSEL5	R/W	1h	Input select for TIMER_5 high Field values (Others are reserved): 0h = TIMI0 1h = TIMI1
11	RESERVED	R		Reserved
10-8	TINPLSEL5	R/W	0h	Input select for TIMER_5 low Field values (Others are reserved): 0h = TIMI0 1h = TIMI1
7	RESERVED	R		Reserved
6-4	TINPHSEL4	R/W	1h	Input select for TIMER_4 high Field values (Others are reserved): 0h = TIMI0 1h = TIMI1
3	RESERVED	R		Reserved
2-0	TINPLSEL4	R/W	0h	Input select for TIMER_4 low Field values (Others are reserved): 0h = TIMI0 1h = TIMI1

Table 5-154. Register Call Summary for BOOTCFG_TINPSEL1

BOOT_CFG Functional Description • Timer Pin Manager Control Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_TINPSEL1 Register (Offset = 2DCh) [reset = 1010h]: [0]

5.1.4.44 BOOTCFG_TOUTPSEL0 Register (Offset = 2F8h) [reset = 302h]

BOOTCFG_TOUTPSEL0 is shown in Figure 5-49 and described in Table 5-156.

Register to select timer pin outputs.

Table 5-155. BOOTCFG_TOUTPSEL0 Instances

Instance	Physical Address
BOOT_CFG	0262 02F8h

Figure 5-49. BOOTCFG_TOUTPSEL0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		TOUTPSEL1						RESERVED		TOUTPSEL0					
R-		R/W-3h						R-		R/W-2h					

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-156. BOOTCFG_TOUTPSEL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R		Reserved
13-8	TOUTPSEL1	R/W	3h	Output select for TIMO1 Field values (Others are reserved) 0h = TOUTL0 1h = TOUTH0 2h = TOUTL1 3h = TOUTH1 4h = TOUTL2 5h = TOUTH2 6h = TOUTL3 7h = TOUTH3 8h = TOUTL4 9h = TOUTH4 Ah = TOUTL5 Bh = TOUTH5
7-6	RESERVED	R		Reserved
5-0	TOUTPSEL0	R/W	2h	Output select for TIMO0 Field values (Others are reserved): 0h = TOUTL0 1h = TOUTH0 2h = TOUTL1 3h = TOUTH1 4h = TOUTL2 5h = TOUTH2 6h = TOUTL3 7h = TOUTH3 8h = TOUTL4 9h = TOUTH4 Ah = TOUTL5 Bh = TOUTH5

Table 5-157. Register Call Summary for BOOTCFG_TOUTPSEL0

BOOT_CFG Functional Description • Timer Pin Manager Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_TOUTPSEL0 Register (Offset = 2F8h) [reset = 302h]: [0] • BOOT_CFG Registers: [0]

5.1.4.45 BOOTCFG_RSTMUX0 Register (Offset = 308h) [reset = 80h]

BOOTCFG_RSTMUX0 is shown in Figure 5-50 and described in Table 5-159.

Reset multiplex register controls the WDT (Watchdog Timer) timeout event actions for C66x core 0.

Table 5-158. BOOTCFG_RSTMUX0 Instances

Instance	Physical Address
BOOT_CFG	0262 0308h

Figure 5-50. BOOTCFG_RSTMUX0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED						RSTMUX_EVSTATCLR0	RESERVED
R-						R/WtoPH-0h	R-
7	6	5	4	3	2	1	0
RSTMUX_DELAY0			RSTMUX_EVSTAT0	RSTMUX_OMODE0			RSTMUX_LOCK0
R/W-4h			R-0h	R/W-0h			R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; R/WtoPH = Read/Write to Pulse High; *-n* = value after reset

Table 5-159. BOOTCFG_RSTMUX0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R		Reserved
9	RSTMUX_EVSTATCLR0	R/WtoPH	0h	Clear event status 0 - Writing 0 has no effect 1 - Writing 1 to this bit clears the EVTSTAT bit
8	RESERVED	R		Reserved
7-5	RSTMUX_DELAY0	R/W	4h	Delay cycles between NMI and local reset to C66x core incase of C66x WD timer, or delay between GIC interrupt and device reset in case of A15 WD timer Field values (Others are reserved): 000b - 256 CLK/6 cycles delay between NMI and local reset, when OMODE=100b 001b - 512 CLK/6 cycles delay between NMI and local reset, when OMODE=100b 010b - 1024 CLK/6 cycles delay between NMI and local reset, when OMODE=100b 011b - 2048 CLK/6 cycles delay between NMI and local reset, when OMODE=100b 100b - 4096 CLK/6 cycles delay between NMI and local reset, when OMODE=100b 101b - 8192 CLK/6 cycles delay between NMI and local reset, when OMODE=100b 110b - 16384 CLK/6 cycles delay between NMI and local reset, when OMODE=100b 111b - 32768 CLK/6 cycles delay between NMI and local reset, when OMODE=100b

Table 5-159. BOOTCFG_RSTMUX0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	RSTMUX_EVSTAT0	R	0h	Event status 0 - No event received 1 - WD timer event received by Reset Mux block
3-1	RSTMUX_OMODE0	R/W	0h	Watchdog timer event operation mode Field values (Others are reserved): 000b - WD Timer event to the Reset Mux block does not cause any output event 001b - Reserved 010b - WD Timer event to the Reset Mux block causes local reset to C66x core 011b - WD Timer event to the Reset Mux block causes NMI to C66x core 100b - WD Timer event to the Reset Mux block causes NMI followed by local reset to C66x core. Delay between NMI and local reset is set in DELAY bit field. 101b - WD Timer event to the Reset Mux block causes device reset (via reset_req_pi_n signal to PLL Controller)
0	RSTMUX_LOCK0	R/W	0h	Lock register fields 0 - Register fields are not locked 1 - Register fields are locked until the next timer reset. This bit is expected to be set to 1 after the software configures the watchdog timer to prevent any accidental modification of the register

Table 5-160. Register Call Summary for BOOTCFG_RSTMUX0

BOOT_CFG Functional Description • Reset Mux Control Registers: [0][1][2][3]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_RSTMUX0 Register (Offset = 308h) [reset = 80h]: [0]

5.1.4.46 BOOTCFG_RSTMUX8 Register (Offset = 328h) [reset = 80h]

BOOTCFG_RSTMUX8 is shown in Figure 5-51 and described in Table 5-162.

Reset multiplex register controls the WDT (Watchdog Timer) timeout event actions for A15 core 0.

Table 5-161. BOOTCFG_RSTMUX8 Instances

Instance	Physical Address
BOOT_CFG	0262 0328h

Figure 5-51. BOOTCFG_RSTMUX8 Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED						RSTMUX_EVSTATCLR8	RESERVED
R-						R/WtoPH-0h	R-
7	6	5	4	3	2	1	0
RSTMUX_DELAY8			RSTMUX_EVSTAT8	RSTMUX_OMODE8			RSTMUX_LOCK8
R/W-4h			R-0h	R/W-0h			R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; R/WtoPH = Read/Write to Pulse High; *-n* = value after reset

Table 5-162. BOOTCFG_RSTMUX8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R		Reserved
9	RSTMUX_EVSTATCLR8	R/WtoPH	0h	Clear event status 0 - Writing 0 has no effect 1 - Writing 1 to this bit clears the EVTSTAT bit
8	RESERVED	R		Reserved
7-5	RSTMUX_DELAY8	R/W	4h	Delay cycles between NMI and local reset to C66x core incase of C66x WD timer, or delay between GIC interrupt and device reset in case of A15 WD timer Field values (Others are reserved): 000b - 256 CLK/6 cycles delay between Interrupt and device reset, when OMODE=100b 001b - 512 CLK/6 cycles delay between Interrupt and device reset, when OMODE=100b 010b - 1024 CLK/6 cycles delay between Interrupt and device reset, when OMODE=100b 011b - 2048 CLK/6 cycles delay between Interrupt and device reset, when OMODE=100b 100b - 4096 CLK/6 cycles delay between Interrupt and device reset, when OMODE=100b 101b - 8192 CLK/6 cycles delay between Interrupt and device reset, when OMODE=100b 110b - 16384 CLK/6 cycles delay between Interrupt and device reset, when OMODE=100b 111b - 32768 CLK/6 cycles delay between Interrupt and device reset, when OMODE=100b

Table 5-162. BOOTCFG_RSTMUX8 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	RSTMUX_EVSTAT8	R	0h	Event status 0 - No event received 1 - WD timer event received by Reset Mux block
3-1	RSTMUX_OMODE8	R/W	0h	Watchdog timer event operation mode Field values (Others are reserved): 000b - WD Timer event to the Reset Mux block does not cause any output event 001b - Reserved 010b - WD Timer event to the Reset Mux block causes device reset (via reset_req_pi_n signal to PLLcontroller). Note that A15 core does not support local reset 011b - WD Timer event to the Reset Mux block causes an interrupt to be sent to GIC (may be routed to nFIQ bysoftware). 100b - WD Timer event to the Reset Mux block causes an interrupt to be sent to GIC (may be routed to nFIQ bysoftware) followed by device reset (via reset_req_pi_n signal to PLL Controller). The delay between interrupt and device reset is set in the DELAY field. 101b - WD Timer event to the Reset Mux block causes device reset (via reset_req_pi_n signal to PLLcontroller)
0	RSTMUX_LOCK8	R/W	0h	Lock register fields 0 - Register fields are not locked 1 - Register fields are locked until the next timer reset. This bit is expected to be set to 1 after the software configures the watchdog timer to prevent any accidental modification of the register

Table 5-163. Register Call Summary for BOOTCFG_RSTMUX8

BOOT_CFG Functional Description • Reset Mux Control Registers: [0][1][2][3]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_RSTMUX8 Register (Offset = 328h) [reset = 80h]: [0]

5.1.4.47 BOOTCFG_MAIN_PLL_CTL0 Register (Offset = 350h) [reset = 5000000h]

BOOTCFG_MAIN_PLL_CTL0 is shown in Figure 5-52 and described in Table 5-165.

Register to control Main PLL

Table 5-164. BOOTCFG_MAIN_PLL_CTL0 Instances

Instance	Physical Address
BOOT_CFG	0262 0350h

Figure 5-52. BOOTCFG_MAIN_PLL_CTL0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BWADJ								RESERVED					PLLM							RESERVED						PLLD						
R/W-05h								R-					R/W-0h							R-						R/W-0h					

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-165. BOOTCFG_MAIN_PLL_CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	BWADJ	R/W	05h	8-bit LSB of a 12-bit BWADJ field that selects the values for the loop bandwidth adjustment. BWADJ[7-0] is controlled from this register and BWADJ[11-8] is controlled from BOOTCFG_MAIN_PLL_CTL1 register. BWADJ[11-0] should be programmed to a value equal to half of PLLM[12-0] value (round down if PLLM has an odd value). Example: If PLLM = 15, then BWADJ = 7
23-19	RESERVED	R		Reserved
18-12	PLLM	R/W	0h	7-bit MSB of a 13-bit PLLM field that selects the values for the multiplication factor. PLLM field is loaded with the multiply factor minus 1. The PLLM[5-0] bits of the multiplier are controlled by the PLLM register inside the PLL Controller and the PLLM[12-6] bits are controlled by this register. This field should be written just before writing to PLLM[5-0] bits of the PLLM register to have the complete 13-bit value latched when the GO operation is initiated in the PLL Controller
11-6	RESERVED	R		Reserved
5-0	PLLD	R/W	0h	A 6-bit field that selects the values for the reference divider. PLLD field is loaded with reference divider value minus 1

Table 5-166. Register Call Summary for BOOTCFG_MAIN_PLL_CTL0

BOOT_CFG Functional Description • PLL Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_MAIN_PLL_CTL0 Register (Offset = 350h) [reset = 5000000h]: [0] • BOOT_CFG Registers: [0]

5.1.4.48 BOOTCFG_MAIN_PLL_CTL1 Register (Offset = 354h) [reset = 40h]

BOOTCFG_MAIN_PLL_CTL1 is shown in Figure 5-53 and described in Table 5-168.

Register to control Main PLL.

Table 5-167. BOOTCFG_MAIN_PLL_CTL1 Instances

Instance	Physical Address
BOOT_CFG	0262 0354h

Figure 5-53. BOOTCFG_MAIN_PLL_CTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED	ENSAT	RESERVED		BWADJ			
R-	R/W-1h	R-0h		R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-168. BOOTCFG_MAIN_PLL_CTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	-	Reserved
6	ENSAT	R/W	1h	Enables saturation behavior, needs to be set to 1 for proper PLLoperation
5-4	RESERVED	R	0h	Reserved
3-0	BWADJ	R/W	0h	4-bit MSB of a 12-bit BWADJ field that selects the values for the loop bandwidth adjustment. BWADJ[7-0] is controlled from MAINPLLCTL0 register and BWADJ[11-8] is controlled from this register. BWADJ[11-0] should be programmed to a value equal to half of PLLM[12-0] value (round down if PLLM has an odd value). Example: If PLLM = 15, then BWADJ = 7

Table 5-169. Register Call Summary for BOOTCFG_MAIN_PLL_CTL1

BOOT_CFG Functional Description • PLL Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_MAIN_PLL_CTL0 Register (Offset = 350h) [reset = 5000000h]: [0] • BOOTCFG_MAIN_PLL_CTL1 Register (Offset = 354h) [reset = 40h]: [0] • BOOT_CFG Registers: [0]

5.1.4.49 BOOTCFG_NSS_PLL_CTL0 Register (Offset = 358h) [reset = 98804C0h]

BOOTCFG_NSS_PLL_CTL0 is shown in Figure 5-54 and described in Table 5-171.

Register to control NSS PLL.

Table 5-170. BOOTCFG_NSS_PLL_CTL0 Instances

Instance	Physical Address
BOOT_CFG	0262 0358h

Figure 5-54. BOOTCFG_NSS_PLL_CTL0 Register

31	30	29	28	27	26	25	24
BWADJ							
R/W-9h							
23	22	21	20	19	18	17	16
BYPASS	CLKOD				PLLM		
R/W-1h	R/W-1h				R/W-13h		
15	14	13	12	11	10	9	8
PLLM							
R/W-13h							
7	6	5	4	3	2	1	0
PLLM			PLLD				
R/W-13h			R/W-0h				

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-171. BOOTCFG_NSS_PLL_CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	BWADJ	R/W	9h	8-bit LSB of a 12-bit BWADJ field that selects the values for the loop bandwidth adjustment. BWADJ[7-0] is controlled from this register and BWADJ[11-8] is controlled from BOOTCFG_NSS_PLL_CTL1 register. BWADJ[11-0] should be programmed to a value equal to half of PLLM[12-0] value (round down if PLLM has an odd value). Example: If PLLM = 15, then BWADJ = 7
23	BYPASS	R/W	1h	Enable bypass mode 0 - Bypass disabled 1 - Bypass enabled
22-19	CLKOD	R/W	1h	PLL output divider (post VCO), supported value = 1 (i.e. divide-by-2)
18-6	PLLM	R/W	13h	A 13-bit field that selects the values for the multiplication factor. PLLM field is loaded with the multiply factor minus 1
5-0	PLLD	R/W	0h	A 6-bit field that selects the values for the reference divider. PLLD field is loaded with reference divide value minus 1

Table 5-172. Register Call Summary for BOOTCFG_NSS_PLL_CTL0

BOOT_CFG Functional Description • PLL Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_NSS_PLL_CTL0 Register (Offset = 358h) [reset = 98804C0h]: [0] • BOOTCFG_NSS_PLL_CTL1 Register (Offset = 35Ch) [reset = 40h]: [0] • BOOT_CFG Registers: [0]

5.1.4.50 BOOTCFG_NSS_PLL_CTL1 Register (Offset = 35Ch) [reset = 40h]

BOOTCFG_NSS_PLL_CTL1 is shown in Figure 5-55 and described in Table 5-174.

Register to control NSS PLL.

Table 5-173. BOOTCFG_NSS_PLL_CTL1 Instances

Instance	Physical Address
BOOT_CFG	0262 035Ch

Figure 5-55. BOOTCFG_NSS_PLL_CTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED	PLLRST	RESERVED					
R-0h	R/W-0h	R-0h					
7	6	5	4	3	2	1	0
RESERVED	ENSAT	RESERVED		BWADJ			
R-0h	R/W-1h	R-0h		R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-174. BOOTCFG_NSS_PLL_CTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R		Reserved
14	PLLRST	R/W	0h	PLL reset bit 0 - PLL reset is released 1 - PLL reset is asserted
13-7	RESERVED	R	0h	Reserved
6	ENSAT	R/W	1h	Enables saturation behavior, needs to be set to 1 for proper PLL operation
5-4	RESERVED	R	0h	Reserved
3-0	BWADJ	R/W	0h	4-bit MSB of a 12-bit BWADJ field that selects the values for the loop bandwidth adjustment. BWADJ[7-0] is controlled from BOOTCFG_NSS_PLL_CTL0 register and BWADJ[11-8] is controlled from this register. BWADJ[11-0] should be programmed to a value equal to half of PLLM[12-0] value (round down if PLLM has an odd value). Example: If PLLM = 15, then BWADJ = 7

Table 5-175. Register Call Summary for BOOTCFG_NSS_PLL_CTL1

BOOT_CFG Functional Description • PLL Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_NSS_PLL_CTL0 Register (Offset = 358h) [reset = 98804C0h]: [0] • BOOTCFG_NSS_PLL_CTL1 Register (Offset = 35Ch) [reset = 40h]: [0] • BOOT_CFG Registers: [0]

5.1.4.51 BOOTCFG_DDR3A_PLL_CTL0 Register (Offset = 360h) [reset = 98804C0h]

BOOTCFG_DDR3A_PLL_CTL0 is shown in Figure 5-56 and described in Table 5-177.

Register to control DDR3A PLL.

Table 5-176. BOOTCFG_DDR3A_PLL_CTL0 Instances

Instance	Physical Address
BOOT_CFG	0262 0360h

Figure 5-56. BOOTCFG_DDR3A_PLL_CTL0 Register

31	30	29	28	27	26	25	24
BWADJ							
R/W-9h							
23	22	21	20	19	18	17	16
BYPASS	CLKOD				PLLM		
R/W-1h	R/W-1h				R/W-13h		
15	14	13	12	11	10	9	8
PLLM							
R/W-13h							
7	6	5	4	3	2	1	0
PLLM		PLLD					
R/W-13h		R/W-0h					

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-177. BOOTCFG_DDR3A_PLL_CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	BWADJ	R/W	9h	8-bit LSB of a 12-bit BWADJ field that selects the values for the loop bandwidth adjustment. BWADJ[7-0] is controlled from this register and BWADJ[11-8] is controlled from BOOTCFG_DDR3A_PLL_CTL1 register. BWADJ[11-0] should be programmed to a value equal to half of PLLM[12-0] value (round down if PLLM has an odd value). Example: If PLLM = 15, then BWADJ = 7
23	BYPASS	R/W	1h	Enable bypass mode 0 - Bypass disabled 1 - Bypass enabled
22-19	CLKOD	R/W	1h	PLL output divider (post VCO), supported value = 1 (i.e. divide-by-2)
18-6	PLLM	R/W	13h	A 13-bit field that selects the values for the multiplication factor. PLLM field is loaded with the multiply factor minus 1
5-0	PLLD	R/W	0h	A 6-bit field that selects the values for the reference divider. PLLD field is loaded with reference divide value minus 1

Table 5-178. Register Call Summary for BOOTCFG_DDR3A_PLL_CTL0

BOOT_CFG Functional Description • PLL Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_DDR3A_PLL_CTL0 Register (Offset = 360h) [reset = 98804C0h]: [0] • BOOT_CFG Registers: [0] • BOOTCFG_DDR3A_PLL_CTL1 Register (Offset = 364h) [reset = 40h]: [0]

5.1.4.52 BOOTCFG_DDR3A_PLL_CTL1 Register (Offset = 364h) [reset = 40h]

BOOTCFG_DDR3A_PLL_CTL1 is shown in Figure 5-57 and described in Table 5-180.

Register to control DDR3A PLL.

Table 5-179. BOOTCFG_DDR3A_PLL_CTL1 Instances

Instance	Physical Address
BOOT_CFG	0262 0364h

Figure 5-57. BOOTCFG_DDR3A_PLL_CTL1 Register

31	30	29	28	27	26	25	24
DDR3A_PHY_RST	RESERVED						
R/W-0h	R-						
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED	PLLRST	RESERVED					
R-0h	R/W-0h	R-0h					
7	6	5	4	3	2	1	0
RESERVED	ENSAT	RESERVED		BWADJ			
R-0h	R/W-1h	R-0h		R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-180. BOOTCFG_DDR3A_PLL_CTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DDR3A_PHY_RST	R/W	0h	DDR3 PHY reset control 0 : Deassert MMR based reset control to DDR3 PHY. This allows the reset to be controlled by PSC 1 : Assert MMR based reset control to DDR3 PHY. This effectively asserts DDR3 PHY reset. For more information, see Section 7.2.3, *EMIF Integration*.
30-15	RESERVED	R		Reserved
14	PLLRST	R/W	0h	PLL reset bit 0 - PLL reset is released 1 - PLL reset is asserted
13-7	RESERVED	R	0h	Reserved
6	ENSAT	R/W	1h	Enables saturation behavior, needs to be set to 1 for proper PLL operation
5-4	RESERVED	R	0h	Reserved
3-0	BWADJ	R/W	0h	4-bit MSB of a 12-bit BWADJ field that selects the values for the loop bandwidth adjustment. BWADJ[7-0] is controlled from BOOTCFG_DDR3A_PLL_CTL0 register and BWADJ[11-8] is controlled from this register. BWADJ[11-0] should be programmed to a value equal to half of PLLM[12-0] value (round down if PLLM has an odd value). Example: If PLLM = 15, then BWADJ = 7

Table 5-181. Register Call Summary for BOOTCFG_DDR3A_PLL_CTL1

BOOT_CFG Functional Description • PLL Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_DDR3A_PLL_CTL0 Register (Offset = 360h) [reset = 98804C0h]: [0] • BOOT_CFG Registers: [0] • BOOTCFG_DDR3A_PLL_CTL1 Register (Offset = 364h) [reset = 40h]: [0]

5.1.4.53 BOOTCFG_ARM_PLL_CTL0 Register (Offset = 370h) [reset = 98804C0h]

BOOTCFG_ARM_PLL_CTL0 is shown in Figure 5-58 and described in Table 5-183.

Register to control ARM PLL.

Table 5-182. BOOTCFG_ARM_PLL_CTL0 Instances

Instance	Physical Address
BOOT_CFG	0262 0370h

Figure 5-58. BOOTCFG_ARM_PLL_CTL0 Register

31	30	29	28	27	26	25	24
BWADJ							
R/W-9h							
23	22	21	20	19	18	17	16
BYPASS	CLKOD				PLLM		
R/W-1h	R/W-1h				R/W-13h		
15	14	13	12	11	10	9	8
PLLM							
R/W-13h							
7	6	5	4	3	2	1	0
PLLM			PLLD				
R/W-13h			R/W-0h				

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-183. BOOTCFG_ARM_PLL_CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	BWADJ	R/W	9h	8-bit LSB of a 12-bit BWADJ field that selects the values for the loop bandwidth adjustment. BWADJ[7-0] is controlled from this register and BWADJ[11-8] is controlled from BOOTCFG_ARM_PLL_CTL1 register. BWADJ[11-0] should be programmed to a value equal to half of PLLM[12-0] value (round down if PLLM has an odd value). Example: If PLLM = 15, then BWADJ = 7
23	BYPASS	R/W	1h	Enable bypass mode 0 - Bypass disabled 1 - Bypass enabled
22-19	CLKOD	R/W	1h	PLL output divider (post VCO), supported value = 1 (i.e. divide-by-2)
18-6	PLLM	R/W	13h	A 13-bit field that selects the values for the multiplication factor. PLLM field is loaded with the multiply factor minus 1
5-0	PLLD	R/W	0h	A 6-bit field that selects the values for the reference divider. PLLD field is loaded with reference divide value minus 1

Table 5-184. Register Call Summary for BOOTCFG_ARM_PLL_CTL0

BOOT_CFG Functional Description • PLL Control Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_ARM_PLL_CTL0 Register (Offset = 370h) [reset = 98804C0h]: [0] • BOOTCFG_ARM_PLL_CTL1 Register (Offset = 374h) [reset = 40h]: [0]

5.1.4.54 BOOTCFG_ARM_PLL_CTL1 Register (Offset = 374h) [reset = 40h]

BOOTCFG_ARM_PLL_CTL1 is shown in Figure 5-59 and described in Table 5-186.

Register to control ARM PLL.

Table 5-185. BOOTCFG_ARM_PLL_CTL1 Instances

Instance	Physical Address
BOOT_CFG	0262 0374h

Figure 5-59. BOOTCFG_ARM_PLL_CTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED	PLLRST	RESERVED					
R-0h	R/W-0h	R-0h					
7	6	5	4	3	2	1	0
RESERVED	ENSAT	RESERVED		BWADJ			
R-0h	R/W-1h	R-0h		R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-186. BOOTCFG_ARM_PLL_CTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R		Reserved
14	PLLRST	R/W	0h	PLL reset bit 0 - PLL reset is released 1 - PLL reset is asserted
13-7	RESERVED	R	0h	Reserved
6	ENSAT	R/W	1h	Enables saturation behavior, needs to be set to 1 for proper PLL operation
5-4	RESERVED	R	0h	Enables fast locking circuit
3-0	BWADJ	R/W	0h	4-bit MSB of a 12-bit BWADJ field that selects the values for the loop bandwidth adjustment. BWADJ[7-0] is controlled from BOOTCFG_ARM_PLL_CTL0 register and BWADJ[11-8] is controlled from this register. BWADJ[11-0] should be programmed to a value equal to half of PLLM[12-0] value (round down if PLLM has an odd value). Example: If PLLM = 15, then BWADJ = 7

Table 5-187. Register Call Summary for BOOTCFG_ARM_PLL_CTL1

BOOT_CFG Functional Description • PLL Control Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_ARM_PLL_CTL0 Register (Offset = 370h) [reset = 98804C0h]: [0] • BOOTCFG_ARM_PLL_CTL1 Register (Offset = 374h) [reset = 40h]: [0]

5.1.4.55 BOOTCFG_DSS_PLL_CTL0 Register (Offset = 380h) [reset = 98804C0h]

BOOTCFG_DSS_PLL_CTL0 is shown in Figure 5-60 and described in Table 5-189.

Register to control DSS PLL.

Table 5-188. BOOTCFG_DSS_PLL_CTL0 Instances

Instance	Physical Address
BOOT_CFG	0262 0380h

Figure 5-60. BOOTCFG_DSS_PLL_CTL0 Register

31	30	29	28	27	26	25	24
BWADJ							
R/W-9h							
23	22	21	20	19	18	17	16
BYPASS	CLKOD				PLLM		
R/W-1h	R/W-1h				R/W-13h		
15	14	13	12	11	10	9	8
PLLM							
R/W-13h							
7	6	5	4	3	2	1	0
PLLM				PLLD			
R/W-13h				R/W-0h			

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-189. BOOTCFG_DSS_PLL_CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	BWADJ	R/W	9h	8-bit LSB of a 12-bit BWADJ field that selects the values for the loop bandwidth adjustment. BWADJ[7-0] is controlled from this register and BWADJ[11-8] is controlled from BOOTCFG_DSS_PLL_CTL1 register. BWADJ[11-0] should be programmed to a value equal to half of PLLM[12-0] value (round down if PLLM has an odd value). Example: If PLLM = 15, then BWADJ = 7
23	BYPASS	R/W	1h	Enable bypass mode 0 - Bypass disabled 1 - Bypass enabled
22-19	CLKOD	R/W	1h	PLL output divider (post VCO), supported value = 1 (i.e. divide-by-2)
18-6	PLLM	R/W	13h	A 13-bit field that selects the values for the multiplication factor. PLLM field is loaded with the multiply factor minus 1
5-0	PLLD	R/W	0h	A 6-bit field that selects the values for the reference divider. PLLD field is loaded with reference divide value minus 1

Table 5-190. Register Call Summary for BOOTCFG_DSS_PLL_CTL0

BOOT_CFG Functional Description • PLL Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_DSS_PLL_CTL0 Register (Offset = 380h) [reset = 98804C0h]: [0] • BOOTCFG_DSS_PLL_CTL1 Register (Offset = 384h) [reset = 40h]: [0] • BOOT_CFG Registers: [0]

5.1.4.56 BOOTCFG_DSS_PLL_CTL1 Register (Offset = 384h) [reset = 40h]

BOOTCFG_DSS_PLL_CTL1 is shown in Figure 5-61 and described in Table 5-192.

Register to control DSS PLL.

Table 5-191. BOOTCFG_DSS_PLL_CTL1 Instances

Instance	Physical Address
BOOT_CFG	0262 0384h

Figure 5-61. BOOTCFG_DSS_PLL_CTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED	PLLRST	RESERVED					
R-0h	R/W-0h	R-0h					
7	6	5	4	3	2	1	0
RESERVED	ENSAT	RESERVED		BWADJ			
R-0h	R/W-1h	R-0h		R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-192. BOOTCFG_DSS_PLL_CTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R		Reserved
14	PLLRST	R/W	0h	PLL reset bit 0 - PLL reset is released 1 - PLL reset is asserted
13-7	RESERVED	R	0h	Reserved
6	ENSAT	R/W	1h	Enables saturation behavior, needs to be set to 1 for proper PLL operation
5-4	RESERVED	R	0h	Reserved
3-0	BWADJ	R/W	0h	4-bit MSB of a 12-bit BWADJ field that selects the values for the loop bandwidth adjustment. BWADJ[7-0] is controlled from BOOTCFG_DSS_PLL_CTL0 register and BWADJ[11-8] is controlled from this register. BWADJ[11-0] should be programmed to a value equal to half of PLLM[12-0] value (round down if PLLM has an odd value). Example: If PLLM = 15, then BWADJ = 7

Table 5-193. Register Call Summary for BOOTCFG_DSS_PLL_CTL1

BOOT_CFG Functional Description • PLL Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_DSS_PLL_CTL0 Register (Offset = 380h) [reset = 98804C0h]: [0] • BOOTCFG_DSS_PLL_CTL1 Register (Offset = 384h) [reset = 40h]: [0] • BOOT_CFG Registers: [0]

5.1.4.57 BOOTCFG_ICSS_PLL_CTL0 Register (Offset = 388h) [reset = 98804C0h]

BOOTCFG_ICSS_PLL_CTL0 is shown in Figure 5-62 and described in Table 5-195.

Register to control ICSS PLL.

Table 5-194. BOOTCFG_ICSS_PLL_CTL0 Instances

Instance	Physical Address
BOOT_CFG	0262 0388h

Figure 5-62. BOOTCFG_ICSS_PLL_CTL0 Register

31	30	29	28	27	26	25	24
BWADJ							
R/W-9h							
23	22	21	20	19	18	17	16
BYPASS	CLKOD				PLLM		
R/W-1h	R/W-1h				R/W-13h		
15	14	13	12	11	10	9	8
PLLM							
R/W-13h							
7	6	5	4	3	2	1	0
PLLM			PLLD				
R/W-13h			R/W-0h				

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-195. BOOTCFG_ICSS_PLL_CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	BWADJ	R/W	9h	8-bit LSB of a 12-bit BWADJ field that selects the values for the loop bandwidth adjustment. BWADJ[7-0] is controlled from this register and BWADJ[11-8] is controlled from BOOTCFG_ICSS_PLL_CTL1 register. BWADJ[11-0] should be programmed to a value equal to half of PLLM[12-0] value (round down if PLLM has an odd value). Example: If PLLM = 15, then BWADJ = 7
23	BYPASS	R/W	1h	Enable bypass mode 0 - Bypass disabled 1 - Bypass enabled
22-19	CLKOD	R/W	1h	PLL output divider (post VCO), supported value = 1 (i.e. divide-by-2)
18-6	PLLM	R/W	13h	A 13-bit field that selects the values for the multiplication factor. PLLM field is loaded with the multiply factor minus 1
5-0	PLLD	R/W	0h	A 6-bit field that selects the values for the reference divider. PLLD field is loaded with reference divide value minus 1

Table 5-196. Register Call Summary for BOOTCFG_ICSS_PLL_CTL0

BOOT_CFG Functional Description • PLL Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_ICSS_PLL_CTL1 Register (Offset = 38Ch) [reset = 40h]: [0] • BOOT_CFG Registers: [0] • BOOTCFG_ICSS_PLL_CTL0 Register (Offset = 388h) [reset = 98804C0h]: [0]

5.1.4.58 BOOTCFG_ICSS_PLL_CTL1 Register (Offset = 38Ch) [reset = 40h]

BOOTCFG_ICSS_PLL_CTL1 is shown in Figure 5-63 and described in Table 5-198.

Register to control ICSS PLL.

Table 5-197. BOOTCFG_ICSS_PLL_CTL1 Instances

Instance	Physical Address
BOOT_CFG	0262 038Ch

Figure 5-63. BOOTCFG_ICSS_PLL_CTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED	PLLRST	RESERVED					
R-0h	R/W-0h	R-0h					
7	6	5	4	3	2	1	0
RESERVED	ENSAT	RESERVED		BWADJ			
R-0h	R/W-1h	R-0h		R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-198. BOOTCFG_ICSS_PLL_CTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R		Reserved
14	PLLRST	R/W	0h	PLL reset bit 0 - PLL reset is released 1 - PLL reset is asserted
13-7	RESERVED	R	0h	Reserved
6	ENSAT	R/W	1h	Enables saturation behavior, needs to be set to 1 for proper PLL operation
5-4	RESERVED	R	0h	Reserved
3-0	BWADJ	R/W	0h	4-bit MSB of a 12-bit BWADJ field that selects the values for the loop bandwidth adjustment. BWADJ[7-0] is controlled from BOOTCFG_ICSS_PLL_CTL0 register and BWADJ[11-8] is controlled from this register. BWADJ[11-0] should be programmed to a value equal to half of PLLM[12-0] value (round down if PLLM has an odd value). Example: If PLLM = 15, then BWADJ = 7

Table 5-199. Register Call Summary for BOOTCFG_ICSS_PLL_CTL1

BOOT_CFG Functional Description • PLL Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_ICSS_PLL_CTL1 Register (Offset = 38Ch) [reset = 40h]: [0] • BOOT_CFG Registers: [0] • BOOTCFG_ICSS_PLL_CTL0 Register (Offset = 388h) [reset = 98804C0h]: [0]

5.1.4.59 BOOTCFG_UART_PLL_CTL0 Register (Offset = 390h) [reset = 98804C0h]

BOOTCFG_UART_PLL_CTL0 is shown in Figure 5-64 and described in Table 5-201.

Register to control UART PLL.

Table 5-200. BOOTCFG_UART_PLL_CTL0 Instances

Instance	Physical Address
BOOT_CFG	0262 0390h

Figure 5-64. BOOTCFG_UART_PLL_CTL0 Register

31	30	29	28	27	26	25	24
BWADJ							
R/W-9h							
23	22	21	20	19	18	17	16
BYPASS	CLKOD				PLLM		
R/W-1h	R/W-1h				R/W-13h		
15	14	13	12	11	10	9	8
PLLM							
R/W-13h							
7	6	5	4	3	2	1	0
PLLM			PLLD				
R/W-13h			R/W-0h				

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-201. BOOTCFG_UART_PLL_CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	BWADJ	R/W	9h	8-bit LSB of a 12-bit BWADJ field that selects the values for the loop bandwidth adjustment. BWADJ[7-0] is controlled from this register and BWADJ[11-8] is controlled from BOOTCFG_UART_PLL_CTL1 register. BWADJ[11-0] should be programmed to a value equal to half of PLLM[12-0] value (round down if PLLM has an odd value). Example: If PLLM = 15, then BWADJ = 7
23	BYPASS	R/W	1h	Enable bypass mode 0 - Bypass disabled 1 - Bypass enabled
22-19	CLKOD	R/W	1h	PLL output divider (post VCO), supported value = 1 (i.e. divide-by-2)
18-6	PLLM	R/W	13h	A 13-bit field that selects the values for the multiplication factor. PLLM field is loaded with the multiply factor minus 1
5-0	PLLD	R/W	0h	A 6-bit field that selects the values for the reference divider. PLLD field is loaded with reference divide value minus 1

Table 5-202. Register Call Summary for BOOTCFG_UART_PLL_CTL0

BOOT_CFG Functional Description • PLL Control Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_UART_PLL_CTL1 Register (Offset = 394h) [reset = 40h]: [0] • BOOTCFG_UART_PLL_CTL0 Register (Offset = 390h) [reset = 98804C0h]: [0]

5.1.4.60 BOOTCFG_UART_PLL_CTL1 Register (Offset = 394h) [reset = 40h]

BOOTCFG_UART_PLL_CTL1 is shown in Figure 5-65 and described in Table 5-204.

Register to control UART PLL.

Table 5-203. BOOTCFG_UART_PLL_CTL1 Instances

Instance	Physical Address
BOOT_CFG	0262 0394h

Figure 5-65. BOOTCFG_UART_PLL_CTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED	PLLRST	RESERVED					
R-0h	R/W-0h	R-0h					
7	6	5	4	3	2	1	0
RESERVED	ENSAT	RESERVED		BWADJ			
R-0h	R/W-1h	R-0h		R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-204. BOOTCFG_UART_PLL_CTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R		Reserved
14	PLLRST	R/W	0h	PLL reset bit 0 - PLL reset is released 1 - PLL reset is asserted
13-7	RESERVED	R	0h	Reserved
6	ENSAT	R/W	1h	Enables saturation behavior, needs to be set to 1 for proper PLL operation
5-4	RESERVED	R	0h	Reserved
3-0	BWADJ	R/W	0h	4-bit MSB of a 12-bit BWADJ field that selects the values for the loop bandwidth adjustment. BWADJ[7-0] is controlled from BOOTCFG_UART_PLL_CTL0 register and BWADJ[11-8] is controlled from this register. BWADJ[11-0] should be programmed to a value equal to half of PLLM[12-0] value (round down if PLLM has an odd value.) Example: If PLLM = 15, then BWADJ = 7

Table 5-205. Register Call Summary for BOOTCFG_UART_PLL_CTL1

BOOT_CFG Functional Description • PLL Control Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_UART_PLL_CTL1 Register (Offset = 394h) [reset = 40h]: [0] • BOOTCFG_UART_PLL_CTL0 Register (Offset = 390h) [reset = 98804C0h]: [0]

5.1.4.61 BOOTCFG_ARMENDIAN_CFGx_0 Register (Offset = 400h + x*10h) [reset = See Table 5-18]

BOOTCFG_ARMENDIAN_CFGx_0 (*where x = 0 to 7*) is shown in Figure 5-66 and described in Table 5-207.

These registers control the way A15 core access to peripheral MMRs in region *x* shown up in the A15 processor registers. The purpose is to provide an endian-invariant view of the peripheral MMRs when performing a 32-bit access.

Table 5-206. BOOTCFG_ARMENDIAN_CFGx_0 Instances

Instance	Physical Address
BOOT_CFG	0262 0400h + (x*10h)

Figure 5-66. BOOTCFG_ARMENDIAN_CFGx_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARMENDIAN_CFGX_ADDR																								RESERVED							
R/W-See Table 5-18																								R-							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-207. BOOTCFG_ARMENDIAN_CFGx_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	ARMENDIAN_CFGX_ADDR	R/W	See Table 5-18	24-bit Base Address of Configuration Region *x* This base address defines the start of a contiguous block of Memory Mapped Register space for which a word swap is done by the Arm CorePac bridge
7-0	RESERVED	R		Reserved

Table 5-208. Register Call Summary for BOOTCFG_ARMENDIAN_CFGx_0

BOOT_CFG Functional Description • ARMSS Endian Configuration Registers: [0]
BOOT_CFG Registers • BOOTCFG_ARMENDIAN_CFGx_0 Register (Offset = 400h + x*10h) [reset = See]: [0] • BOOT_CFG Registers: [0]

5.1.4.62 BOOTCFG_ARMENDIAN_CFGx_1 Register (Offset = 404h + x*10h) [reset = See Table 5-18]

BOOTCFG_ARMENDIAN_CFGx_1 (*where x = 0 to 7*) is shown in Figure 5-67 and described in Table 5-210.

These registers control the way A15 core access to peripheral MMRs in region *x* shown up in the A15 processor registers. The purpose is to provide an endian-invariant view of the peripheral MMRs when performing a 32-bit access.

Table 5-209. BOOTCFG_ARMENDIAN_CFGx_1 Instances

Instance	Physical Address
BOOT_CFG	0262 0404h + (x*10h)

Figure 5-67. BOOTCFG_ARMENDIAN_CFGx_1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED				ARMENDIAN_CFGX_SIZE			
R-				R/W-See Table 5-18			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-210. BOOTCFG_ARMENDIAN_CFGx_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R		Reserved
3-0	ARMENDIAN_CFGX_SIZE	R/W	See Table 5-18	4-bit encoded size of Configuration Region *x* The value in the SIZE field defines the size of the contiguous block of Memory Mapped Register space for which a word swap is done by the Arm CorePac bridge (starting from base address of the region) 0h = 64KB 1h = 128KB 2h = 256KB 3h = 512KB 4h = 1MB 5h = 2MB 6h = 4MB 7h = 8MB 8h = 16MB 9h = 32MB Ah = 64MB Bh = 128MB

Table 5-211. Register Call Summary for BOOTCFG_ARMENDIAN_CFGx_1

BOOT_CFG Functional Description • ARMSS Endian Configuration Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_ARMENDIAN_CFGx_1 Register (Offset = 404h + x*10h) [reset = See]: [0]

5.1.4.63 BOOTCFG_ARMENDIAN_CFGx_2 Register (Offset = 408h + x*10h) [reset = See Table 5-18]

BOOTCFG_ARMENDIAN_CFGx_2 (*where x = 0 to 7*) is shown in Figure 5-68 and described in Table 5-213.

These registers control the way A15 core access to peripheral MMRs in region *x* shown up in the A15 processor registers. The purpose is to provide an endian-invariant view of the peripheral MMRs when performing a 32-bit access.

Table 5-212. BOOTCFG_ARMENDIAN_CFGx_2 Instances

Instance	Physical Address
BOOT_CFG	0262 0408h + (x*10h)

Figure 5-68. BOOTCFG_ARMENDIAN_CFGx_2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED							ARMENDIAN_CFGX_DIS
R-							R/W-See Table 5-18

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-213. BOOTCFG_ARMENDIAN_CFGx_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R		Reserved
0	ARMENDIAN_CFGX_DIS	R/W	See Table 5-18	Disabling the word swap of Configuration Region *x* 0 - Enable word swap for region 1 - Disable word swap for region

Table 5-214. Register Call Summary for BOOTCFG_ARMENDIAN_CFGx_2

BOOT_CFG Functional Description • ARMSS Endian Configuration Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_ARMENDIAN_CFGx_2 Register (Offset = 408h + x*10h) [reset = See]: [0]

5.1.4.64 BOOTCFG_ARMTBR_TRBx_W0 Register (Offset = 480h + x*10h) [reset = See Table 5-20]

BOOTCFG_ARMTBR_TRBx_W0 (*where x = 0 to 2*) is shown in Figure 5-69 and described in Table 5-216.

These registers hold the software writable copy of USB xHCI Transfer Request Block (TRB) *x* for a trace buffer. Each TRB consists of 4 consecutive 32-bit words (0..3) defining the buffer address, buffer size and other control, as per xHCI definition. There are three TRBs (0..2) used for continuously draining the ARMSS trace buffer.

Table 5-215. BOOTCFG_ARMTBR_TRBx_W0 Instances

Instance	Physical Address
BOOT_CFG	0262 0480h + (x*10h)

Figure 5-69. BOOTCFG_ARMTBR_TRBx_W0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARMTBR_TRBX_W0																															
R/W-See Table 5-20																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-216. BOOTCFG_ARMTBR_TRBx_W0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ARMTBR_TRBX_W0	R/W	See Table 5-20	

Table 5-217. Register Call Summary for BOOTCFG_ARMTBR_TRBx_W0

BOOT_CFG Functional Description • Registers Associated with the ARMSS and DEBUG_SS Trace Buffers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_ARMTBR_TRBx_W0 Register (Offset = 480h + x*10h) [reset = See]: [0]

5.1.4.65 BOOTCFG_ARMTBR_TRBx_W1 Register (Offset = 484h + x*10h) [reset = See Table 5-20]

BOOTCFG_ARMTBR_TRBx_W1 (*where x = 0 to 2*) is shown in Figure 5-70 and described in Table 5-219.

These registers hold the software writable copy of USB xHCI Transfer Request Block (TRB) *x* for a trace buffer. Each TRB consists of 4 consecutive 32-bit words (0..3) defining the buffer address, buffer size and other control, as per xHCI definition. There are three TRBs (0..2) used for continuously draining the ARMSS trace buffer.

Table 5-218. BOOTCFG_ARMTBR_TRBx_W1 Instances

Instance	Physical Address
BOOT_CFG	0262 0484h + (x*10h)

Figure 5-70. BOOTCFG_ARMTBR_TRBx_W1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARMTBR_TRBX_W1																															
R/W-See Table 5-20																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-219. BOOTCFG_ARMTBR_TRBx_W1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ARMTBR_TRBX_W1	R/W	See Table 5-20	

Table 5-220. Register Call Summary for BOOTCFG_ARMTBR_TRBx_W1

BOOT_CFG Functional Description • Registers Associated with the ARMSS and DEBUG_SS Trace Buffers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_ARMTBR_TRBx_W1 Register (Offset = 484h + x*10h) [reset = See]: [0]

5.1.4.66 BOOTCFG_ARMTBR_TRBx_W2 Register (Offset = 488h + x*10h) [reset = See Table 5-20]

BOOTCFG_ARMTBR_TRBx_W2 (*where x = 0 to 2*) is shown in Figure 5-71 and described in Table 5-222.

These registers hold the software writable copy of USB xHCI Transfer Request Block (TRB) *x* for a trace buffer. Each TRB consists of 4 consecutive 32-bit words (0..3) defining the buffer address, buffer size and other control, as per xHCI definition. There are three TRBs (0..2) used for continuously draining the ARMSS trace buffer.

Table 5-221. BOOTCFG_ARMTBR_TRBx_W2 Instances

Instance	Physical Address
BOOT_CFG	0262 0488h + (x*10h)

Figure 5-71. BOOTCFG_ARMTBR_TRBx_W2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARMTBR_TRBX_W2																															
R/W-See Table 5-20																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-222. BOOTCFG_ARMTBR_TRBx_W2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ARMTBR_TRBX_W2	R/W	See Table 5-20	

Table 5-223. Register Call Summary for BOOTCFG_ARMTBR_TRBx_W2

BOOT_CFG Functional Description • Registers Associated with the ARMSS and DEBUG_SS Trace Buffers: [0]
BOOT_CFG Registers • BOOTCFG_ARMTBR_TRBx_W2 Register (Offset = 488h + x*10h) [reset = See]: [0] • BOOT_CFG Registers: [0]

5.1.4.67 BOOTCFG_ARMTBR_TRBx_W3 Register (Offset = 48Ch + x*10h) [reset = See Table 5-20]

BOOTCFG_ARMTBR_TRBx_W3 (*where x = 0 to 2*) is shown in Figure 5-72 and described in Table 5-225.

These registers hold the software writable copy of USB xHCI Transfer Request Block (TRB) *x* for a trace buffer. Each TRB consists of 4 consecutive 32-bit words (0..3) defining the buffer address, buffer size and other control, as per xHCI definition. There are three TRBs (0..2) used for continuously draining the ARMSS trace buffer.

Table 5-224. BOOTCFG_ARMTBR_TRBx_W3 Instances

Instance	Physical Address
BOOT_CFG	0262 048Ch + (x*10h)

Figure 5-72. BOOTCFG_ARMTBR_TRBx_W3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARMTBR_TRBX_W3																															
R/W-See Table 5-20																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-225. BOOTCFG_ARMTBR_TRBx_W3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ARMTBR_TRBX_W3	R/W	See Table 5-20	

Table 5-226. Register Call Summary for BOOTCFG_ARMTBR_TRBx_W3

BOOT_CFG Functional Description • Registers Associated with the ARMSS and DEBUG_SS Trace Buffers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_ARMTBR_TRBx_W3 Register (Offset = 48Ch + x*10h) [reset = See]: [0]

5.1.4.68 BOOTCFG_ARMTBR_SHDW_TRBx_W0 Register (Offset = 4C0h + x*10h) [reset = See]

BOOTCFG_ARMTBR_SHDW_TRBx_W0 (*where x = 0 to 2*) is shown in Figure 5-73 and described in Table 5-228.

These are shadow registers for Transfer Request Block (TRB) *x* and shall reflect the same value as ARMTBR_TRBx_W[3:0] registers. These registers per trace buffer holds the USB readable versions of the descriptor.These registers are expected to be written by the USB module, but writes to these registers shall be ignored silently, without returning any error in the config bus. The purpose of the shadow registers are to enable USB module to use these descriptors continuously without requiring software intervention to re-initialize the Transfer Request Blocks (TRB).

Table 5-227. BOOTCFG_ARMTBR_SHDW_TRBx_W0 Instances

Instance	Physical Address
BOOT_CFG	0262 04C0h + (x*10h)

Figure 5-73. BOOTCFG_ARMTBR_SHDW_TRBx_W0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARMTBR_SHDW_TRBX_W0																															
R-See Table 5-20																															

LEGEND: R = Read Only; *-n* = value after reset

Table 5-228. BOOTCFG_ARMTBR_SHDW_TRBx_W0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ARMTBR_SHDW_TRBX_W0	R	See Table 5-20	

Table 5-229. Register Call Summary for BOOTCFG_ARMTBR_SHDW_TRBx_W0

BOOT_CFG Functional Description • Registers Associated with the ARMSS and DEBUG_SS Trace Buffers: [0]
BOOT_CFG Registers • BOOTCFG_ARMTBR_SHDW_TRBx_W0 Register (Offset = 4C0h + x*10h) [reset = See]: [0] • BOOT_CFG Registers: [0]

5.1.4.69 BOOTCFG_ARMTBR_SHDW_TRBx_W1 Register (Offset = 4C4h + x*10h) [reset = See]

BOOTCFG_ARMTBR_SHDW_TRBx_W1 (*where x = 0 to 2*) is shown in Figure 5-74 and described in Table 5-231.

These are shadow registers for Transfer Request Block (TRB) *x* and shall reflect the same value as ARMTBR_TRBx_W[3:0] registers. These registers per trace buffer holds the USB readable versions of the descriptor.These registers are expected to be written by the USB module, but writes to these registers shall be ignored silently, without returning any error in the config bus. The purpose of the shadow registers are to enable USB module to use these descriptors continuously without requiring software intervention to re-initialize the Transfer Request Blocks (TRB).

Table 5-230. BOOTCFG_ARMTBR_SHDW_TRBx_W1 Instances

Instance	Physical Address
BOOT_CFG	0262 04C4h + (x*10h)

Figure 5-74. BOOTCFG_ARMTBR_SHDW_TRBx_W1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARMTBR_SHDW_TRBX_W1																															
R-See Table 5-20																															

LEGEND: R = Read Only; *-n* = value after reset

Table 5-231. BOOTCFG_ARMTBR_SHDW_TRBx_W1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ARMTBR_SHDW_TRBX_W1	R	See Table 5-20	

Table 5-232. Register Call Summary for BOOTCFG_ARMTBR_SHDW_TRBx_W1

BOOT_CFG Functional Description • Registers Associated with the ARMSS and DEBUG_SS Trace Buffers: [0]
BOOT_CFG Registers • BOOTCFG_ARMTBR_SHDW_TRBx_W1 Register (Offset = 4C4h + x*10h) [reset = See]: [0] • BOOT_CFG Registers: [0]

5.1.4.70 BOOTCFG_ARMTBR_SHDW_TRBx_W2 Register (Offset = 4C8h + x*10h) [reset = See]

BOOTCFG_ARMTBR_SHDW_TRBx_W2 (*where x = 0 to 2*) is shown in Figure 5-75 and described in Table 5-234.

These are shadow registers for Transfer Request Block (TRB) *x* and shall reflect the same value as ARMTBR_TRBx_W[3:0] registers. These registers per trace buffer holds the USB readable versions of the descriptor.These registers are expected to be written by the USB module, but writes to these registers shall be ignored silently, without returning any error in the config bus. The purpose of the shadow registers are to enable USB module to use these descriptors continuously without requiring software intervention to re-initialize the Transfer Request Blocks (TRB).

Table 5-233. BOOTCFG_ARMTBR_SHDW_TRBx_W2 Instances

Instance	Physical Address
BOOT_CFG	0262 04C8h + (x*10h)

Figure 5-75. BOOTCFG_ARMTBR_SHDW_TRBx_W2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARMTBR_SHDW_TRBX_W2																															
R-See Table 5-20																															

LEGEND: R = Read Only; *-n* = value after reset

Table 5-234. BOOTCFG_ARMTBR_SHDW_TRBx_W2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ARMTBR_SHDW_TRBX_W2	R	See Table 5-20	

Table 5-235. Register Call Summary for BOOTCFG_ARMTBR_SHDW_TRBx_W2

BOOT_CFG Functional Description • Registers Associated with the ARMSS and DEBUG_SS Trace Buffers: [0]
BOOT_CFG Registers • BOOTCFG_ARMTBR_SHDW_TRBx_W2 Register (Offset = 4C8h + x*10h) [reset = See]: [0] • BOOT_CFG Registers: [0]

5.1.4.71 BOOTCFG_ARMTBR_SHDW_TRBx_W3 Register (Offset = 4CCh + x*10h) [reset = See]

BOOTCFG_ARMTBR_SHDW_TRBx_W3 (*where x = 0 to 2*) is shown in Figure 5-76 and described in Table 5-237.

These are shadow registers for Transfer Request Block (TRB) *x* and shall reflect the same value as ARMTBR_TRBx_W[3:0] registers. These registers per trace buffer holds the USB readable versions of the descriptor.These registers are expected to be written by the USB module, but writes to these registers shall be ignored silently, without returning any error in the config bus. The purpose of the shadow registers are to enable USB module to use these descriptors continuously without requiring software intervention to re-initialize the Transfer Request Blocks (TRB).

Table 5-236. BOOTCFG_ARMTBR_SHDW_TRBx_W3 Instances

Instance	Physical Address
BOOT_CFG	0262 04CCh + (x*10h)

Figure 5-76. BOOTCFG_ARMTBR_SHDW_TRBx_W3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARMTBR_SHDW_TRBX_W3																															
R-See Table 5-20																															

LEGEND: R = Read Only; *-n* = value after reset

Table 5-237. BOOTCFG_ARMTBR_SHDW_TRBx_W3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ARMTBR_SHDW_TRBX_W3	R	See Table 5-20	

Table 5-238. Register Call Summary for BOOTCFG_ARMTBR_SHDW_TRBx_W3

BOOT_CFG Functional Description • Registers Associated with the ARMSS and DEBUG_SS Trace Buffers: [0]
BOOT_CFG Registers • BOOTCFG_ARMTBR_SHDW_TRBx_W3 Register (Offset = 4CCh + x*10h) [reset = See]: [0] • BOOT_CFG Registers: [0]

5.1.4.72 BOOTCFG_DBGTBR_TRBx_W0 Register (Offset = 500h + x*10h) [reset = See]

BOOTCFG_DBGTBR_TRBx_W0 (*where x = 0 to 2*) is shown in Figure 5-77 and described in Table 5-240.

These registers hold the software writable copy of USB xHCI Transfer Request Block (TRB) *x* for a trace buffer. Each TRB consists of 4 consecutive 32-bit words (0..3) defining the buffer address, buffer size and other control, as per xHCI definition. There are three TRBs (0..2) used for continuously draining the Debug Subsystem trace buffer.

Table 5-239. BOOTCFG_DBGTBR_TRBx_W0 Instances

Instance	Physical Address
BOOT_CFG	0262 0500h + (x*10h)

Figure 5-77. BOOTCFG_DBGTBR_TRBx_W0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBGTBR_TRBX_W0																															
R/W-See Table 5-20																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-240. BOOTCFG_DBGTBR_TRBx_W0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBGTBR_TRBX_W0	R/W	See Table 5-20	

Table 5-241. Register Call Summary for BOOTCFG_DBGTBR_TRBx_W0

BOOT_CFG Functional Description • Registers Associated with the ARMSS and DEBUG_SS Trace Buffers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_DBGTBR_TRBx_W0 Register (Offset = 500h + x*10h) [reset = See]: [0]

5.1.4.73 BOOTCFG_DBGTBR_TRBx_W1 Register (Offset = 504h + x*10h) [reset = See]

BOOTCFG_DBGTBR_TRBx_W1 (*where x = 0 to 2*) is shown in Figure 5-78 and described in Table 5-243.

These registers hold the software writable copy of USB xHCI Transfer Request Block (TRB) *x* for a trace buffer. Each TRB consists of 4 consecutive 32-bit words (0..3) defining the buffer address, buffer size and other control, as per xHCI definition. There are three TRBs (0..2) used for continuously draining the Debug Subsystem trace buffer.

Table 5-242. BOOTCFG_DBGTBR_TRBx_W1 Instances

Instance	Physical Address
BOOT_CFG	0262 0504h + (x*10h)

Figure 5-78. BOOTCFG_DBGTBR_TRBx_W1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBGTBR_TRBX_W1																															
R/W-See Table 5-20																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-243. BOOTCFG_DBGTBR_TRBx_W1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBGTBR_TRBX_W1	R/W	See Table 5-20	

Table 5-244. Register Call Summary for BOOTCFG_DBGTBR_TRBx_W1

BOOT_CFG Functional Description • Registers Associated with the ARMSS and DEBUG_SS Trace Buffers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_DBGTBR_TRBx_W1 Register (Offset = 504h + x*10h) [reset = See]: [0]

5.1.4.74 BOOTCFG_DBGTBR_TRBx_W2 Register (Offset = 508h + x*10h) [reset = See]

BOOTCFG_DBGTBR_TRBx_W2 (*where x = 0 to 2*) is shown in Figure 5-79 and described in Table 5-246.

These registers hold the software writable copy of USB xHCI Transfer Request Block (TRB) *x* for a trace buffer. Each TRB consists of 4 consecutive 32-bit words (0..3) defining the buffer address, buffer size and other control, as per xHCI definition. There are three TRBs (0..2) used for continuously draining the Debug Subsystem trace buffer.

Table 5-245. BOOTCFG_DBGTBR_TRBx_W2 Instances

Instance	Physical Address
BOOT_CFG	0262 0508h + (x*10h)

Figure 5-79. BOOTCFG_DBGTBR_TRBx_W2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBGTBR_TRBX_W2																															
R/W-See Table 5-20																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-246. BOOTCFG_DBGTBR_TRBx_W2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBGTBR_TRBX_W2	R/W	See Table 5-20	

Table 5-247. Register Call Summary for BOOTCFG_DBGTBR_TRBx_W2

BOOT_CFG Functional Description • Registers Associated with the ARMSS and DEBUG_SS Trace Buffers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_DBGTBR_TRBx_W2 Register (Offset = 508h + x*10h) [reset = See]: [0]

5.1.4.75 BOOTCFG_DBGTBR_TRBx_W3 Register (Offset = 50Ch + x*10h) [reset = See]

BOOTCFG_DBGTBR_TRBx_W3 (*where x = 0 to 2*) is shown in Figure 5-80 and described in Table 5-249.

These registers hold the software writable copy of USB xHCI Transfer Request Block (TRB) *x* for a trace buffer. Each TRB consists of 4 consecutive 32-bit words (0..3) defining the buffer address, buffer size and other control, as per xHCI definition. There are three TRBs (0..2) used for continuously draining the Debug Subsystem trace buffer.

Table 5-248. BOOTCFG_DBGTBR_TRBx_W3 Instances

Instance	Physical Address
BOOT_CFG	0262 050Ch + (x*10h)

Figure 5-80. BOOTCFG_DBGTBR_TRBx_W3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BGTBR_SHDW_TRBX_W0																															
R/W-See Table 5-20																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-249. BOOTCFG_DBGTBR_TRBx_W3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BGTBR_SHDW_TRBX_W0	R/W	See Table 5-20	

Table 5-250. Register Call Summary for BOOTCFG_DBGTBR_TRBx_W3

BOOT_CFG Functional Description • Registers Associated with the ARMSS and DEBUG_SS Trace Buffers: [0]
BOOT_CFG Registers • BOOTCFG_DBGTBR_TRBx_W3 Register (Offset = 50Ch + x*10h) [reset = See]: [0] • BOOT_CFG Registers: [0]

5.1.4.76 BOOTCFG_DBGTBR_SHDW_TRBx_W0 Register (Offset = 540h + x*10h) [reset = See]

BOOTCFG_DBGTBR_SHDW_TRBx_W0 (*where x = 0 to 2*) is shown in Figure 5-81 and described in Table 5-252.

These are shadow registers for Transfer Request Block (TRB) *x* and shall reflect the same value as DBGTBR_TRBx_W[3:0] registers. These registers per trace buffer holds the USB readable versions of the descriptor. These registers are expected to be written by the USB module, but writes to these registers shall be ignored silently, without returning any error in the config bus. The purpose of the shadow registers are to enable USB module to use these descriptors continuously without requiring software intervention to re-initialize the Transfer Request Blocks (TRB).

Table 5-251. BOOTCFG_DBGTBR_SHDW_TRBx_W0 Instances

Instance	Physical Address
BOOT_CFG	0262 0540h + (x*10h)

Figure 5-81. BOOTCFG_DBGTBR_SHDW_TRBx_W0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBGTBR_SHDW_TRBX_W0																															
R-See Table 5-20																															

LEGEND: R = Read Only; *-n* = value after reset

Table 5-252. BOOTCFG_DBGTBR_SHDW_TRBx_W0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBGTBR_SHDW_TRBX_W0	R	See Table 5-20	

Table 5-253. Register Call Summary for BOOTCFG_DBGTBR_SHDW_TRBx_W0

BOOT_CFG Functional Description • Registers Associated with the ARMSS and DEBUG_SS Trace Buffers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_DBGTBR_SHDW_TRBx_W0 Register (Offset = 540h + x*10h) [reset = See]: [0]

5.1.4.77 BOOTCFG_DBGTBR_SHDW_TRBx_W1 Register (Offset = 544h + x*10h) [reset = See]

BOOTCFG_DBGTBR_SHDW_TRBx_W1 (*where x = 0 to 2*) is shown in Figure 5-82 and described in Table 5-255.

These are shadow registers for Transfer Request Block (TRB) *x* and shall reflect the same value as DBGTBR_TRBx_W[3:0] registers. These registers per trace buffer holds the USB readable versions of the descriptor.These registers are expected to be written by the USB module, but writes to these registers shall be ignored silently, without returning any error in the config bus. The purpose of the shadow registers are to enable USB module to use these descriptors continuously without requiring software intervention to re-initialize the Transfer Request Blocks (TRB).

Table 5-254. BOOTCFG_DBGTBR_SHDW_TRBx_W1 Instances

Instance	Physical Address
BOOT_CFG	0262 0544h + (x*10h)

Figure 5-82. BOOTCFG_DBGTBR_SHDW_TRBx_W1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBGTBR_SHDW_TRBX_W1																															
R-See Table 5-20																															

LEGEND: R = Read Only; *-n* = value after reset

Table 5-255. BOOTCFG_DBGTBR_SHDW_TRBx_W1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBGTBR_SHDW_TRBX_W1	R	See Table 5-20	

Table 5-256. Register Call Summary for BOOTCFG_DBGTBR_SHDW_TRBx_W1

BOOT_CFG Functional Description • Registers Associated with the ARMSS and DEBUG_SS Trace Buffers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_DBGTBR_SHDW_TRBx_W1 Register (Offset = 544h + x*10h) [reset = See]: [0]

5.1.4.78 BOOTCFG_DBGTBR_SHDW_TRBx_W2 Register (Offset = 548h + x*10h) [reset = See]

BOOTCFG_DBGTBR_SHDW_TRBx_W2 (*where x = 0 to 2*) is shown in Figure 5-83 and described in Table 5-258.

These are shadow registers for Transfer Request Block (TRB) *x* and shall reflect the same value as DBGTBR_TRBx_W[3:0] registers. These registers per trace buffer holds the USB readable versions of the descriptor.These registers are expected to be written by the USB module, but writes to these registers shall be ignored silently, without returning any error in the config bus. The purpose of the shadow registers are to enable USB module to use these descriptors continuously without requiring software intervention to re-initialize the Transfer Request Blocks (TRB).

Table 5-257. BOOTCFG_DBGTBR_SHDW_TRBx_W2 Instances

Instance	Physical Address
BOOT_CFG	0262 0548h + (x*10h)

Figure 5-83. BOOTCFG_DBGTBR_SHDW_TRBx_W2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBGTBR_SHDW_TRBX_W2																															
R-See Table 5-20																															

LEGEND: R = Read Only; *-n* = value after reset

Table 5-258. BOOTCFG_DBGTBR_SHDW_TRBx_W2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBGTBR_SHDW_TRBX_W2	R	See Table 5-20	

Table 5-259. Register Call Summary for BOOTCFG_DBGTBR_SHDW_TRBx_W2

BOOT_CFG Functional Description • Registers Associated with the ARMSS and DEBUG_SS Trace Buffers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_DBGTBR_SHDW_TRBx_W2 Register (Offset = 548h + x*10h) [reset = See]: [0]

5.1.4.79 BOOTCFG_DBGTBR_SHDW_TRBx_W3 Register (Offset = 54Ch + x*10h) [reset = See]

BOOTCFG_DBGTBR_SHDW_TRBx_W3 (*where x = 0 to 2*) is shown in Figure 5-84 and described in Table 5-261.

These are shadow registers for Transfer Request Block (TRB) *x* and shall reflect the same value as DBGTBR_TRBx_W[3:0] registers. These registers per trace buffer holds the USB readable versions of the descriptor.These registers are expected to be written by the USB module, but writes to these registers shall be ignored silently, without returning any error in the config bus. The purpose of the shadow registers are to enable USB module to use these descriptors continuously without requiring software intervention to re-initialize the Transfer Request Blocks(TRB).

Table 5-260. BOOTCFG_DBGTBR_SHDW_TRBx_W3 Instances

Instance	Physical Address
BOOT_CFG	0262 054Ch + (x*10h)

Figure 5-84. BOOTCFG_DBGTBR_SHDW_TRBx_W3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBGTBR_SHDW_TRBX_W3																															
R-See Table 5-20																															

LEGEND: R = Read Only; *-n* = value after reset

Table 5-261. BOOTCFG_DBGTBR_SHDW_TRBx_W3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBGTBR_SHDW_TRBX_W3	R	See Table 5-20	

Table 5-262. Register Call Summary for BOOTCFG_DBGTBR_SHDW_TRBx_W3

BOOT_CFG Functional Description • Registers Associated with the ARMSS and DEBUG_SS Trace Buffers: [0]
BOOT_CFG Registers • BOOTCFG_DBGTBR_SHDW_TRBx_W3 Register (Offset = 54Ch + x*10h) [reset = See]: [0] • BOOT_CFG Registers: [0]

5.1.4.80 BOOTCFG_SPARE1 Register (Offset = 654h) [reset = 0h]

BOOTCFG_SPARE1 is shown in Figure 5-85 and described in Table 5-264.

DSS Divider Control Register.

Table 5-263. BOOTCFG_SPARE1 Instances

Instance	Physical Address
BOOT_CFG	0262 0654h

Figure 5-85. BOOTCFG_SPARE1 Register

31	30	29	28	27	26	25	24
SPARE_PULSE1	SPARE_OUT1						
WPH-0h	R/W-0h						
23	22	21	20	19	18	17	16
SPARE_OUT1							
R/W-0h							
15	14	13	12	11	10	9	8
SPARE_OUT1							
R/W-0h							
7	6	5	4	3	2	1	0
SPARE_OUT1	DIV_FACTOR				DIV_CLKSEL	LOAD_DIV_RATIO	DIV_RSTN
R/W-0h	R/W-0h				R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; WPH = Write One to Pulse High;-*n* = value after reset

Table 5-264. BOOTCFG_SPARE1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SPARE_PULSE1	WPH	0h	When this bit is set to 1, it generates a pulse for one BOOT_CFG_VBUS_CLK clock period. Read return zero.
30-7	SPARE_OUT1	R/W	0h	Reserved.
6-3	DIV_FACTOR	R/W	0h	DSS divider value. Clock is divided by DIV_FACTOR+1.
2	DIV_CLKSEL	R/W	0h	DSS divided clock select. 0 - Use DSS clock 1 - Use divided DSS clock
1	LOAD_DIV_RATIO	R/W	0h	DSS divide value load. Pulse this bit by writing 1 followed by a 0 to load the DIV_FACTOR and start the divider.
0	DIV_RSTN	R/W	0h	DSS divider reset. Clear this bit to 0 before changing the DIV_FACTOR and set to 1 after DIV_FACTOR is programmed.

Table 5-265. Register Call Summary for BOOTCFG_SPARE1

BOOT_CFG Registers
- BOOT_CFG Registers: [0]
- BOOTCFG_SPARE1 Register (Offset = 654h) [reset = 0h]: [0]

5.1.4.81 BOOTCFG_DDR_CLKCTL Register (Offset = 690h) [reset = 0h]

BOOTCFG_DDR_CLKCTL is shown in Figure 5-86 and described in Table 5-267.

DDR PLL Clock Mux Control Register

Table 5-266. BOOTCFG_DDR_CLKCTL Instances

Instance	Physical Address
BOOT_CFG	0262 0690h

Figure 5-86. BOOTCFG_DDR_CLKCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							DDR_CLK_MUXSEL
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-267. BOOTCFG_DDR_CLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	DDR_CLK_MUXSEL	R/W	0h	DDR PLL Reference Clock Source 0 - Select HF Oscillator (when SYSCLKSEL pin is low) or external SYSCLK_P/N input (when SYSCLKSEL pin is high) 1 - Select external DDR_CLK_P/N input

Table 5-268. Register Call Summary for BOOTCFG_DDR_CLKCTL

BOOT_CFG Functional Description • Clock Muxing, Enabling and Division Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_DDR_CLKCTL Register (Offset = 690h) [reset = 0h]: [0]

5.1.4.82 BOOTCFG_ICSS_CLKCTL Register (Offset = 694h) [reset = 0h]

BOOTCFG_ICSS_CLKCTL is shown in Figure 5-87 and described in Table 5-270.

ICSS Clock Control Register

Table 5-269. BOOTCFG_ICSS_CLKCTL Instances

Instance	Physical Address
BOOT_CFG	0262 0694h

Figure 5-87. BOOTCFG_ICSS_CLKCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							ICSS1_PLL_MUXSEL
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED							ICSS0_PLL_MUXSEL
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-270. BOOTCFG_ICSS_CLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	ICSS1_PLL_MUXSEL	R/W	0h	ICSS1 Core Clock Source. 0 = Selects ICSS PLL output 1 = Selects NSS PLL output
7-1	RESERVED	R	0h	Reserved
0	ICSS0_PLL_MUXSEL	R/W	0h	ICSS0 Core Clock Source. 0 = Selects ICSS PLL output 1 = Selects NSS PLL output

Table 5-271. Register Call Summary for BOOTCFG_ICSS_CLKCTL

BOOT_CFG Functional Description • Clock Muxing, Enabling and Division Registers: [0]
BOOT_CFG Registers • BOOTCFG_ICSS_CLKCTL Register (Offset = 694h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.83 BOOTCFG_ETHERNET_CLKCTL Register (Offset = 698h) [reset = 0h]

BOOTCFG_ETHERNET_CLKCTL is shown in Figure 5-88 and described in Table 5-273.

Ethernet Clock Control Register

Table 5-272. BOOTCFG_ETHERNET_CLKCTL Instances

Instance	Physical Address
BOOT_CFG	0262 0698h

Figure 5-88. BOOTCFG_ETHERNET_CLKCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					RMII_MII_CLKOUT_EN	RMII_MII_CLKSEL	RESERVED
R-0h					RW-0h	RW-0h	R-0h

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 5-273. BOOTCFG_ETHERNET_CLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2	RMII_MII_CLKOUT_EN	RW	0h	BOOTROM code will set this bit based on the BOOTMODE selection 0 - CLKOUT pin is disabled 1 - CLKOUT pin is enabled
1	RMII_MII_CLKSEL	RW	0h	BOOTROM code will set this bit based on the BOOTMODE selection 0 - RMII_CLK will drive the CLKOUT pin 1 - MII_CLK will drive the CLKOUT pin
0	RESERVED	R	0h	Reserved

Table 5-274. Register Call Summary for BOOTCFG_ETHERNET_CLKCTL

BOOT_CFG Functional Description • Clock Muxing, Enabling and Division Registers: [0]
BOOT_CFG Registers • BOOTCFG_ETHERNET_CLKCTL Register (Offset = 698h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.84 BOOTCFG_USB0_CLKCTL Register (Offset = 69Ch) [reset = 0h]

BOOTCFG_USB0_CLKCTL is shown in Figure 5-89 and described in Table 5-276.

USB0 Clock Control Register

Table 5-275. BOOTCFG_USB0_CLKCTL Instances

Instance	Physical Address
BOOT_CFG	0262 069Ch

Figure 5-89. BOOTCFG_USB0_CLKCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			USB0_CLKCORE_DIV				
R-0h			RW-Fh				
7	6	5	4	3	2	1	0
RESERVED						USB0_CLKCORE_LOAD_DIV	USB0_CLKCORE_SEL
R-0h						RW-0h	RW-0h

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 5-276. BOOTCFG_USB0_CLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-8	USB0_CLKCORE_DIV	RW	Fh	USB0 Core Clock Divider Value. This field must be set to <divider value> - 1. For example, a value of Fh means that the clock is divided by 16.
7-2	RESERVED	R	0h	Reserved
1	USB0_CLKCORE_LOAD_DIV	RW	0h	USB0 Core Clock Load Divider Control 0 - Clear 1 - Load the clock divider value set in the USB0_CLKCORE_DIV bit field
0	USB0_CLKCORE_SEL	RW	0h	USB0 Core Clock Source 0 - Selects UART PLL output 1 - Selects NSS PLL output

Table 5-277. Register Call Summary for BOOTCFG_USB0_CLKCTL

BOOT_CFG Functional Description • Clock Muxing, Enabling and Division Registers: [0]
BOOT_CFG Registers • BOOTCFG_USB0_CLKCTL Register (Offset = 69Ch) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.85 BOOTCFG_USB1_CLKCTL Register (Offset = 6A0h) [reset = 0h]

BOOTCFG_USB1_CLKCTL is shown in Figure 5-90 and described in Table 5-279.

USB1 Clock Control Register

Table 5-278. BOOTCFG_USB1_CLKCTL Instances

Instance	Physical Address
BOOT_CFG	0262 06A0h

Figure 5-90. BOOTCFG_USB1_CLKCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED			USB1_CLKCORE_DIV				
R-			RW-Fh				
7	6	5	4	3	2	1	0
RESERVED						USB1_CLKCORE_LOAD_DIV	USB1_CLKCORE_SEL
R-						RW-0h	RW-0h

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 5-279. BOOTCFG_USB1_CLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R		Reserved
12-8	USB1_CLKCORE_DIV	RW	Fh	USB1 Core Clock Divider Value. This field must be set to <divider value> - 1. For example, a value of Fh means that the clock is divided by 16.
7-2	RESERVED	R		Reserved
1	USB1_CLKCORE_LOAD_DIV	RW	0h	USB1 Core Clock Load Divider Control 0 - Clear 1 - Load the clock divider value set in the USB1_CLKCORE_DIV bit field
0	USB1_CLKCORE_SEL	RW	0h	USB1 Core Clock Source 0 - Selects UART PLL output 1 - Selects NSS PLL output

Table 5-280. Register Call Summary for BOOTCFG_USB1_CLKCTL

BOOT_CFG Functional Description • Clock Muxing, Enabling and Division Registers: [0]
BOOT_CFG Registers • BOOTCFG_USB1_CLKCTL Register (Offset = 6A0h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.86 BOOTCFG_SERIALPORT_CLKCTL Register (Offset = 6A4h) [reset = 3333h]

BOOTCFG_SERIALPORT_CLKCTL is shown in Figure 5-91 and described in Table 5-282.

Serial Port Clock Control Register

Table 5-281. BOOTCFG_SERIALPORT_CLKCTL Instances

Instance	Physical Address
BOOT_CFG	0262 06A4h

Figure 5-91. BOOTCFG_SERIALPORT_CLKCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	MCBSP_CLKS_SEL			RESERVED	MCASP2_AUXCLK_SEL		
R-0h	R/W-3h			R-0h	R/W-3h		
7	6	5	4	3	2	1	0
RESERVED	MCASP1_AUXCLK_SEL			RESERVED	MCASP0_AUXCLK_SEL		
R-0h	R/W-3h			R-0h	R/W-3h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-282. BOOTCFG_SERIALPORT_CLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14-12	MCBSP_CLKS_SEL	R/W	3h	McBSP CLKS Source Selection 000b - AUDIO_OSCCLK 001b - MLB_IO_CLK 010b - MLBP_IO_CLK 011b - SYS_OSCCLK 100b - XREFCLK (external pin) 101b - UART_PLL/4 All others are reserved.
11	RESERVED	R	0h	Reserved
10-8	MCASP2_AUXCLK_SEL	R/W	3h	McASP2 AUXCLK Source Selection 000b - AUDIO_OSCCLK 001b - MLB_IO_CLK 010b - MLBP_IO_CLK 011b - SYS_OSCCLK 100b - XREFCLK (external pin) 101b - UART_PLL/4 All others are reserved.
7	RESERVED	R	0h	Reserved

Table 5-282. BOOTCFG_SERIALPORT_CLKCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-4	MCASP1_AUXCLK_SEL	R/W	3h	McASP1 AUXCLK Source Selection 000b - AUDIO_OSCCLK 001b - MLB_IO_CLK 010b - MLBP_IO_CLK 011b - SYS_OSCCLK 100b - XREFCLK (external pin) 101b - UART_PLL/4 All others are reserved.
3	RESERVED	R	0h	Reserved
2-0	MCASP0_AUXCLK_SEL	R/W	3h	McASP0 AUXCLK Source Selection 000b - AUDIO_OSCCLK 001b - MLB_IO_CLK 010b - MLBP_IO_CLK 011b - SYS_OSCCLK 100b - XREFCLK (external pin) 101b - UART_PLL/4 All others are reserved.

Table 5-283. Register Call Summary for BOOTCFG_SERIALPORT_CLKCTL

BOOT_CFG Functional Description • Clock Muxing, Enabling and Division Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_SERIALPORT_CLKCTL Register (Offset = 6A4h) [reset = 3333h]: [0]

5.1.4.87 BOOTCFG_OSC_CTL Register (Offset = 6A8h) [reset = 200h]

BOOTCFG_OSC_CTL is shown in Figure 5-92 and described in Table 5-285.

Controls the Internal System Oscillator and Audio Oscillator

Table 5-284. BOOTCFG_OSC_CTL Instances

Instance	Physical Address
BOOT_CFG	0262 06A8h

Figure 5-92. BOOTCFG_OSC_CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				AUDIOOSC_GZ	AUDIOOSC_SW2	AUDIOOSC_SW1	AUDIOOSC_RES_SEL
R-0h				R/W-1h	R/W-0h	R/W-1h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED						SYSOSC_GZ	SYSOSC_RES_SEL
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-285. BOOTCFG_OSC_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	AUDIOOSC_GZ	R/W	1h	Audio Oscillator enable control 0 - Audio Oscillator is enabled 1 - Audio Oscillator is disabled
10	AUDIOOSC_SW2	R/W	0h	Selects the frequency range of the Audio oscillator SW2:SW1 control bits 00b - select 5 MHz to 15 MHz range 01b - select 15 MHz to 30 MHz range (default) 10b - select 30 MHz to 40 MHz range 11b - select 40 MHz to 54 MHz range
9	AUDIOOSC_SW1	R/W	1h	Selects the frequency range of the Audio oscillator SW2:SW1 control bits 00b - select 5 MHz to 15 MHz range 01b - select 15 MHz to 30 MHz range (default) 10b - select 30 MHz to 40 MHz range 11b - select 40 MHz to 54 MHz range
8	AUDIOOSC_RES_SEL	R/W	0h	Audio Oscillator Resistor 0 - Selects Internal Resistor 1 - Selects External Resistor
7-2	RESERVED	R	0h	Reserved

Table 5-285. BOOTCFG_OSC_CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	SYSOSC_GZ	R/W	0h	System Oscillator enable control 0 - System Oscillator is enabled 1 - System Oscillator is disabled
0	SYSOSC_RES_SEL	R/W	0h	System Oscillator Resistor 0 - Selects Internal Resistor 1 - Selects External Resistor

Table 5-286. Register Call Summary for BOOTCFG_OSC_CTL

BOOT_CFG Registers
- BOOT_CFG Registers: [0]
- BOOTCFG_OSC_CTL Register (Offset = 6A8h) [reset = 200h]: [0]

5.1.4.88 BOOTCFG_PCIE_CLKCTL Register (Offset = 6ACh) [reset = X]

BOOTCFG_PCIE_CLKCTL is shown in Figure 5-93 and described in Table 5-288.

PCIE Clock Control Register

Table 5-287. BOOTCFG_PCIE_CLKCTL Instances

Instance	Physical Address
BOOT_CFG	0262 06ACh

Figure 5-93. BOOTCFG_PCIE_CLKCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-X							
23	22	21	20	19	18	17	16
RESERVED							
R-X							
15	14	13	12	11	10	9	8
RESERVED							
R-X							
7	6	5	4	3	2	1	0
RESERVED					PCIE_REFCLK_INPUT_SEL		
R-X					R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-288. BOOTCFG_PCIE_CLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	X	Reserved
2-0	PCIE_REFCLK_INPUT_SEL	R/W	0h	PCIE PHY Reference Clock Selection Field values (Others are reserved): 000b - External reference clock from the pads -refclkp_i / refclkm_i 001b - Reserved 010b - Reserved 011b - Reserved 100b - External reference clock from the pads -refclkp_i / refclkm_i 101b - Reserved 110b - Reserved 111b - Reserved

Table 5-289. Register Call Summary for BOOTCFG_PCIE_CLKCTL

BOOT_CFG Functional Description • Clock Muxing, Enabling and Division Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_PCIE_CLKCTL Register (Offset = 6ACh) [reset = X]: [0]

5.1.4.89 BOOTCFG_CHIP_MISC_CTL0 Register (Offset = 700h) [reset = 0h]

BOOTCFG_CHIP_MISC_CTL0 is shown in Figure 5-94 and described in Table 5-291.

Chip Miscellaneous Control Register 0

Table 5-290. BOOTCFG_CHIP_MISC_CTL0 Instances

Instance	Physical Address
BOOT_CFG	0262 0700h

Figure 5-94. BOOTCFG_CHIP_MISC_CTL0 Register

31	30	29	28	27	26	25	24
RESERVED			USB1_PME_EN	RESERVED			
R-			R/W-0h	R-0h			
23	22	21	20	19	18	17	16
RESERVED					USB0_PME_EN	RESERVED	
R-0h					R/W-0h	R-	
15	14	13	12	11	10	9	8
RESERVED			MSMC_BLOCK_PARITY_RST	RESERVED			
R-			R/W-0h	R-			
7	6	5	4	3	2	1	0
RESERVED							
R-							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-291. BOOTCFG_CHIP_MISC_CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R		Reserved
28	USB1_PME_EN	R/W	0h	Enable wakeup event generation from USB1 0 - Disable PME event generation 1 - Enable PME event generation
27-19	RESERVED	R	0h	Reserved
18	USB0_PME_EN	R/W	0h	Enable wakeup event generation from USB0 0 - Disable PME event generation 1 - Enable PME event generation
17-13	RESERVED	R		Reserved
12	MSMC_BLOCK_PARITY_RST	R/W	0h	Controls MSMC parity RAM reset on device reset 0 - MSMC parity RAM will be reset on device reset 1 - MSMC parity RAM will not be reset on device reset
11-0	RESERVED	R		Reserved

Table 5-292. Register Call Summary for BOOTCFG_CHIP_MISC_CTL0

BOOT_CFG Registers
- BOOT_CFG Registers: [0]
- BOOTCFG_CHIP_MISC_CTL0 Register (Offset = 700h) [reset = 0h]: [0]

5.1.4.90 BOOTCFG_SYSENDSTAT Register (Offset = 710h) [reset = 0h]

BOOTCFG_SYSENDSTAT is shown in Figure 5-95 and described in Table 5-294.

This register provides a way for reading the system endianness in an endian-neutral way from A15 core. This register captures the LENDIAN bootmode pin. The value is latched on the rising edge of PORn or RESETFULLn

Table 5-293. BOOTCFG_SYSENDSTAT Instances

Instance	Physical Address
BOOT_CFG	0262 0710h

Figure 5-95. BOOTCFG_SYSENDSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED							SYSENDSTAT
R-							R-1h

LEGEND: R = Read Only; *-n* = value after reset

Table 5-294. BOOTCFG_SYSENDSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R		Reserved
0	SYSENDSTAT	R	1h	System Endian Status 0h - Reserved 1h - System is in Little Endian mode

Table 5-295. Register Call Summary for BOOTCFG_SYSENDSTAT

BOOT_CFG Registers
- BOOTCFG_SYSENDSTAT Register (Offset = 710h) [reset = 0h]: [0]
- BOOT_CFG Registers: [0]

5.1.4.91 BOOTCFG_PLLLOCK_PINCTL Register (Offset = 714h) [reset = 1h]

BOOTCFG_PLLLOCK_PINCTL is shown in Figure 5-96 and described in Table 5-297.

This register controls the PLLLOCK mux and selects which PLL Lock signal is routed to the PLLLOCK device pin.Only one PLL signal should be selected at any time.

Table 5-296. BOOTCFG_PLLLOCK_PINCTL Instances

Instance	Physical Address
BOOT_CFG	0262 0714h

Figure 5-96. BOOTCFG_PLLLOCK_PINCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
UARTPLL_LOCK_SEL	ICSSPLL_LOCK_SEL	DSSPLL_LOCK_SEL	ARMPLL_LOCK_SEL	NSSPLL_LOCK_SEL	PCIEPLL_LOCK_SEL	DDR3APLL_LOCK_SEL	MAINPLL_LOCK_SEL
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-297. BOOTCFG_PLLLOCK_PINCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R		Reserved
7	UARTPLL_LOCK_SEL	R/W	0h	UART PLL lock select
6	ICSSPLL_LOCK_SEL	R/W	0h	ICSS PLL lock select
5	DSSPLL_LOCK_SEL	R/W	0h	DSS PLL lock select
4	ARMPLL_LOCK_SEL	R/W	0h	ARM PLL lock select
3	NSSPLL_LOCK_SEL	R/W	0h	NSS PLL lock select
2	PCIEPLL_LOCK_SEL	R/W	0h	PCIE PHY PLL lock select
1	DDR3APLL_LOCK_SEL	R/W	0h	DDR3A PLL lock select
0	MAINPLL_LOCK_SEL	R/W	1h	Main PLL lock select

Table 5-298. Register Call Summary for BOOTCFG_PLLLOCK_PINCTL

BOOT_CFG Functional Description • BOOTCFG_PLLLOCK_PINCTL, BOOTCFG_PLLLOCK_STAT And BOOTCFG_PLLLOCK_EVAL Registers: [0] • Registers Associated With Some Device External Signals: [0]
BOOT_CFG Registers • BOOTCFG_PLLLOCK_PINCTL Register (Offset = 714h) [reset = 1h]: [0] • BOOT_CFG Registers: [0]

5.1.4.92 BOOTCFG_PLLLOCK_STAT Register (Offset = 718h) [reset = 0h]

BOOTCFG_PLLLOCK_STAT is shown in Figure 5-97 and described in Table 5-300.

This register holds the current status of the PLL Lock bit when BOOTCFG_PLLLOCK_EVAL bit is set and further captures loss ofLOCK in a sticky fashion. Note that the PLL being in its initial unlocked state is also a sticky state and therefore,software has to reevaluate the state using BOOTCFG_PLLLOCK_EVAL after initial PLL configuration and lock.

Table 5-299. BOOTCFG_PLLLOCK_STAT Instances

Instance	Physical Address
BOOT_CFG	0262 0718h

Figure 5-97. BOOTCFG_PLLLOCK_STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
UARTPLL_LOCK_STAT	ICSSPLL_LOCK_STAT	DSSPLL_LOCK_STAT	ARMPLL_LOCK_STAT	NSSPLL_LOCK_STAT	PCIEPLL_LOCK_STAT	DDR3APLL_LOCK_STAT	MAINPLL_LOCK_STAT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 5-300. BOOTCFG_PLLLOCK_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R		Reserved
7	UARTPLL_LOCK_STAT	R	0h	0 - PLL LOCK is de-asserted (either initial state or lock was lost) 1 - PLL LOCK is asserted (PLL never lost Lock after the last BOOTCFG_PLLLOCK_EVAL happened)
6	ICSSPLL_LOCK_STAT	R	0h	0 - PLL LOCK is de-asserted (either initial state or lock was lost) 1 - PLL LOCK is asserted (PLL never lost Lock after the last BOOTCFG_PLLLOCK_EVAL happened)
5	DSSPLL_LOCK_STAT	R	0h	0 - PLL LOCK is de-asserted (either initial state or lock was lost) 1 - PLL LOCK is asserted (PLL never lost Lock after the last BOOTCFG_PLLLOCK_EVAL happened)
4	ARMPLL_LOCK_STAT	R	0h	0 - PLL LOCK is de-asserted (either initial state or lock was lost) 1 - PLL LOCK is asserted (PLL never lost Lock after the last BOOTCFG_PLLLOCK_EVAL happened)
3	NSSPLL_LOCK_STAT	R	0h	0 - PLL LOCK is de-asserted (either initial state or lock was lost) 1 - PLL LOCK is asserted (PLL never lost Lock after the last BOOTCFG_PLLLOCK_EVAL happened)
2	PCIEPLL_LOCK_STAT	R	0h	0 - PLL LOCK is de-asserted (either initial state or lock was lost) 1 - PLL LOCK is asserted (PLL never lost Lock after the last BOOTCFG_PLLLOCK_EVAL happened)
1	DDR3APLL_LOCK_STAT	R	0h	0 - PLL LOCK is de-asserted (either initial state or lock was lost) 1 - PLL LOCK is asserted (PLL never lost Lock after the last BOOTCFG_PLLLOCK_EVAL happened)

Table 5-300. BOOTCFG_PLLLOCK_STAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	MAINPLL_LOCK_STAT	R	0h	0 - PLL LOCK is de-asserted (either initial state or lock was lost) 1 - PLL LOCK is asserted (PLL never lost Lock after the last BOOTCFG_PLLLOCK_EVAL happened)

Table 5-301. Register Call Summary for BOOTCFG_PLLLOCK_STAT

BOOT_CFG Functional Description • BOOTCFG_PLLLOCK_PINCTL, BOOTCFG_PLLLOCK_STAT And BOOTCFG_PLLLOCK_EVAL Registers: [0] • Registers Associated With Some Device External Signals: [0]
BOOT_CFG Registers • BOOTCFG_PLLLOCK_EVAL Register (Offset = 71Ch) [reset = 0h]: [0][1][2][3][4][5][6][7][8] • BOOT_CFG Registers: [0] • BOOTCFG_PLLLOCK_STAT Register (Offset = 718h) [reset = 0h]: [0]

5.1.4.93 BOOTCFG_PLLLOCK_EVAL Register (Offset = 71Ch) [reset = 0h]

BOOTCFG_PLLLOCK_EVAL is shown in Figure 5-98 and described in Table 5-303.

This register is used to evaluate the PLL Lock signal from the corresponding on-chip PLL and capture the status in the BOOTCFG_PLLLOCK_STAT register.

Table 5-302. BOOTCFG_PLLLOCK_EVAL Instances

Instance	Physical Address
BOOT_CFG	0262 071Ch

Figure 5-98. BOOTCFG_PLLLOCK_EVAL Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
UARTPLL_LOCK_EVAL	ICSSPLL_LOCK_EVAL	DSSPLL_LOCK_EVAL	ARMPLL_LOCK_EVAL	NSSPLL_LOCK_EVAL	PCIEPLL_LOCK_EVAL	DDR3APLL_LOCK_EVAL	MAINPLL_LOCK_EVAL
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-303. BOOTCFG_PLLLOCK_EVAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R		Reserved
7	UARTPLL_LOCK_EVAL	R/W	0h	Write 0 - No action Write 1 - PLL Lock will be reevaluated and captured in the BOOTCFG_PLLLOCK_STAT register for the corresponding PLL Reads always return 0
6	ICSSPLL_LOCK_EVAL	R/W	0h	Write 0 - No action Write 1 - PLL Lock will be reevaluated and captured in the BOOTCFG_PLLLOCK_STAT register for the corresponding PLL Reads always return 0
5	DSSPLL_LOCK_EVAL	R/W	0h	Write 0 - No action Write 1 - PLL Lock will be reevaluated and captured in the BOOTCFG_PLLLOCK_STAT register for the corresponding PLL Reads always return 0
4	ARMPLL_LOCK_EVAL	R/W	0h	Write 0 - No action Write 1 - PLL Lock will be reevaluated and captured in the BOOTCFG_PLLLOCK_STAT register for the corresponding PLL Reads always return 0
3	NSSPLL_LOCK_EVAL	R/W	0h	Write 0 - No action Write 1 - PLL Lock will be reevaluated and captured in the BOOTCFG_PLLLOCK_STAT register for the corresponding PLL Reads always return 0
2	PCIEPLL_LOCK_EVAL	R/W	0h	Write 0 - No action Write 1 - PLL Lock will be reevaluated and captured in the BOOTCFG_PLLLOCK_STAT register for the corresponding PLL Reads always return 0

Table 5-303. BOOTCFG_PLLLOCK_EVAL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	DDR3APLL_LOCK_EVAL	R/W	0h	Write 0 - No action Write 1 - PLL Lock will be reevaluated and captured in the BOOTCFG_PLLLOCK_STAT register for the corresponding PLL Reads always return 0
0	MAINPLL_LOCK_EVAL	R/W	0h	Write 0 - No action Write 1 - PLL Lock will be reevaluated and captured in the BOOTCFG_PLLLOCK_STAT register for the corresponding PLL Reads always return 0

Table 5-304. Register Call Summary for BOOTCFG_PLLLOCK_EVAL

BOOT_CFG Functional Description • BOOTCFG_PLLLOCK_PINCTL, BOOTCFG_PLLLOCK_STAT And BOOTCFG_PLLLOCK_EVAL Registers: [0] • Registers Associated With Some Device External Signals: [0]
BOOT_CFG Registers • BOOTCFG_PLLLOCK_EVAL Register (Offset = 71Ch) [reset = 0h]: [0] • BOOT_CFG Registers: [0] • BOOTCFG_PLLLOCK_STAT Register (Offset = 718h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9]

5.1.4.94 BOOTCFG_PLLCLKSEL_STAT Register (Offset = 720h) [reset = 0h]

BOOTCFG_PLLCLKSEL_STAT is shown in Figure 5-99 and described in Table 5-306.

This register is used to read the status of the PLL input clock selection pins.

Table 5-305. BOOTCFG_PLLCLKSEL_STAT Instances

Instance	Physical Address
BOOT_CFG	0262 0720h

Figure 5-99. BOOTCFG_PLLCLKSEL_STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED							SYSCLKSEL_STAT
R-							R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 5-306. BOOTCFG_PLLCLKSEL_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R		Reserved
0	SYSCLKSEL_STAT	R	0h	Status of SYSCLK PLL mux selection 0 - HF Oscillator drives the SYSCLK as reference input clock to the PLLs 1 - SYSCLK_P and SYSCLK_N pins drive the reference input clock to the PLLs

Table 5-307. Register Call Summary for BOOTCFG_PLLCLKSEL_STAT

BOOT_CFG Functional Description • BOOTCFG_PLLCLKSEL_STAT Register: [0] • Registers Associated With Some Device External Signals: [0]
BOOT_CFG Registers • BOOTCFG_PLLCLKSEL_STAT Register (Offset = 720h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.95 BOOTCFG_USB0_PHY_CTL0 Register (Offset = 738h) [reset = 0h]

BOOTCFG_USB0_PHY_CTL0 is shown in Figure 5-100 and described in Table 5-309.

USB 0 PHY Control Register 0.

Table 5-308. BOOTCFG_USB0_PHY_CTL0 Instances

Instance	Physical Address
BOOT_CFG	0262 0738h

Figure 5-100. BOOTCFG_USB0_PHY_CTL0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED					USB0_UTMI_VBUSVLDEXT	USB0_UTMI_TXBITSTUFFENH	USB0_UTMI_TXBITSTUFFEN
R-					R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-309. BOOTCFG_USB0_PHY_CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R		Reserved
2	USB0_UTMI_VBUSVLDEXT	R/W	0h	External VBUS Valid Indicator. This signal is valid in Device mode and only when the VBUSVLDEXTSEL signal is set to 1. VBUSVLDEXT indicates whether the VBUS signal on the USB cable is valid. In addition, VBUSVLDEXT enables the pull-up resistor on the D+ line. 0: The VBUS signal is not valid, and the pull-up resistor on D+ is disabled. 1: The VBUS signal is valid, and the pull-up resistor on D+ is enabled.
1	USB0_UTMI_TXBITSTUFFENH	R/W	0h	High-Byte Transmit Bit-Stuffing Enable. This controller signal controls bit stuffing on DATAINH(7:0) when OPMODE(1:0) = 3h. 0: Bit stuffing is disabled. 1: Bit stuffing is enabled.
0	USB0_UTMI_TXBITSTUFFEN	R/W	0h	Low-Byte Transmit Bit-Stuffing Enable. This controller signal controls bit stuffing on DATAIN(7:0) when OPMODE(1:0) = 3h. 0: Bit stuffing is disabled. 1: Bit stuffing is enabled.

Table 5-310. Register Call Summary for BOOTCFG_USB0_PHY_CTL0

BOOT_CFG Functional Description • USB PHY Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_USB0_PHY_CTL0 Register (Offset = 738h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.96 BOOTCFG_USB0_PHY_CTL1 Register (Offset = 73Ch) [reset = 0h]

BOOTCFG_USB0_PHY_CTL1 is shown in Figure 5-101 and described in Table 5-312.

USB 0 PHY Control Register 1.

Table 5-311. BOOTCFG_USB0_PHY_CTL1 Instances

Instance	Physical Address
BOOT_CFG	0262 073Ch

Figure 5-101. BOOTCFG_USB0_PHY_CTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED				USB0_PHY_OTG_MUX_VAL_IDDIG	USB0_PHY_OTG_MUX_SEL_IDDIG	USB0_PHY_OTG_MUX_VAL_DRVVBUS	USB0_PHY_OTG_MUX_SEL_DRVVBUS
R-				R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED							
R-							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-312. BOOTCFG_USB0_PHY_CTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R		Reserved
19	USB0_PHY_OTG_MUX_VAL_IDDIG	R/W	0h	Mux value for iddig if PHY_OTG_MUX_SEL_IDDIG=1
18	USB0_PHY_OTG_MUX_SEL_IDDIG	R/W	0h	Mux select for iddig 0: Driven from PHY 1: Use PHY_OTG_MUX_SEL_IDDIG
17	USB0_PHY_OTG_MUX_VAL_DRVVBUS	R/W	0h	Mux value for drvvbus if PHY_OTG_MUX_SEL_DRVVBUS=1
16	USB0_PHY_OTG_MUX_SEL_DRVVBUS	R/W	0h	Mux select for drvvbus 0: Driven from controller 1: UsePHY_OTG_MUX_VAL_DRVVBUS
15-0	RESERVED	R		Reserved

Table 5-313. Register Call Summary for BOOTCFG_USB0_PHY_CTL1

BOOT_CFG Functional Description • USB PHY Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_USB0_PHY_CTL1 Register (Offset = 73Ch) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.97 BOOTCFG_USB0_PHY_CTL2 Register (Offset = 740h) [reset = 448CDC4h]

BOOTCFG_USB0_PHY_CTL2 is shown in Figure 5-102 and described in Table 5-315.

USB 0 PHY Control Register 2.

Table 5-314. BOOTCFG_USB0_PHY_CTL2 Instances

Instance	Physical Address
BOOT_CFG	0262 0740h

Figure 5-102. BOOTCFG_USB0_PHY_CTL2 Register

31	30	29	28	27	26	25	24
RESERVED					USB0_PHY_PC_TXVREFTUNE		
R-					R/W-8h		
23	22	21	20	19	18	17	16
USB0_PHY_PC_TXVREFTUNE	USB0_PHY_PC_TXRISETUNE		USB0_PHY_PC_TXRESTUNE		USB0_PHY_PC_TXPREEMPPULSETUNE	USB0_PHY_PC_TXPREEMPAMPTUNE	
R/W-8h	R/W-2h		R/W-1h		R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
USB0_PHY_PC_TXHSXVTUNE		USB0_PHY_PC_TXFSLSTUNE				USB0_PHY_PC_SQRXTUNE	
R/W-3h		R/W-3h				R/W-3h	
7	6	5	4	3	2	1	0
USB0_PHY_PC_SQRXTUNE	USB0_PHY_PC_OTGTUNE			RESERVED	USB0_PHY_PC_COMPDISTUNE		
R/W-3h	R/W-4h			R-	R/W-4h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-315. BOOTCFG_USB0_PHY_CTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R		Reserved
26-23	USB0_PHY_PC_TXVREFTUNE	R/W	8h	HS DC Voltage Level Adjustment. This bus adjusts the high-speed DC level voltage. 0000: -10% 0001: -8.75% 0010: -7.5% 0011: -6.25% 0100: -5% 0101: -3.75% 0110: -2.5% 0111: -1.25% 1000: Design default 1001: +1.25% 1010: +2.5% 1011: +3.75% 1100: +5% 1101: +6.25% 1110: +7.5% 1111: +8.75%
22-21	USB0_PHY_PC_TXRISETUNE	R/W	2h	HS Transmitter Rise/Fail Time Adjustment. This bus adjusts the rise/fail times of the high-speed waveform. 00: +20% 01: +15% 10: Design default 11: -10%

Table 5-315. BOOTCFG_USB0_PHY_CTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20-19	USB0_PHY_PC_TXRESTUNE	R/W	1h	USB Source impedance Adjustment. In some applications, there can be significant series resistance on the D+ and D- paths between the transceiver and cable. This bus adjusts the driver source impedance to compensate for added series resistance on the USB. Note: Any setting other than the default can result in source impedance variation across process, voltage, and temperature conditions that does not meet USB 2.0 specification limits. 00: Source impedance is increased by approximately 1.5 ohm 01: Design default 10: Source impedance is decreased by approximately 2 ohm 11: Source impedance is decreased by approximately 4 ohm
18	USB0_PHY_PC_TXPREEMPPULSETUNE	R/W	0h	HS Transmitter Pre-Emphasis Duration Control. This signal controls the duration for which the HS pre-emphasis current is sourced onto DP or DM. The HS Transmitter pre-emphasis duration is defined in terms of unit amounts. One unit of pre-emphasis duration is approximately 580 ps and is defined as 1X pre-emphasis duration. This signal is valid only if either TXPREEMPAMPTUNE0 or TXPREEMPAMPTUNE2 is set to 1. 0 (design default): 2X, long pre-emphasis current duration 1: 1X, short pre-emphasis current duration
17-16	USB0_PHY_PC_TXPREEMPAMPTUNE	R/W	0h	HS Transmitter Pre-Emphasis Current Control. This signal controls the amount of current sourced to DP and after a J-to-K or K-to-J transition. The HS Transmitter pre-emphasis current is defined in terms of unit amounts. One unit amount is approximately 600 uA and is defined as 1X pre-emphasis current. 00 (design default): HS Transmitter pre-emphasis is disabled. 01: HS Transmitter pre-emphasis circuit sources 1X pre-emphasis current. 10: HS Transmitter pre-emphasis circuit sources 2X pre-emphasis current. 11: HS Transmitter pre-emphasis circuit sources 3X pre-emphasis current. If these signals are not used, set them to 00b.
15-14	USB0_PHY_PC_TXHSXVTUNE	R/W	3h	Transmitter High-Speed Crossover Adjustment. This bus adjusts the voltage at which the DP and DM signals cross while transmitting in HS mode. 00: Reserved 01: -15mV 10: +15mV 11: Default setting
13-10	USB0_PHY_PC_TXFSLSTUNE	R/W	3h	FS/LS Source impedance Adjustment. This bus adjusts the low- and full-speed single-ended source impedance while driving high.The following adjustment values are based on nominal process, voltage, and temperature. 0000: +5% 0001: +2.5% 0011: Design default 0111: -2.5% 1111: -5% All other bit settings are reserved.

Table 5-315. BOOTCFG_USB0_PHY_CTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9-7	USB0_PHY_PC_SQRXTUNE	R/W	3h	Squelch Threshold Adjustment. This bus adjusts the voltage level for the threshold used to detect valid high-speed data. 000: +15% 001: +10% 010: +5% 011: Design default 100: -5% 101: -10% 110: -15% 111: -20%
6-4	USB0_PHY_PC_OTGTUNE	R/W	4h	VBUS Valid Threshold Adjustment. This bus adjusts the voltage level for the VBUS valid threshold. 000: -12% 001: -9% 010: -6% 011: -3% 100: Design default 101: +3% 110: +6% 111: +9%
3	RESERVED	R		Reserved
2-0	USB0_PHY_PC_COMPDISTUNE	R/W	4h	Disconnect Threshold Adjustment. This bus adjusts the voltage level for the threshold used to detect a disconnect event at the host. 000: -6% 001: -4.5% 010: -3% 011: -1.5% 100: Design default 101: +1.5% 110: +3% 111: +4.5%

Table 5-316. Register Call Summary for BOOTCFG_USB0_PHY_CTL2

BOOT_CFG Functional Description • USB PHY Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_USB0_PHY_CTL2 Register (Offset = 740h) [reset = 448CDC4h]: [0] • BOOT_CFG Registers: [0]

5.1.4.98 BOOTCFG_USB0_PHY_CTL4 Register (Offset = 748h) [reset = 1518000h]

BOOTCFG_USB0_PHY_CTL4 is shown in Figure 5-103 and described in Table 5-318.

USB 0 PHY Control Register 4.

Table 5-317. BOOTCFG_USB0_PHY_CTL4 Instances

Instance	Physical Address
BOOT_CFG	0262 0748h

Figure 5-103. BOOTCFG_USB0_PHY_CTL4 Register

31	30	29	28	27	26	25	24
RESERVED							USB0_PHY_OSC_FSEL
R-							R/W-5h
23	22	21	20	19	18	17	16
USB0_PHY_OSC_FSEL		RESERVED	USB0_PHY_OSC_REFCLKSEL		RESERVED	USB0_CTRL_MISC_DEBUG_EN	USB0_PHY_OTG_VBUSVLDEXTSEL
R/W-5h		R-	R/W-2h		R-	R/W-0h	R/W-1h
15	14	13	12	11	10	9	8
USB0_PHY_OTG_OTGDISABLE	RESERVED						
R/W-1h	R-						
7	6	5	4	3	2	1	0
RESERVED							
R-							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-318. BOOTCFG_USB0_PHY_CTL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R		Reserved
24-22	USB0_PHY_OSC_FSEL	R/W	5h	Reference Clock Frequency Select. Selects the USB2.0 picoPHY reference clock frequency. 000: Reserved 001: Reserved 010: 12MHz 011: 19.2MHz 100: Reserved 101: 24MHz (default) 110: Reserved 111: 50MHz
21	RESERVED	R		Reserved
20-19	USB0_PHY_OSC_REFCLKSEL	R/W	2h	Reference Clock Select for PLL Block. This signal selects the reference clock source for the PLL Block. 00: Reserved 01: External Clock 10: Internal Clock (default) 11: Reserved
18	RESERVED	R		Reserved
17	USB0_CTRL_MISC_DEBUG_EN	R/W	0h	Enable CTL_MISC_DEBUG_DATA 0: disable 1: enable

Table 5-318. BOOTCFG_USB0_PHY_CTL4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	USB0_PHY_OTG_VBUSVLDEXTSEL	R/W	1h	External VBUS Valid Select. This signal selects either the VBUSVLDEXT input or the internal Session Valid comparator to generate the OTGSESSVLD output. To avoid potential glitches in DP, VBUSVLDEXTSEL must be static prior to a power-on reset and remain static during normal operation. The OTGSESSVLD signal, in conjunction with XCVRSEL(1:0), OPMODE(1:0), TERMSEL, DPPULLDOWN, and DMPULLDOWN, control the DP pull-up resistor. If VBUSVLDEXT and the internal Session Valid comparator output are asserted, and VBUSEXTSEL changes, it is possible for OTGSESSVLD to glitch low, causing the DP resistor to be temporarily disabled. 0: The internal Session Valid comparator is used to generate OTGSESSVLD and assert the DP pull-up resistor. 1: The VBUSVLDEXT input is used to generate OTGSESSVLD and assert the DP pull-up resistor.
15	USB0_PHY_OTG_OTGDISABLE	R/W	1h	OTG Block Disable. This signal powers down the VBUS Valid comparator, but not the Session Valid comparator, ADP probe and sense comparators, nor the ID detection circuitry. To save power, if the application does not use the OTG function, this input can be set high. 0: The OTG block is powered up. 1: The OTG block is powered down.
14-0	RESERVED	R		Reserved

Table 5-319. Register Call Summary for BOOTCFG_USB0_PHY_CTL4

BOOT_CFG Functional Description • USB PHY Control Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_USB0_PHY_CTL4 Register (Offset = 748h) [reset = 1518000h]: [0]

5.1.4.99 BOOTCFG_USB1_PHY_CTL0 Register (Offset = 750h) [reset = 0h]

BOOTCFG_USB1_PHY_CTL0 is shown in Figure 5-104 and described in Table 5-321.

USB1 PHY Control Register 0.

Table 5-320. BOOTCFG_USB1_PHY_CTL0 Instances

Instance	Physical Address
BOOT_CFG	0262 0750h

Figure 5-104. BOOTCFG_USB1_PHY_CTL0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED				USB1_UTMI_WORDINTERFACE	USB1_UTMI_VBUSVLDEXT	USB1_UTMI_TXBITSTUFFENH	USB1_UTMI_TXBITSTUFFEN
R-				R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-321. BOOTCFG_USB1_PHY_CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R		Reserved
3	USB1_UTMI_WORDINTERFACE	R/W	0h	
2	USB1_UTMI_VBUSVLDEXT	R/W	0h	External VBUS Valid Indicator. This signal is valid in Device mode and only when the VBUSVLDEXTSEL signal is set to 1. VBUSVLDEXT indicates whether the VBUS signal on the USB cable is valid. In addition, VBUSVLDEXT enables the pull-up resistor on the D+ line. 0: The VBUS signal is not valid, and the pull-up resistor on D+ is disabled. 1: The VBUS signal is valid, and the pull-up resistor on D+ is enabled.
1	USB1_UTMI_TXBITSTUFFENH	R/W	0h	High-Byte Transmit Bit-Stuffing Enable. This controller signal controls bit stuffing on DATAINH(7:0) when OPMODE(1:0) = 3h. 0: Bit stuffing is disabled. 1: Bit stuffing is enabled.
0	USB1_UTMI_TXBITSTUFFEN	R/W	0h	Low-Byte Transmit Bit-Stuffing Enable. This controller signal controls bit stuffing on DATAIN(7:0) when OPMODE(1:0) = 3h. 0: Bit stuffing is disabled. 1: Bit stuffing is enabled.

Table 5-322. Register Call Summary for BOOTCFG_USB1_PHY_CTL0

BOOT_CFG Functional Description • USB PHY Control Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_USB1_PHY_CTL0 Register (Offset = 750h) [reset = 0h]: [0]

5.1.4.100 BOOTCFG_USB1_PHY_CTL1 Register (Offset = 754h) [reset = 0h]

BOOTCFG_USB1_PHY_CTL1 is shown in Figure 5-105 and described in Table 5-324.

USB 1 PHY Control Register 1.

Table 5-323. BOOTCFG_USB1_PHY_CTL1 Instances

Instance	Physical Address
BOOT_CFG	0262 0754h

Figure 5-105. BOOTCFG_USB1_PHY_CTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED				USB1_PHY_OTG_MUX_VAL_IDDIG	USB1_PHY_OTG_MUX_SEL_IDDIG	USB1_PHY_OTG_MUX_VAL_DRVVBUS	USB1_PHY_OTG_MUX_SEL_DRVVBUS
R-				R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED							
R-							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-324. BOOTCFG_USB1_PHY_CTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R		Reserved
19	USB1_PHY_OTG_MUX_VAL_IDDIG	R/W	0h	Mux value for iddig if PHY_OTG_MUX_SEL_IDDIG=1
18	USB1_PHY_OTG_MUX_SEL_IDDIG	R/W	0h	Mux select for iddig 0: Driven from PHY 1: Use PHY_OTG_MUX_SEL_IDDIG
17	USB1_PHY_OTG_MUX_VAL_DRVVBUS	R/W	0h	Mux value for drvvbus if PHY_OTG_MUX_SEL_DRVVBUS=1
16	USB1_PHY_OTG_MUX_SEL_DRVVBUS	R/W	0h	Mux select for drvvbus 0: Driven from controller 1: Use PHY_OTG_MUX_VAL_DRVVBUS
15-0	RESERVED	R		Reserved

Table 5-325. Register Call Summary for BOOTCFG_USB1_PHY_CTL1

BOOT_CFG Functional Description • USB PHY Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_USB1_PHY_CTL1 Register (Offset = 754h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.101 BOOTCFG_USB1_PHY_CTL2 Register (Offset = 758h) [reset = 448CDC4h]

BOOTCFG_USB1_PHY_CTL2 is shown in Figure 5-106 and described in Table 5-327.

USB 1 PHY Control Register 2.

Table 5-326. BOOTCFG_USB1_PHY_CTL2 Instances

Instance	Physical Address
BOOT_CFG	0262 0758h

Figure 5-106. BOOTCFG_USB1_PHY_CTL2 Register

31	30	29	28	27	26	25	24
RESERVED					USB1_PHY_PC_TXVREFTUNE		
R-					R/W-8h		
23	22	21	20	19	18	17	16
USB1_PHY_PC_TXVREFTUNE	USB1_PHY_PC_TXRISETUNE		USB1_PHY_PC_TXRESTUNE		USB1_PHY_PC_TXPREEMPPULSETUNE	USB1_PHY_PC_TXPREEMPAMPTUNE	
R/W-8h	R/W-2h		R/W-1h		R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
USB1_PHY_PC_TXHSXVTUNE		USB1_PHY_PC_TXFSLSTUNE				USB1_PHY_PC_SQRXTUNE	
R/W-3h		R/W-3h				R/W-3h	
7	6	5	4	3	2	1	0
USB1_PHY_PC_SQRXTUNE	USB1_PHY_PC_OTGTUNE			RESERVED	USB1_PHY_PC_COMPDISTUNE		
R/W-3h	R/W-4h			R-	R/W-4h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-327. BOOTCFG_USB1_PHY_CTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R		Reserved
26-23	USB1_PHY_PC_TXVREFTUNE	R/W	8h	HS DC Voltage Level Adjustment. This bus adjusts the high-speed DC level voltage. 0000: -10% 0001: -8.75% 0010: -7.5% 0011: -6.25% 0100: -5% 0101: -3.75% 0110: -2.5% 0111: -1.25% 1000: Design default 1001: +1.25% 1010: +2.5% 1011: +3.75% 1100: +5% 1101: +6.25% 1110: +7.5% 1111: +8.75%
22-21	USB1_PHY_PC_TXRISETUNE	R/W	2h	HS Transmitter Rise/Fail Time Adjustment. This bus adjusts the rise/fail times of the high-speed waveform. 00: +20% 01: +15% 10: Design default 11: -10%

Table 5-327. BOOTCFG_USB1_PHY_CTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20-19	USB1_PHY_PC_TXRESTUNE	R/W	1h	USB Source impedance Adjustment. In some applications, there can be significant series resistance on the D+ and D- paths between the transceiver and cable. This bus adjusts the driver source impedance to compensate for added series resistance on the USB. Note: Any setting other than the default can result in source impedance variation across process, voltage, and temperature conditions that does not meet USB 2.0 specification limits. 00: Source impedance is increased by approximately 1.5 ohm 01: Design default 10: Source impedance is decreased by approximately 2 ohm 11: Source impedance is decreased by approximately 4 ohm
18	USB1_PHY_PC_TXPREEMPPULSETUNE	R/W	0h	HS Transmitter Pre-Emphasis Duration Control. This signal controls the duration for which the HS pre-emphasis current is sourced onto DP or DM. The HS Transmitter pre-emphasis duration is defined in terms of unit amounts. One unit of pre-emphasis duration is approximately 580 ps and is defined as 1X pre-emphasis duration. This signal is valid only if either TXPREEMPAMPTUNE1 or TXPREEMPAMPTUNE2 is set to 1. 0 (design default): 2X, long pre-emphasis current duration 1: 1X, short pre-emphasis current duration
17-16	USB1_PHY_PC_TXPREEMPAMPTUNE	R/W	0h	HS Transmitter Pre-Emphasis Current Control. This signal controls the amount of current sourced to DP and after a J-to-K or K-to-J transition. The HS Transmitter pre-emphasis current is defined in terms of unit amounts. One unit amount is approximately 600 uA and is defined as 1X pre-emphasis current. 00 (design default): HS Transmitter pre-emphasis is disabled. 01: HS Transmitter pre-emphasis circuit sources 1X pre-emphasis current. 10: HS Transmitter pre-emphasis circuit sources 2X pre-emphasis current. 11: HS Transmitter pre-emphasis circuit sources 3X pre-emphasis current. If these signals are not used, set them to 00b.
15-14	USB1_PHY_PC_TXHSXVTUNE	R/W	3h	Transmitter High-Speed Crossover Adjustment. This bus adjusts the voltage at which the DP and DM signals cross while transmitting in HS mode. 00: Reserved 01: -15mV 10: +15mV 11: Default setting
13-10	USB1_PHY_PC_TXFSLSTUNE	R/W	3h	FS/LS Source impedance Adjustment. This bus adjusts the low- and full-speed single-ended source impedance while driving high. The following adjustment values are based on nominal process, voltage, and temperature. 0000: +5% 0001: +2.5% 0011: Design default 0111: -2.5% 1111: -5% All other bit settings are reserved.

Table 5-327. BOOTCFG_USB1_PHY_CTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9-7	USB1_PHY_PC_SQRXTUNE	R/W	3h	Squelch Threshold Adjustment. This bus adjusts the voltage level for the threshold used to detect valid high-speed data. 000: +15% 001: +10% 010: +5% 011: Design default 100: -5% 101: -10% 110: -15% 111: -20%
6-4	USB1_PHY_PC_OTGTUNE	R/W	4h	VBUS Valid Threshold Adjustment. This bus adjusts the voltage level for the VBUS valid threshold. 000: -12% 001: -9% 010: -6% 011: -3% 100: Design default 101: +3% 110: +6% 111: +9%
3	RESERVED	R		Reserved
2-0	USB1_PHY_PC_COMPDISTUNE	R/W	4h	Disconnect Threshold Adjustment. This bus adjusts the voltage level for the threshold used to detect a disconnect event at the host. 000: -6% 001: -4.5% 010: -3% 011: -1.5% 100: Design default 101: +1.5% 110: +3% 111: +4.5%

Table 5-328. Register Call Summary for BOOTCFG_USB1_PHY_CTL2

BOOT_CFG Functional Description • USB PHY Control Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_USB1_PHY_CTL2 Register (Offset = 758h) [reset = 448CDC4h]: [0]

5.1.4.102 BOOTCFG_USB1_PHY_CTL4 Register (Offset = 760h) [reset = 1518000h]

BOOTCFG_USB1_PHY_CTL4 is shown in Figure 5-107 and described in Table 5-330.

USB 1 PHY Control Register 4.

Table 5-329. BOOTCFG_USB1_PHY_CTL4 Instances

Instance	Physical Address
BOOT_CFG	0262 0760h

Figure 5-107. BOOTCFG_USB1_PHY_CTL4 Register

31	30	29	28	27	26	25	24
RESERVED							USB1_PHY_OSC_FSEL
R-							R/W-5h
23	22	21	20	19	18	17	16
USB1_PHY_OSC_FSEL		RESERVED	USB1_PHY_OSC_REFCLKSEL		RESERVED	USB1_CTRL_MISC_DEBUG_EN	USB1_PHY_OTG_VBUSVLDEXTSEL
R/W-5h		R-	R/W-2h		R-	R/W-0h	R/W-1h
15	14	13	12	11	10	9	8
USB1_PHY_OTG_OTGDISABLE	RESERVED						
R/W-1h	R-						
7	6	5	4	3	2	1	0
RESERVED							
R-							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-330. BOOTCFG_USB1_PHY_CTL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R		Reserved
24-22	USB1_PHY_OSC_FSEL	R/W	5h	Reference Clock Frequency Select. Selects the USB2.0 picoPHY reference clock frequency. 000: Reserved 001: Reserved 010: 12MHz 011: 19.2MHz 100: Reserved 101: 24MHz (default) 110: Reserved 111: 50MHz
21	RESERVED	R		Reserved
20-19	USB1_PHY_OSC_REFCLKSEL	R/W	2h	Reference Clock Select for PLL Block. This signal selects the reference clock source for the PLL Block. 00: Reserved 01: External Clock 10: Internal Clock (default) 11: Reserved
18	RESERVED	R		Reserved
17	USB1_CTRL_MISC_DEBUG_EN	R/W	0h	Enable CTL_MISC_DEBUG_DATA 0: disable 1: enable

Table 5-330. BOOTCFG_USB1_PHY_CTL4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	USB1_PHY_OTG_VBUSVLDEXTSEL	R/W	1h	External VBUS Valid Select. This signal selects either the VBUSVLDEXT input or the internal Session Valid comparator to generate the OTGSESSVLD output. To avoid potential glitches in DP, VBUSVLDEXTSEL must be static prior to a power-on reset and remain static during normal operation. The OTGSESSVLD signal, in conjunction with XCVRSEL(1:0), OPMODE(1:0), TERMSEL, DPPULLDOWN, and DMPULLDOWN, control the DP pull-up resistor. If VBUSVLDEXT and the internal Session Valid comparator output are asserted, and VBUSEXTSEL changes, it is possible for OTGSESSVLD to glitch low, causing the DP resistor to be temporarily disabled. 0: The internal Session Valid comparator is used to generate OTGSESSVLD and assert the DP pull-up resistor. 1: The VBUSVLDEXT input is used to generate OTGSESSVLD and assert the DP pull-up resistor.
15	USB1_PHY_OTG_OTGDISABLE	R/W	1h	OTG Block Disable. This signal powers down the VBUS Valid comparator, but not the Session Valid comparator, ADP probe and sense comparators, nor the ID detection circuitry. To save power, if the application does not use the OTG function, this input can be set high. 0: The OTG block is powered up. 1: The OTG block is powered down.
14-0	RESERVED	R		Reserved

Table 5-331. Register Call Summary for BOOTCFG_USB1_PHY_CTL4

BOOT_CFG Functional Description • USB PHY Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_USB1_PHY_CTL4 Register (Offset = 760h) [reset = 1518000h]: [0] • BOOT_CFG Registers: [0]

5.1.4.103 BOOTCFG_USB0_EBC_IN_CTL Register (Offset = 768h) [reset = 0h]

BOOTCFG_USB0_EBC_IN_CTL is shown in Figure 5-108 and described in Table 5-333.

This register controls the routing of Trace Buffer's (TBR) DMA event called DAVDMA_TOP_INTR_PEND to the USB0 module's EBC (External Buffer Control) signal.

Table 5-332. BOOTCFG_USB0_EBC_IN_CTL Instances

Instance	Physical Address
BOOT_CFG	0262 0768h

Figure 5-108. BOOTCFG_USB0_EBC_IN_CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED		EBC14_SEL0		RESERVED		EBC15_SEL0	
R-		R/W-0h		R-		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-333. BOOTCFG_USB0_EBC_IN_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R		Reserved
5-4	EBC14_SEL0	R/W	0h	00: 1h is driven into ctrl_ebc_dev_usb0_inep_pkt_buff_avail(1) (default) 01: 1h is driven into ctrl_ebc_dev_usb0_inep_pkt_buff_avail(1) 10: ARM TBR DMA event routed to ctrl_ebc_dev_usb0_inep_pkt_buff_avail(1) 11: Debug TBR DMA event routed to ctrl_ebc_dev_usb0_inep_pkt_buff_avail(1)
3-2	RESERVED	R		Reserved
1-0	EBC15_SEL0	R/W	0h	00: 1h is driven into ctrl_ebc_dev_usb0_inep_pkt_buff_avail(0) (default) 01: 1h is driven into ctrl_ebc_dev_usb0_inep_pkt_buff_avail(0) 10: ARM TBR DMA event routed to ctrl_ebc_dev_usb0_inep_pkt_buff_avail(0) 11: Debug TBR DMA event routed to ctrl_ebc_dev_usb0_inep_pkt_buff_avail(0)

Table 5-334. Register Call Summary for BOOTCFG_USB0_EBC_IN_CTL

BOOT_CFG Registers
- BOOT_CFG Registers: [0]
- BOOTCFG_USB0_EBC_IN_CTL Register (Offset = 768h) [reset = 0h]: [0]

5.1.4.104 BOOTCFG_USB1_EBC_IN_CTL Register (Offset = 76Ch) [reset = 0h]

BOOTCFG_USB1_EBC_IN_CTL is shown in Figure 5-109 and described in Table 5-336.

This register controls the routing of Trace Buffer's (TBR) DMA event called DAVDMA_TOP_INTR_PEND to the USB1 module's EBC (External Buffer Control) signal.

Table 5-335. BOOTCFG_USB1_EBC_IN_CTL Instances

Instance	Physical Address
BOOT_CFG	0262 076Ch

Figure 5-109. BOOTCFG_USB1_EBC_IN_CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED		EBC14_SEL1		RESERVED		EBC15_SEL1	
R-		R/W-0h		R-		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-336. BOOTCFG_USB1_EBC_IN_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R		Reserved
5-4	EBC14_SEL1	R/W	0h	00: 1h is driven into ctrl_ebc_dev_usb1_inep_pkt_buff_avail(1) (default) 01: 1h is driven into ctrl_ebc_dev_usb1_inep_pkt_buff_avail(1) 10: ARM TBR DMA event routed to ctrl_ebc_dev_usb1_inep_pkt_buff_avail(1) 11: Debug TBR DMA event routed to ctrl_ebc_dev_usb1_inep_pkt_buff_avail(1)
3-2	RESERVED	R		Reserved
1-0	EBC15_SEL1	R/W	0h	00: 1h is driven into ctrl_ebc_dev_usb1_inep_pkt_buff_avail(0) (default) 01: 1h is driven into ctrl_ebc_dev_usb1_inep_pkt_buff_avail(0) 10: ARM TBR DMA event routed to ctrl_ebc_dev_usb1_inep_pkt_buff_avail(0) 11: Debug TBR DMA event routed to ctrl_ebc_dev_usb1_inep_pkt_buff_avail(0)

Table 5-337. Register Call Summary for BOOTCFG_USB1_EBC_IN_CTL

BOOT_CFG Registers
- BOOT_CFG Registers: [0]
- BOOTCFG_USB1_EBC_IN_CTL Register (Offset = 76Ch) [reset = 0h]: [0]

5.1.4.105 BOOTCFG_SCRATCH0 Register (Offset = 780h) [reset = 0h]

BOOTCFG_SCRATCH0 is shown in Figure 5-110 and described in Table 5-339.

General purpose read-write register intended for scratchpad use.

Table 5-338. BOOTCFG_SCRATCH0 Instances

Instance	Physical Address
BOOT_CFG	0262 0780h

Figure 5-110. BOOTCFG_SCRATCH0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRATCH																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-339. BOOTCFG_SCRATCH0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SCRATCH	R/W	0h	

Table 5-340. Register Call Summary for BOOTCFG_SCRATCH0

BOOT_CFG Functional Description • Scrachpad Registers: [0]
BOOT_CFG Registers • BOOTCFG_SCRATCH0 Register (Offset = 780h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.106 BOOTCFG_SCRATCH1 Register (Offset = 784h) [reset = 0h]

BOOTCFG_SCRATCH1 is shown in Figure 5-111 and described in Table 5-342.

General purpose read-write register intended for scratchpad use.

Table 5-341. BOOTCFG_SCRATCH1 Instances

Instance	Physical Address
BOOT_CFG	0262 0784h

Figure 5-111. BOOTCFG_SCRATCH1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRATCH																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-342. BOOTCFG_SCRATCH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SCRATCH	R/W	0h	

Table 5-343. Register Call Summary for BOOTCFG_SCRATCH1

BOOT_CFG Functional Description • Scrachpad Registers: [0]
BOOT_CFG Registers • BOOTCFG_SCRATCH1 Register (Offset = 784h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.107 BOOTCFG_SCRATCH2 Register (Offset = 788h) [reset = 0h]

BOOTCFG_SCRATCH2 is shown in Figure 5-112 and described in Table 5-345.

General purpose read-write register intended for scratchpad use.

Table 5-344. BOOTCFG_SCRATCH2 Instances

Instance	Physical Address
BOOT_CFG	0262 0788h

Figure 5-112. BOOTCFG_SCRATCH2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRATCH																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-345. BOOTCFG_SCRATCH2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SCRATCH	R/W	0h	

Table 5-346. Register Call Summary for BOOTCFG_SCRATCH2

BOOT_CFG Functional Description • Scrachpad Registers: [0]
BOOT_CFG Registers • BOOTCFG_SCRATCH2 Register (Offset = 788h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.108 BOOTCFG_SCRATCH3 Register (Offset = 78Ch) [reset = 0h]

BOOTCFG_SCRATCH3 is shown in Figure 5-113 and described in Table 5-348.

General purpose read-write register intended for scratchpad use.

Table 5-347. BOOTCFG_SCRATCH3 Instances

Instance	Physical Address
BOOT_CFG	0262 078Ch

Figure 5-113. BOOTCFG_SCRATCH3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRATCH																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-348. BOOTCFG_SCRATCH3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SCRATCH	R/W	0h	

Table 5-349. Register Call Summary for BOOTCFG_SCRATCH3

BOOT_CFG Functional Description • Scrachpad Registers: [0]
BOOT_CFG Registers • BOOTCFG_SCRATCH3 Register (Offset = 78Ch) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.109 BOOTCFG_SCRATCH4 Register (Offset = 790h) [reset = 0h]

BOOTCFG_SCRATCH4 is shown in Figure 5-114 and described in Table 5-351.

General purpose read-write register intended for scratchpad use.

Table 5-350. BOOTCFG_SCRATCH4 Instances

Instance	Physical Address
BOOT_CFG	0262 0790h

Figure 5-114. BOOTCFG_SCRATCH4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRATCH																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-351. BOOTCFG_SCRATCH4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SCRATCH	R/W	0h	

Table 5-352. Register Call Summary for BOOTCFG_SCRATCH4

BOOT_CFG Functional Description • Scrachpad Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_SCRATCH4 Register (Offset = 790h) [reset = 0h]: [0]

5.1.4.110 BOOTCFG_SCRATCH5 Register (Offset = 794h) [reset = 0h]

BOOTCFG_SCRATCH5 is shown in Figure 5-115 and described in Table 5-354.

General purpose read-write register intended for scratchpad use.

Table 5-353. BOOTCFG_SCRATCH5 Instances

Instance	Physical Address
BOOT_CFG	0262 0794h

Figure 5-115. BOOTCFG_SCRATCH5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRATCH																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-354. BOOTCFG_SCRATCH5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SCRATCH	R/W	0h	

Table 5-355. Register Call Summary for BOOTCFG_SCRATCH5

BOOT_CFG Functional Description • Scrachpad Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_SCRATCH5 Register (Offset = 794h) [reset = 0h]: [0]

5.1.4.111 BOOTCFG_SCRATCH6 Register (Offset = 798h) [reset = 0h]

BOOTCFG_SCRATCH6 is shown in Figure 5-116 and described in Table 5-357.

General purpose read-write register intended for scratchpad use.

Table 5-356. BOOTCFG_SCRATCH6 Instances

Instance	Physical Address
BOOT_CFG	0262 0798h

Figure 5-116. BOOTCFG_SCRATCH6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRATCH																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-357. BOOTCFG_SCRATCH6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SCRATCH	R/W	0h	

Table 5-358. Register Call Summary for BOOTCFG_SCRATCH6

BOOT_CFG Functional Description
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_SCRATCH6 Register (Offset = 798h) [reset = 0h]: [0]

5.1.4.112 BOOTCFG_SCRATCH7 Register (Offset = 79Ch) [reset = 0h]

BOOTCFG_SCRATCH7 is shown in Figure 5-117 and described in Table 5-360.

General purpose read-write register intended for scratchpad use.

Table 5-359. BOOTCFG_SCRATCH7 Instances

Instance	Physical Address
BOOT_CFG	0262 079Ch

Figure 5-117. BOOTCFG_SCRATCH7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRATCH																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-360. BOOTCFG_SCRATCH7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SCRATCH	R/W	0h	

Table 5-361. Register Call Summary for BOOTCFG_SCRATCH7

BOOT_CFG Functional Description • Scrachpad Registers: [0]
BOOT_CFG Registers • BOOTCFG_SCRATCH7 Register (Offset = 79Ch) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.113 BOOTCFG_SCRATCH8 Register (Offset = 7A0h) [reset = 0h]

BOOTCFG_SCRATCH8 is shown in Figure 5-118 and described in Table 5-363.

General purpose read-write register intended for scratchpad use.

Table 5-362. BOOTCFG_SCRATCH8 Instances

Instance	Physical Address
BOOT_CFG	0262 07A0h

Figure 5-118. BOOTCFG_SCRATCH8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRATCH																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-363. BOOTCFG_SCRATCH8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SCRATCH	R/W	0h	

Table 5-364. Register Call Summary for BOOTCFG_SCRATCH8

BOOT_CFG Functional Description • Scrachpad Registers: [0]
BOOT_CFG Registers • BOOTCFG_SCRATCH8 Register (Offset = 7A0h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.114 BOOTCFG_SCRATCH9 Register (Offset = 7A4h) [reset = 0h]

BOOTCFG_SCRATCH9 is shown in Figure 5-119 and described in Table 5-366.

General purpose read-write register intended for scratchpad use.

Table 5-365. BOOTCFG_SCRATCH9 Instances

Instance	Physical Address
BOOT_CFG	0262 07A4h

Figure 5-119. BOOTCFG_SCRATCH9 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRATCH																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-366. BOOTCFG_SCRATCH9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SCRATCH	R/W	0h	

Table 5-367. Register Call Summary for BOOTCFG_SCRATCH9

BOOT_CFG Functional Description • Scrachpad Registers: [0]
BOOT_CFG Registers • BOOTCFG_SCRATCH9 Register (Offset = 7A4h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.115 BOOTCFG_SCRATCH10 Register (Offset = 7A8h) [reset = 0h]

BOOTCFG_SCRATCH10 is shown in Figure 5-120 and described in Table 5-369.

General purpose read-write register intended for scratchpad use.

Table 5-368. BOOTCFG_SCRATCH10 Instances

Instance	Physical Address
BOOT_CFG	0262 07A8h

Figure 5-120. BOOTCFG_SCRATCH10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRATCH																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-369. BOOTCFG_SCRATCH10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SCRATCH	R/W	0h	

Table 5-370. Register Call Summary for BOOTCFG_SCRATCH10

BOOT_CFG Functional Description • Scrachpad Registers: [0]
BOOT_CFG Registers • BOOTCFG_SCRATCH10 Register (Offset = 7A8h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.116 BOOTCFG_SCRATCH11 Register (Offset = 7ACh) [reset = 0h]

BOOTCFG_SCRATCH11 is shown in Figure 5-121 and described in Table 5-372.

General purpose read-write register intended for scratchpad use.

Table 5-371. BOOTCFG_SCRATCH11 Instances

Instance	Physical Address
BOOT_CFG	0262 07ACh

Figure 5-121. BOOTCFG_SCRATCH11 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRATCH																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-372. BOOTCFG_SCRATCH11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SCRATCH	R/W	0h	

Table 5-373. Register Call Summary for BOOTCFG_SCRATCH11

BOOT_CFG Functional Description • Scrachpad Registers: [0]
BOOT_CFG Registers • BOOTCFG_SCRATCH11 Register (Offset = 7ACh) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.117 BOOTCFG_SCRATCH12 Register (Offset = 7B0h) [reset = 0h]

BOOTCFG_SCRATCH12 is shown in Figure 5-122 and described in Table 5-375.

General purpose read-write register intended for scratchpad use.

Table 5-374. BOOTCFG_SCRATCH12 Instances

Instance	Physical Address
BOOT_CFG	0262 07B0h

Figure 5-122. BOOTCFG_SCRATCH12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRATCH																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-375. BOOTCFG_SCRATCH12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SCRATCH	R/W	0h	

Table 5-376. Register Call Summary for BOOTCFG_SCRATCH12

BOOT_CFG Functional Description • Scrachpad Registers: [0]
BOOT_CFG Registers • BOOTCFG_SCRATCH12 Register (Offset = 7B0h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.118 BOOTCFG_SCRATCH13 Register (Offset = 7B4h) [reset = 0h]

BOOTCFG_SCRATCH13 is shown in Figure 5-123 and described in Table 5-378.

General purpose read-write register intended for scratchpad use.

Table 5-377. BOOTCFG_SCRATCH13 Instances

Instance	Physical Address
BOOT_CFG	0262 07B4h

Figure 5-123. BOOTCFG_SCRATCH13 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRATCH																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-378. BOOTCFG_SCRATCH13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SCRATCH	R/W	0h	

Table 5-379. Register Call Summary for BOOTCFG_SCRATCH13

BOOT_CFG Functional Description • Scrachpad Registers: [0]
BOOT_CFG Registers • BOOTCFG_SCRATCH13 Register (Offset = 7B4h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.119 BOOTCFG_SCRATCH14 Register (Offset = 7B8h) [reset = 0h]

BOOTCFG_SCRATCH14 is shown in Figure 5-124 and described in Table 5-381.

General purpose read-write register intended for scratchpad use.

Table 5-380. BOOTCFG_SCRATCH14 Instances

Instance	Physical Address
BOOT_CFG	0262 07B8h

Figure 5-124. BOOTCFG_SCRATCH14 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRATCH																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-381. BOOTCFG_SCRATCH14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SCRATCH	R/W	0h	

Table 5-382. Register Call Summary for BOOTCFG_SCRATCH14

BOOT_CFG Functional Description • Scrachpad Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_SCRATCH14 Register (Offset = 7B8h) [reset = 0h]: [0]

5.1.4.120 BOOTCFG_SCRATCH15 Register (Offset = 7BCh) [reset = 0h]

BOOTCFG_SCRATCH15 is shown in Figure 5-125 and described in Table 5-384.

General purpose read-write register intended for scratchpad use.

Table 5-383. BOOTCFG_SCRATCH15 Instances

Instance	Physical Address
BOOT_CFG	0262 07BCh

Figure 5-125. BOOTCFG_SCRATCH15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRATCH																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-384. BOOTCFG_SCRATCH15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SCRATCH	R/W	0h	

Table 5-385. Register Call Summary for BOOTCFG_SCRATCH15

BOOT_CFG Functional Description
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_SCRATCH15 Register (Offset = 7BCh) [reset = 0h]: [0]

5.1.4.121 BOOTCFG_DSP_BOOT_ADDR0_NS Register (Offset = 844h) [reset = 20B000h]

BOOTCFG_DSP_BOOT_ADDR0_NS is shown in Figure 5-126 and described in Table 5-387.

Register to control the non-secure boot address for C66x core 0. This boot address will be used by the C66x core when it is coming out of reset in non-secure state.

Table 5-386. BOOTCFG_DSP_BOOT_ADDR0_NS Instances

Instance	Physical Address
BOOT_CFG	0262 0844h

Figure 5-126. BOOTCFG_DSP_BOOT_ADDR0_NS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ISTP_RST_VAL																						RESERVED									
R/W-00082Ch																						R-									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-387. BOOTCFG_DSP_BOOT_ADDR0_NS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	ISTP_RST_VAL	R/W	00082Ch	Non-secure boot address for C66x core
9-0	RESERVED	R		Reserved

Table 5-388. Register Call Summary for BOOTCFG_DSP_BOOT_ADDR0_NS

BOOT_CFG Functional Description • DSP Boot Address Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_DSP_BOOT_ADDR0_NS Register (Offset = 844h) [reset = 20B000h]: [0]

5.1.4.122 BOOTCFG_OBSCLKCTL Register (Offset = C80h) [reset = 100h]

BOOTCFG_OBSCLKCTL is shown in Figure 5-127 and described in Table 5-390.

The observation clock control register provides control to monitor various on-chip PLL and oscillator clocks onto the device OBSCLK pin.

Table 5-389. BOOTCFG_OBSCLKCTL Instances

Instance	Physical Address
BOOT_CFG	0262 0C80h

Figure 5-127. BOOTCFG_OBSCLKCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0							
23	22	21	20	19	18	17	16
RESERVED					RCOSC_OBSCLK_EN	HFOSC_OBSCLK_EN	NSSPLL_OBSCLK_EN
R-0h					RW-0	RW-0	RW-0
15	14	13	12	11	10	9	8
PLLCTL_OBSCLK_EN	DDR3APLL_OBSCLK_EN	ICSSPLL_OBSCLK_EN	UARTPLL_OBSCLK_EN	ARMPLL_OBSCLK_EN	DSSPLL_OBSCLK_EN	MAINPLL_OBSCLK_EN	RSVD_OBSCLK_EN
RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-1
7	6	5	4	3	2	1	0
RESERVED				PLL_OBSCLK_SEL			
R-0h				RW-0h			

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 5-390. BOOTCFG_OBSCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0	Reserved
18	RCOSC_OBSCLK_EN	RW	0	This clock is not available in the current device.
17	HFOSC_OBSCLK_EN	RW	0	Controls the SYS OSC observation clock enable 0 - Disable 1 - Enable
16	NSSPLL_OBSCLK_EN	RW	0	Controls the NSS/IEP PLL observation clock enable 0 - Disable 1 - Enable
15	PLLCTL_OBSCLK_EN	RW	0	Controls the PLL controller observation clock enable 0 - Disable 1 - Enable
14	DDR3APLL_OBSCLK_EN	RW	0	Controls the DDR PLL observation clock enable 0 - Disable 1 - Enable
13	ICSSPLL_OBSCLK_EN	RW	0	Controls the ICSS PLL observation clock enable 0 - Disable 1 - Enable
12	UARTPLL_OBSCLK_EN	RW	0	Controls the UART PLL observation clock enable 0 - Disable 1 - Enable

Table 5-390. BOOTCFG_OBSCLKCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	ARMPLL_OBSCLK_EN	RW	0	Controls the ARM PLL observation clock enable 0 - Disable 1 - Enable
10	DSSPLL_OBSCLK_EN	RW	0	Controls the DSS PLL observation clock enable 0 - Disable 1 - Enable
9	MAINPLL_OBSCLK_EN	RW	0	Controls the MAIN PLL observation clock enable 0 - Disable 1 - Enable
8	RSVD_OBSCLK_EN	RW	1	This clock is not available in the current device.
7-4	RESERVED	R	0	Reserved
3-0	PLL_OBSCLK_SEL	RW	0h	Controls OBSMUX to select which clock is output onto the OBSCLK[P/N] pin 0h - Reserved 1h - MAIN PLL 2h - DSS PLL 3h - ARM PLL 4h - UART PLL 5h - ICSS PLL 6h - DDR PLL 7h - PLL Controller OBSCLK 8h - NSS/IEP PLL 9h - SYS OSC clock Others - Reserved

Table 5-391. Register Call Summary for BOOTCFG_OBSCLKCTL

BOOT_CFG Registers
- BOOT_CFG Registers: [0]
- BOOTCFG_OBSCLKCTL Register (Offset = C80h) [reset = 100h]: [0]

5.1.4.123 BOOTCFG_EFUSE_BOOTROM Register (Offset = C90h) [reset = 0h]

BOOTCFG_EFUSE_BOOTROM is shown in Figure 5-128 and described in Table 5-393.

This register is allocated for BOOTROM usage.

Table 5-392. BOOTCFG_EFUSE_BOOTROM Instances

Instance	Physical Address
BOOT_CFG	0262 0C90h

Figure 5-128. BOOTCFG_EFUSE_BOOTROM Register

31	30	29	28	27	26	25	24
RESERVED				DEVICE_SPEED			
R-				R-0h			
23	22	21	20	19	18	17	16
DEVICE_SPEED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				ARM_SPEED			
R-				R-0h			
7	6	5	4	3	2	1	0
ARM_SPEED							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 5-393. BOOTCFG_EFUSE_BOOTROM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R		Reserved
27-16	DEVICE_SPEED	R	0h	12 EFUSE bits are allocated to indicate the device operating speed. 0b0000 0000 0001 - 600 MHz 0b0000 0000 1xxx - 1000 MHz 0b0000 01xx xxxx - 600 MHz All other values are reserved
15-12	RESERVED	R		Reserved
11-0	ARM_SPEED	R	0h	12 EFUSE bits are allocated to indicate the ARMSS operating speed. 0b0000 0000 001x - 600 MHz 0b0000 0001 xxxx - 1000 MHz 0b0000 1xxx xxxx - 600 MHz All other values are reserved

Table 5-394. Register Call Summary for BOOTCFG_EFUSE_BOOTROM

BOOT_CFG Registers
- BOOTCFG_EFUSE_BOOTROM Register (Offset = C90h) [reset = 0h]: [0]
- BOOT_CFG Registers: [0]

5.1.4.124 BOOTCFG_EVENT_MUXCTL0 Register (Offset = D00h) [reset = 0h]

BOOTCFG_EVENT_MUXCTL0 is shown in Figure 5-129 and described in Table 5-396.

Selects 1 of 256 events for each of 4 GPIO event muxes.

Table 5-395. BOOTCFG_EVENT_MUXCTL0 Instances

Instance	Physical Address
BOOT_CFG	0262 0D00h

Figure 5-129. BOOTCFG_EVENT_MUXCTL0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVTSEL3								EVTSEL2								EVTSEL1								EVTSEL0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-396. BOOTCFG_EVENT_MUXCTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	EVTSEL3	R/W	0h	Event select for Event Mux3
23-16	EVTSEL2	R/W	0h	Event select for Event Mux2
15-8	EVTSEL1	R/W	0h	Event select for Event Mux1
7-0	EVTSEL0	R/W	0h	Event select for Event Mux0

Table 5-397. Register Call Summary for BOOTCFG_EVENT_MUXCTL0

BOOT_CFG Functional Description • Event Mux Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_EVENT_MUXCTL0 Register (Offset = D00h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.125 BOOTCFG_EVENT_MUXCTL1 Register (Offset = D04h) [reset = 0h]

BOOTCFG_EVENT_MUXCTL1 is shown in Figure 5-130 and described in Table 5-399.

Event select for Event Mux3.

Table 5-398. BOOTCFG_EVENT_MUXCTL1 Instances

Instance	Physical Address
BOOT_CFG	0262 0D04h

Figure 5-130. BOOTCFG_EVENT_MUXCTL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVTSEL7								EVTSEL6								EVTSEL5								EVTSEL4							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-399. BOOTCFG_EVENT_MUXCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	EVTSEL7	R/W	0h	Event select for Event Mux7
23-16	EVTSEL6	R/W	0h	Event select for Event Mux6
15-8	EVTSEL5	R/W	0h	Event select for Event Mux5
7-0	EVTSEL4	R/W	0h	Event select for Event Mux4

Table 5-400. Register Call Summary for BOOTCFG_EVENT_MUXCTL1

BOOT_CFG Functional Description • Event Mux Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_EVENT_MUXCTL1 Register (Offset = D04h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.126 BOOTCFG_EVENT_MUXCTL2 Register (Offset = D08h) [reset = 0h]

BOOTCFG_EVENT_MUXCTL2 is shown in Figure 5-131 and described in Table 5-402.

Event select for Event Mux7.

Table 5-401. BOOTCFG_EVENT_MUXCTL2 Instances

Instance	Physical Address
BOOT_CFG	0262 0D08h

Figure 5-131. BOOTCFG_EVENT_MUXCTL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVTSEL11								EVTSEL10								EVTSEL9								EVTSEL8							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-402. BOOTCFG_EVENT_MUXCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	EVTSEL11	R/W	0h	Event select for Event Mux11
23-16	EVTSEL10	R/W	0h	Event select for Event Mux10
15-8	EVTSEL9	R/W	0h	Event select for Event Mux9
7-0	EVTSEL8	R/W	0h	Event select for Event Mux8

Table 5-403. Register Call Summary for BOOTCFG_EVENT_MUXCTL2

BOOT_CFG Functional Description • Event Mux Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_EVENT_MUXCTL2 Register (Offset = D08h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.127 BOOTCFG_EVENT_MUXCTL3 Register (Offset = D0Ch) [reset = 0h]

BOOTCFG_EVENT_MUXCTL3 is shown in Figure 5-132 and described in Table 5-405.

Event select for Event Mux11.

Table 5-404. BOOTCFG_EVENT_MUXCTL3 Instances

Instance	Physical Address
BOOT_CFG	0262 0D0Ch

Figure 5-132. BOOTCFG_EVENT_MUXCTL3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVTSEL15								EVTSEL14								EVTSEL13								EVTSEL12							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-405. BOOTCFG_EVENT_MUXCTL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	EVTSEL15	R/W	0h	Event select for Event Mux15
23-16	EVTSEL14	R/W	0h	Event select for Event Mux14
15-8	EVTSEL13	R/W	0h	Event select for Event Mux13
7-0	EVTSEL12	R/W	0h	Event select for Event Mux12

Table 5-406. Register Call Summary for BOOTCFG_EVENT_MUXCTL3

BOOT_CFG Functional Description • Event Mux Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_EVENT_MUXCTL3 Register (Offset = D0Ch) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.128 BOOTCFG_EVENT_MUXCTL4 Register (Offset = D10h) [reset = 0h]

BOOTCFG_EVENT_MUXCTL4 is shown in Figure 5-133 and described in Table 5-408.

Event select for Event Mux15.

Table 5-407. BOOTCFG_EVENT_MUXCTL4 Instances

Instance	Physical Address
BOOT_CFG	0262 0D10h

Figure 5-133. BOOTCFG_EVENT_MUXCTL4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVTSEL19								EVTSEL18								EVTSEL17								EVTSEL16							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-408. BOOTCFG_EVENT_MUXCTL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	EVTSEL19	R/W	0h	Event select for Event Mux19
23-16	EVTSEL18	R/W	0h	Event select for Event Mux18
15-8	EVTSEL17	R/W	0h	Event select for Event Mux17
7-0	EVTSEL16	R/W	0h	Event select for Event Mux16

Table 5-409. Register Call Summary for BOOTCFG_EVENT_MUXCTL4

BOOT_CFG Functional Description • Event Mux Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_EVENT_MUXCTL4 Register (Offset = D10h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.129 BOOTCFG_EVENT_MUXCTL5 Register (Offset = D14h) [reset = 0h]

BOOTCFG_EVENT_MUXCTL5 is shown in Figure 5-134 and described in Table 5-411.

Event select for Event Mux19.

Table 5-410. BOOTCFG_EVENT_MUXCTL5 Instances

Instance	Physical Address
BOOT_CFG	0262 0D14h

Figure 5-134. BOOTCFG_EVENT_MUXCTL5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVTSEL23								EVTSEL22								EVTSEL21								EVTSEL20							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-411. BOOTCFG_EVENT_MUXCTL5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	EVTSEL23	R/W	0h	Event select for Event Mux23
23-16	EVTSEL22	R/W	0h	Event select for Event Mux22
15-8	EVTSEL21	R/W	0h	Event select for Event Mux21
7-0	EVTSEL20	R/W	0h	Event select for Event Mux20

Table 5-412. Register Call Summary for BOOTCFG_EVENT_MUXCTL5

BOOT_CFG Functional Description • Event Mux Control Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_EVENT_MUXCTL5 Register (Offset = D14h) [reset = 0h]: [0]

5.1.4.130 BOOTCFG_EVENT_MUXCTL6 Register (Offset = D18h) [reset = 0h]

BOOTCFG_EVENT_MUXCTL6 is shown in Figure 5-135 and described in Table 5-414.

Event select for Event Mux23.

Table 5-413. BOOTCFG_EVENT_MUXCTL6 Instances

Instance	Physical Address
BOOT_CFG	0262 0D18h

Figure 5-135. BOOTCFG_EVENT_MUXCTL6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVTSEL27								EVTSEL26								EVTSEL25								EVTSEL24							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-414. BOOTCFG_EVENT_MUXCTL6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	EVTSEL27	R/W	0h	Event select for Event Mux27
23-16	EVTSEL26	R/W	0h	Event select for Event Mux26
15-8	EVTSEL25	R/W	0h	Event select for Event Mux25
7-0	EVTSEL24	R/W	0h	Event select for Event Mux24

Table 5-415. Register Call Summary for BOOTCFG_EVENT_MUXCTL6

BOOT_CFG Functional Description • Event Mux Control Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_EVENT_MUXCTL6 Register (Offset = D18h) [reset = 0h]: [0]

5.1.4.131 BOOTCFG_EVENT_MUXCTL7 Register (Offset = D1Ch) [reset = 0h]

BOOTCFG_EVENT_MUXCTL7 is shown in Figure 5-136 and described in Table 5-417.

Event select for Event Mux27.

Table 5-416. BOOTCFG_EVENT_MUXCTL7 Instances

Instance	Physical Address
BOOT_CFG	0262 0D1Ch

Figure 5-136. BOOTCFG_EVENT_MUXCTL7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVTSEL31								EVTSEL30								EVTSEL29								EVTSEL28							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-417. BOOTCFG_EVENT_MUXCTL7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	EVTSEL31	R/W	0h	Event select for Event Mux31
23-16	EVTSEL30	R/W	0h	Event select for Event Mux30
15-8	EVTSEL29	R/W	0h	Event select for Event Mux29
7-0	EVTSEL28	R/W	0h	Event select for Event Mux28

Table 5-418. Register Call Summary for BOOTCFG_EVENT_MUXCTL7

BOOT_CFG Functional Description • Event Mux Control Registers: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_EVENT_MUXCTL7 Register (Offset = D1Ch) [reset = 0h]: [0]

5.1.4.132 BOOTCFG_EVENT_MUXCTL8 Register (Offset = D20h) [reset = 0h]

BOOTCFG_EVENT_MUXCTL8 is shown in Figure 5-137 and described in Table 5-420.

Event select for Event Mux31.

Table 5-419. BOOTCFG_EVENT_MUXCTL8 Instances

Instance	Physical Address
BOOT_CFG	0262 0D20h

Figure 5-137. BOOTCFG_EVENT_MUXCTL8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVTSEL35								EVTSEL34								EVTSEL33								EVTSEL32							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-420. BOOTCFG_EVENT_MUXCTL8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	EVTSEL35	R/W	0h	Event select for Event Mux35
23-16	EVTSEL34	R/W	0h	Event select for Event Mux34
15-8	EVTSEL33	R/W	0h	Event select for Event Mux33
7-0	EVTSEL32	R/W	0h	Event select for Event Mux32

Table 5-421. Register Call Summary for BOOTCFG_EVENT_MUXCTL8

BOOT_CFG Functional Description • Event Mux Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_EVENT_MUXCTL8 Register (Offset = D20h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.133 BOOTCFG_EVENT_MUXCTL9 Register (Offset = D24h) [reset = 0h]

BOOTCFG_EVENT_MUXCTL9 is shown in Figure 5-138 and described in Table 5-423.

Event select for Event Mux35.

Table 5-422. BOOTCFG_EVENT_MUXCTL9 Instances

Instance	Physical Address
BOOT_CFG	0262 0D24h

Figure 5-138. BOOTCFG_EVENT_MUXCTL9 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVTSEL39								EVTSEL38								EVTSEL37								EVTSEL36							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-423. BOOTCFG_EVENT_MUXCTL9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	EVTSEL39	R/W	0h	Event select for Event Mux39
23-16	EVTSEL38	R/W	0h	Event select for Event Mux38
15-8	EVTSEL37	R/W	0h	Event select for Event Mux37
7-0	EVTSEL36	R/W	0h	Event select for Event Mux36

Table 5-424. Register Call Summary for BOOTCFG_EVENT_MUXCTL9

BOOT_CFG Functional Description • Event Mux Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_EVENT_MUXCTL9 Register (Offset = D24h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.134 BOOTCFG_EVENT_MUXCTL10 Register (Offset = D28h) [reset = 0h]

BOOTCFG_EVENT_MUXCTL10 is shown in Figure 5-139 and described in Table 5-426.

Event select for Event Mux39.

Table 5-425. BOOTCFG_EVENT_MUXCTL10 Instances

Instance	Physical Address
BOOT_CFG	0262 0D28h

Figure 5-139. BOOTCFG_EVENT_MUXCTL10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVTSEL43								EVTSEL42								EVTSEL41								EVTSEL40							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-426. BOOTCFG_EVENT_MUXCTL10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	EVTSEL43	R/W	0h	Event select for Event Mux43
23-16	EVTSEL42	R/W	0h	Event select for Event Mux42
15-8	EVTSEL41	R/W	0h	Event select for Event Mux41
7-0	EVTSEL40	R/W	0h	Event select for Event Mux40

Table 5-427. Register Call Summary for BOOTCFG_EVENT_MUXCTL10

BOOT_CFG Functional Description • Event Mux Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_EVENT_MUXCTL10 Register (Offset = D28h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.135 BOOTCFG_EVENT_MUXCTL11 Register (Offset = D2Ch) [reset = 0h]

BOOTCFG_EVENT_MUXCTL11 is shown in Figure 5-140 and described in Table 5-429.

Event select for Event Mux43.

Table 5-428. BOOTCFG_EVENT_MUXCTL11 Instances

Instance	Physical Address
BOOT_CFG	0262 0D2Ch

Figure 5-140. BOOTCFG_EVENT_MUXCTL11 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVTSEL47								EVTSEL46								EVTSEL45								EVTSEL44							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-429. BOOTCFG_EVENT_MUXCTL11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	EVTSEL47	R/W	0h	Event select for Event Mux47
23-16	EVTSEL46	R/W	0h	Event select for Event Mux46
15-8	EVTSEL45	R/W	0h	Event select for Event Mux45
7-0	EVTSEL44	R/W	0h	Event select for Event Mux44

Table 5-430. Register Call Summary for BOOTCFG_EVENT_MUXCTL11

BOOT_CFG Functional Description • Event Mux Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_EVENT_MUXCTL11 Register (Offset = D2Ch) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.136 BOOTCFG_EVENT_MUXCTL12 Register (Offset = D30h) [reset = 0h]

BOOTCFG_EVENT_MUXCTL12 is shown in Figure 5-141 and described in Table 5-432.

Event select for Event Mux43.

Table 5-431. BOOTCFG_EVENT_MUXCTL12 Instances

Instance	Physical Address
BOOT_CFG	0262 0D30h

Figure 5-141. BOOTCFG_EVENT_MUXCTL12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVTSEL51								EVTSEL50								EVTSEL49								EVTSEL48							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-432. BOOTCFG_EVENT_MUXCTL12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	EVTSEL51	R/W	0h	Event select for Event Mux47
23-16	EVTSEL50	R/W	0h	Event select for Event Mux46
15-8	EVTSEL49	R/W	0h	Event select for Event Mux45
7-0	EVTSEL48	R/W	0h	Event select for Event Mux44

Table 5-433. Register Call Summary for BOOTCFG_EVENT_MUXCTL12

BOOT_CFG Functional Description • Event Mux Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_EVENT_MUXCTL12 Register (Offset = D30h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.137 BOOTCFG_EVENT_MUXCTL13 Register (Offset = D34h) [reset = 0h]

BOOTCFG_EVENT_MUXCTL13 is shown in Figure 5-142 and described in Table 5-435.

Event select for Event Mux43.

Table 5-434. BOOTCFG_EVENT_MUXCTL13 Instances

Instance	Physical Address
BOOT_CFG	0262 0D34h

Figure 5-142. BOOTCFG_EVENT_MUXCTL13 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVTSEL55								EVTSEL54								EVTSEL53								EVTSEL52							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 5-435. BOOTCFG_EVENT_MUXCTL13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	EVTSEL55	R/W	0h	Event select for Event Mux47
23-16	EVTSEL54	R/W	0h	Event select for Event Mux46
15-8	EVTSEL53	R/W	0h	Event select for Event Mux45
7-0	EVTSEL52	R/W	0h	Event select for Event Mux44

Table 5-436. Register Call Summary for BOOTCFG_EVENT_MUXCTL13

BOOT_CFG Functional Description • Event Mux Control Registers: [0]
BOOT_CFG Registers • BOOTCFG_EVENT_MUXCTL13 Register (Offset = D34h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.138 BOOTCFG_DCAN_RAMINIT Register (Offset = E10h) [reset = 0h]

BOOTCFG_DCAN_RAMINIT is shown in Figure 5-143 and described in Table 5-438.

Register to start/check status of the auto-initialization process of DCAN RAM.

Table 5-437. BOOTCFG_DCAN_RAMINIT Instances

Instance	Physical Address
BOOT_CFG	0262 0E10h

Figure 5-143. BOOTCFG_DCAN_RAMINIT Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-							
23	22	21	20	19	18	17	16
RESERVED							
R/W-							
15	14	13	12	11	10	9	8
RESERVED						DCAN1_RAMINIT_DONE	DCAN0_RAMINIT_DONE
R/W-						R/W1toC-0h	R/W1toC-0h
7	6	5	4	3	2	1	0
RESERVED						DCAN1_RAMINIT_START	DCAN0_RAMINIT_START
R/W-						R/WtoPH-0h	R/WtoPH-0h

LEGEND: R/W = Read/Write; R/W1toC = Read/Write 1 to Clear Bit; R/WtoPH = Read/Write to Pulse High; *-n* = value after reset

Table 5-438. BOOTCFG_DCAN_RAMINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R/W		Reserved
9	DCAN1_RAMINIT_DONE	R/W1toC	0h	DCAN1 RAM initialization status 0 = RAM initialization not complete 1 = RAM initialization complete, write 1 to clear
8	DCAN0_RAMINIT_DONE	R/W1toC	0h	DCAN0 RAM initialization status 0 = RAM initialization not complete 1 = RAM initialization complete, write 1 to clear
7-2	RESERVED	R/W		Reserved
1	DCAN1_RAMINIT_START	R/WtoPH	0h	DCAN1 RAM initialization start Writing 0: No effect Writing 1: Start RAM initialization
0	DCAN0_RAMINIT_START	R/WtoPH	0h	DCAN0 RAM initialization start Writing 0: No effect Writing 1: Start RAM initialization

Table 5-439. Register Call Summary for BOOTCFG_DCAN_RAMINIT

BOOT_CFG Registers
- BOOTCFG_DCAN_RAMINIT Register (Offset = E10h) [reset = 0h]: [0]
- BOOT_CFG Registers: [0]

5.1.4.139 BOOTCFG_ETHERNET_CFG Register (Offset = E20h) [reset = 0h]

BOOTCFG_ETHERNET_CFG is shown in Figure 5-144 and described in Table 5-441.

Ethernet Configuration Register.

Table 5-440. BOOTCFG_ETHERNET_CFG Instances

Instance	Physical Address
BOOT_CFG	0262 0E20h

Figure 5-144. BOOTCFG_ETHERNET_CFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED			RGMII_ID_MODE	RESERVED		MODE_SEL	
R-			R/W-0h	R-		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-441. BOOTCFG_ETHERNET_CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R		Reserved
4	RGMII_ID_MODE	R/W	0h	RGMII Internal Delay Mode Selection 0h = Internal delay 1h = No internal delay
3-2	RESERVED	R		Reserved
1-0	MODE_SEL	R/W	0h	Ethernet Port Interface Mode Selection 0h = MII 1h = RMII 2h = RGMII 3h = Reserved

Table 5-442. Register Call Summary for BOOTCFG_ETHERNET_CFG

BOOT_CFG Registers
- BOOT_CFG Registers: [0]
- BOOTCFG_ETHERNET_CFG Register (Offset = E20h) [reset = 0h]: [0]

5.1.4.140 BOOTCFG_MLB_SIG_IO_CTL Register (Offset = E30h) [reset = 0h]

BOOTCFG_MLB_SIG_IO_CTL is shown in Figure 5-145 and described in Table 5-444.

Register controls the MLB SIG pins IO characteristics.

Table 5-443. BOOTCFG_MLB_SIG_IO_CTL Instances

Instance	Physical Address
BOOT_CFG	0262 0E30h

Figure 5-145. BOOTCFG_MLB_SIG_IO_CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED				N_TRIM			
R-				R/W-			
15	14	13	12	11	10	9	8
RESERVED				P_TRIM			
R-				R/W-			
7	6	5	4	3	2	1	0
RESERVED		PWRDN_RX	RESERVED	EN_EXTRES	RESERVED		
R-		R/W-1h	R-	R/W-0h	R-		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-444. BOOTCFG_MLB_SIG_IO_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R		Reserved
19-16	N_TRIM	R/W	-	MLB NMOS LVDS output drive trim value
15-12	RESERVED	R		Reserved
11-8	P_TRIM	R/W	-	MLB PMOS LVDS output drive trim value
7-6	RESERVED	R		Reserved
5	PWRDN_RX	R/W	1h	Power down control for MLB LVDS receiver 0 = Receiver is powered on 1 = Receiver is powered down
4	RESERVED	R		Reserved
3	EN_EXTRES	R/W	0h	This bit is not used.
2-0	RESERVED	R		Reserved

Table 5-445. Register Call Summary for BOOTCFG_MLB_SIG_IO_CTL

BOOT_CFG Functional Description • Registers for Control of the MLB IOs: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_MLB_SIG_IO_CTL Register (Offset = E30h) [reset = 0h]: [0]

5.1.4.141 BOOTCFG_MLB_DAT_IO_CTL Register (Offset = E34h) [reset = 28h]

BOOTCFG_MLB_DAT_IO_CTL is shown in Figure 5-146 and described in Table 5-447.

Register controls the MLB DAT pins IO characteristics.

Table 5-446. BOOTCFG_MLB_DAT_IO_CTL Instances

Instance	Physical Address
BOOT_CFG	0262 0E34h

Figure 5-146. BOOTCFG_MLB_DAT_IO_CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED				N_TRIM			
R-				R/W-			
15	14	13	12	11	10	9	8
RESERVED				P_TRIM			
R-				R/W-			
7	6	5	4	3	2	1	0
RESERVED		PWRDN_RX	RESERVED	EN_EXTRES	RESERVED		
R-		R/W-1h	R-	R/W-0h	R-		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-447. BOOTCFG_MLB_DAT_IO_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R		Reserved
19-16	N_TRIM	R/W	-	MLB NMOS LVDS output drive trim value
15-12	RESERVED	R		Reserved
11-8	P_TRIM	R/W	-	MLB PMOS LVDS output drive trim value
7-6	RESERVED	R		Reserved
5	PWRDN_RX	R/W	1h	Power down control for MLB LVDS receiver 0 = Receiver is powered on 1 = Receiver is powered down
4	RESERVED	R		Reserved
3	EN_EXTRES	R/W	0h	This bit is not used.
2-0	RESERVED	R		Reserved

Table 5-448. Register Call Summary for BOOTCFG_MLB_DAT_IO_CTL

BOOT_CFG Functional Description • Registers for Control of the MLB IOs: [0]
BOOT_CFG Registers • BOOTCFG_MLB_DAT_IO_CTL Register (Offset = E34h) [reset = 28h]: [0] • BOOT_CFG Registers: [0]

5.1.4.142 BOOTCFG_MLB_CLK_IO_CTL Register (Offset = E38h) [reset = 11h]

BOOTCFG_MLB_CLK_IO_CTL is shown in Figure 5-147 and described in Table 5-450.

Register controls the MLB CLK pins IO characteristics.

Table 5-449. BOOTCFG_MLB_CLK_IO_CTL Instances

Instance	Physical Address
BOOT_CFG	0262 0E38h

Figure 5-147. BOOTCFG_MLB_CLK_IO_CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED		PWRDN_RX	RESERVED				
R-		R/W-1h	R-				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-450. BOOTCFG_MLB_CLK_IO_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R		Reserved
5	PWRDN_RX	R/W	1h	Power down control for MLB LVDS receiver 0 = Receiver is powered on 1 = Receiver is powered down
4-0	RESERVED	R		Reserved

Table 5-451. Register Call Summary for BOOTCFG_MLB_CLK_IO_CTL

BOOT_CFG Functional Description • Registers for Control of the MLB IOs: [0]
BOOT_CFG Registers • BOOTCFG_MLB_CLK_IO_CTL Register (Offset = E38h) [reset = 11h]: [0] • BOOT_CFG Registers: [0]

5.1.4.143 BOOTCFG_EPWM_CTL Register (Offset = E40h) [reset = 0h]

BOOTCFG_EPWM_CTL is shown in Figure 5-148 and described in Table 5-453.

EPWM Control Register.

Table 5-452. BOOTCFG_EPWM_CTL Instances

Instance	Physical Address
BOOT_CFG	0262 0E40h

Figure 5-148. BOOTCFG_EPWM_CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							EPWM_EALLOW
R-							R/W-0h
15	14	13	12	11	10	9	8
RESERVED		EPWM_SOCB_SEL	EPWM_SOCA_SEL	RESERVED			EPWM3_SYNCSEL
R-		R/W-0h	R/W-0h	R-			R/W-0h
7	6	5	4	3	2	1	0
RESERVED		EPWM5_TBCLKEN	EPWM4_TBCLKEN	EPWM3_TBCLKEN	EPWM2_TBCLKEN	EPWM1_TBCLKEN	EPWM0_TBCLKEN
R-		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-453. BOOTCFG_EPWM_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R		Reserved
16	EPWM_EALLOW	R/W	0h	EPWM trip zone and HRPWM config register writeaccess enable 0 = Disable 1 = Enable
15-14	RESERVED	R		Reserved
13	EPWM_SOCB_SEL	R/W	0h	Start of Conversion (SOCB) output source 0 = EPWM[5:0] SOCB outputs 1 = ICSS0 Host INT1
12	EPWM_SOCA_SEL	R/W	0h	Start of Conversion (SOCA) output source 0 = EPWM[5:0] SOCA outputs 1 = ICSS0 Host INT0
11-9	RESERVED	R		Reserved
8	EPWM3_SYNCSEL	R/W	0h	EPWM3 SyncIn source 0 = EPWM3 sync is from EPWM2 (daisy chained) 1 = EPWM3 sync is from ICSS/Pin
7-6	RESERVED	R		Reserved
5	EPWM5_TBCLKEN	R/W	0h	EPWM5 timebase clock 0 = Disable 1 = Enable
4	EPWM4_TBCLKEN	R/W	0h	EPWM4 timebase clock 0 = Disable 1 = Enable

Table 5-453. BOOTCFG_EPWM_CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	EPWM3_TBCLKEN	R/W	0h	EPWM3 timebase clock 0 = Disable 1 = Enable
2	EPWM2_TBCLKEN	R/W	0h	EPWM2 timebase clock 0 = Disable 1 = Enable
1	EPWM1_TBCLKEN	R/W	0h	EPWM1 timebase clock 0 = Disable 1 = Enable
0	EPWM0_TBCLKEN	R/W	0h	EPWM0 timebase clock 0 = Disable 1 = Enable

Table 5-454. Register Call Summary for BOOTCFG_EPWM_CTL

BOOT_CFG Functional Description • eCAP/ePWM/eQEP Control and Status Registers: [0]
BOOT_CFG Registers • BOOTCFG_EPWM_CTL Register (Offset = E40h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.144 BOOTCFG_ECAP_CAPEVT_CTL Register (Offset = E50h) [reset = 100h]

BOOTCFG_ECAP_CAPEVT_CTL is shown in Figure 5-149 and described in Table 5-456.

ECAP Event Control Register.

Table 5-455. BOOTCFG_ECAP_CAPEVT_CTL Instances

Instance	Physical Address
BOOT_CFG	0262 0E50h

Figure 5-149. BOOTCFG_ECAP_CAPEVT_CTL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECAP1_CAP_EVT					RESERVED			ECAP0_CAP_EVT				
R-0h			R/W-1h					R-0h			R/W-0h				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-456. BOOTCFG_ECAP_CAPEVT_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-8	ECAP1_CAP_EVT	R/W	1h	Select the eCAP_1 capture event
7-5	RESERVED	R	0h	Reserved
4-0	ECAP0_CAP_EVT	R/W	0h	Select the eCAP_0 capture event

Table 5-457. Register Call Summary for BOOTCFG_ECAP_CAPEVT_CTL

BOOT_CFG Functional Description • eCAP/ePWM/eQEP Control and Status Registers: [0]
BOOT_CFG Registers • BOOTCFG_ECAP_CAPEVT_CTL Register (Offset = E50h) [reset = 100h]: [0] • BOOT_CFG Registers: [0]

5.1.4.145 BOOTCFG_EQEP_STAT Register (Offset = E60h) [reset = 0h]

BOOTCFG_EQEP_STAT is shown in Figure 5-150 and described in Table 5-459.

EQEP Status Register.

Table 5-458. BOOTCFG_EQEP_STAT Instances

Instance	Physical Address
BOOT_CFG	0262 0E60h

Figure 5-150. BOOTCFG_EQEP_STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED					PHASE_ERR2	PHASE_ERR1	PHASE_ERR0
R-					R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 5-459. BOOTCFG_EQEP_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R		Reserved
2	PHASE_ERR2	R	0h	EQEP2 phase error status 0 = No error 1 = Phase error
1	PHASE_ERR1	R	0h	EQEP1 phase error status 0 = No error 1 = Phase error
0	PHASE_ERR0	R	0h	EQEP0 phase error status 0 = No error 1 = Phase error

Table 5-460. Register Call Summary for BOOTCFG_EQEP_STAT

BOOT_CFG Functional Description • eCAP/ePWM/eQEP Control and Status Registers: [0]
BOOT_CFG Registers • BOOTCFG_EQEP_STAT Register (Offset = E60h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.146 BOOTCFG_LVDS_BG_CTL Register (Offset = E70h) [reset = 0h]

BOOTCFG_LVDS_BG_CTL is shown in Figure 5-151 and described in Table 5-462.

Register controls the LVDS Bandgap cells.

Table 5-461. BOOTCFG_LVDS_BG_CTL Instances

Instance	Physical Address
BOOT_CFG	0262 0E70h

Figure 5-151. BOOTCFG_LVDS_BG_CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
LVDS1_PWRDN_BG	LVDS1_TRIM_EN	LVDS1_TRIM		LVDS0_PWRDN_BG	LVDS0_TRIM_EN	LVDS0_TRIM	
R/W-0	R/W-	R/W-		R/W-0	R/W-	R/W-	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-462. BOOTCFG_LVDS_BG_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R		Reserved
7	LVDS1_PWRDN_BG	R/W	0h	Power down control for the LVDS1 bandgap buffer 0 = powered on 1 = reserved
6	LVDS1_TRIM_EN	R/W		LVDS1 bandgap trim enable. When set high, LVDS1_TRIM bits are used
5-4	LVDS1_TRIM	R/W		LVDS1 bandgap trim input, vbias trims for CPTS_REFCLK, MLBP_SIG, MLBP_DAT and MLBP_CLK.
3	LVDS0_PWRDN_BG	R/W	0h	Power down control for the LVDS0 bandgap buffer 0 = powered on 1 = reserved
2	LVDS0_TRIM_EN	R/W		LVDS0 bandgap trim enable. When set high, LVDS0_TRIM bits are used.
1-0	LVDS0_TRIM	R/W		LVDS0 bandgap trim input, vbias trims for DDR_CLK and SYSCLK.

Table 5-463. Register Call Summary for BOOTCFG_LVDS_BG_CTL

BOOT_CFG Functional Description • Registers for Control of the MLB IOs: [0][1]
BOOT_CFG Registers • BOOTCFG_LVDS_BG_CTL Register (Offset = E70h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.147 BOOTCFG_LDO_USB_CTL Register (Offset = E80h) [reset = 0h]

BOOTCFG_LDO_USB_CTL is shown in Figure 5-152 and described in Table 5-465.

Controls USB PHY supply LDO (Low Drop Output) regulator. LDO provides 0.85V supply to both the USB PHYs.

Table 5-464. BOOTCFG_LDO_USB_CTL Instances

Instance	Physical Address
BOOT_CFG	0262 0E80h

Figure 5-152. BOOTCFG_LDO_USB_CTL Register

31	30	29	28	27	26	25	24
RESERVED						VSET	
R-						R/W-	
23	22	21	20	19	18	17	16
VSET							
R/W-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED			BGAP_TRIM				
R-			R/W-				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-465. BOOTCFG_LDO_USB_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R		Reserved
25-16	VSET	R/W		Used to program the output voltage (vddar) of GPLDOBG (General Purpose Low Drop Output with integrated BandGap)
15-5	RESERVED	R		Reserved
4-0	BGAP_TRIM	R/W		BandGap reference magnitude trim. Requires efuse trim

Table 5-466. Register Call Summary for BOOTCFG_LDO_USB_CTL

BOOT_CFG Functional Description • Registers for Control of Some Device LDOs: [0]
BOOT_CFG Registers • BOOT_CFG Registers: [0] • BOOTCFG_LDO_USB_CTL Register (Offset = E80h) [reset = 0h]: [0]

5.1.4.148 BOOTCFG_LDO_PCIE_CTL Register (Offset = E84h) [reset = 0h]

BOOTCFG_LDO_PCIE_CTL is shown in Figure 5-153 and described in Table 5-468.

Controls PCIe PHY supply LDO (Low Drop Output) regulator. LDO provides 0.85V supply to the PCIe PHY.

Table 5-467. BOOTCFG_LDO_PCIE_CTL Instances

Instance	Physical Address
BOOT_CFG	0262 0E84h

Figure 5-153. BOOTCFG_LDO_PCIE_CTL Register

31	30	29	28	27	26	25	24
RESERVED						VSET	
R-						R/W-	
23	22	21	20	19	18	17	16
VSET							
R/W-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED			BGAP_TRIM				
R-			R/W-				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-468. BOOTCFG_LDO_PCIE_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R		Reserved
25-16	VSET	R/W		Used to program the output voltage (vddar) of GPLDOBG (General Purpose Low Drop Output with integrated BandGap)
15-5	RESERVED	R		Reserved
4-0	BGAP_TRIM	R/W		BandGap reference magnitude trim. Requires efuse trim

Table 5-469. Register Call Summary for BOOTCFG_LDO_PCIE_CTL

BOOT_CFG Functional Description • Registers for Control of Some Device LDOs: [0]
BOOT_CFG Registers • BOOTCFG_LDO_PCIE_CTL Register (Offset = E84h) [reset = 0h]: [0] • BOOT_CFG Registers: [0]

5.1.4.149 BOOTCFG_PADCONFIG0 to BOOTCFG_PADCONFIG259 Register (Offset = 1000h to 140Ch) [reset = See Table 5-5]

BOOTCFG_PADCONFIG0 to BOOTCFG_PADCONFIG259 registers are shown in Figure 5-154 and described in Table 5-471.

Registers to control pad configuration and muxing.

Table 5-470. BOOTCFG_PADCONFIG0 to BOOTCFG_PADCONFIG259 Instances

Instance	Physical Address
BOOT_CFG	0262 1000h to 0262 140Ch

Figure 5-154. BOOTCFG_PADCONFIG0 to BOOTCFG_PADCONFIG259 Registers

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED			BUFFERCLASS		RESERVED	PULLTYPESEL	PULLUDEN
R-			R/W-See Table 5-5		R/W-	R/W-See Table 5-5	R/W-See Table 5-5
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED				MUXMODE			
R-				R/W-See Table 5-5			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 5-471. BOOTCFG_PADCONFIG0 to BOOTCFG_PADCONFIG259 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-21	RESERVED	R		Reserved
20-19	BUFFERCLASS	R/W	See Table 5-5	Buffer class selection (CS1, CS0) For 3.3V I/Os: 0h and 1h - 50B — 50 ohm nominal impedance for the buffer, class B driver 100MHz in 50 ohm mode 2h and 3h - 40D — 40 ohm nominal impedance for the buffer, class D driver 200MHz in 40 ohm mode For 1.8V I/Os: 0h - 50B — 50 ohm nominal impedance for the buffer, class B driver 100MHz in 50 ohm mode 1h - 40C — 40 ohm nominal impedance for the buffer, class C driver 150MHz in 40 ohm mode 2h - 40D — 40 ohm nominal impedance for the buffer, class D driver 200MHz in 40 ohm mode 3h - 40E — 40 ohm nominal impedance for the buffer, class E driver 250MHz in 40 ohm mode
18	RESERVED	R/W		This bit must not be modified.
17	PULLTYPESEL	R/W	See Table 5-5	Pull-up or pull-down resistor selection 0h - Pull-down is selected 1h - Pull-up is selected
16	PULLUDEN	R/W	See Table 5-5	Pull-up/pull-down resistor enable 0h - Pull-up/pull-down enabled 1h - Pull-up/pull-down disabled
15-4	RESERVED	R		Reserved

Table 5-471. BOOTCFG_PADCONFIG0 to BOOTCFG_PADCONFIG259 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	MUXMODE	R/W	See Table 5-5	Pad functional signal mux selection 0h - Primary function 1h - Secondary function 2h - Tertiary function 3h - Quaternary function 4h - Quinary function 5h - Senary function

Table 5-472. Register Call Summary for BOOTCFG_PADCONFIG0

BOOT_CFG Functional Description • Pad Configuration Registers: [0]
BOOT_CFG Registers • BOOTCFG_PADCONFIG0 to BOOTCFG_PADCONFIG259 Register (Offset = 1000h to 140Ch) [reset = See]: [0] • BOOT_CFG Registers: [0]

5.2 Power Management

This chapter describes the power-management architecture implemented in the device.

5.2.1 Power Management Overview

To provide a versatile architecture that supports multiple power-management techniques, the power management framework is built with three levels of resource management: clock, power, and voltage.

These management levels are enforced by defining the managed entities or building blocks of the power management architecture, called the clock, power, and voltage domains. A domain is a group of modules or subsections of the device that share a common entity (for example, common clock source, common voltage source, or common power switch).

To minimize device power consumption, the clocks can be gated (turned off), or the power to the modules can be switched off, when they are not in use. Independent power and clock control of sections of the device allows for turning on and off specific sections of the device without affecting other sections.

5.2.2 Power Sleep Controller (PSC)

The Power Sleep Controller (PSC) includes a Global Power Sleep Controller (GPSC) and a number of Local Power Sleep Controllers (LPSC) that control overall device power by turning off unused power domains and gating off clocks to individual peripherals and modules. The PSC provides the user with an interface to control several important power and clock operations.

5.2.2.1 PSC Terminology

Term	Definition
LPSC	Local Power/Sleep Controller; one per module (domain)
GPSC	Global Power/Sleep Controller; manages the LPSCs
PSC	Power/Sleep Controller, including one GPSC and multiple LPSCs

5.2.2.2 Features

The PSC includes the following features:

- Provides software interface to:
  - Control module power on/off
  - Control module clock on/off
- Supports emulation features: power, clock, and reset.

5.2.2.3 Power Domains

The device has several power domains that can be turned on for operation or off to minimize power dissipation. The Global Power Sleep Controller (GPSC) is used to control the power gating of various power domains.

Table 5-473 shows the power domains assigned to modules in the device.

NOTE: ELM is not supported on this family of devices.

Table 5-473. Power Domains

Power Domain Number (x)	Power Domain Name	Modules	PD Default	Notes
0	ALWAYS_ON	PMMC, DCAN_[0:1], PSC, SERDES, TIMER_6, GTC, Message Manager, BOOT_CFG, I2C_0, MPU_[0:2], DFT_SS, EFUSE, PLLs, PLL controller, ICEMELTER, GPIO_[0:1]	ON	Always on. Cannot be disabled.
1	DEBUG	DEBUGSS, ETB (TETB), TBR (DBG, PMMC, ARMSS)	ON	
2	NSS	Network Coprocessor (NSS)	OFF	NSS power domain (NAVSS + EMAC) is standalone and can be used with CRYPTO power domain either enabled or disabled. However, it is not possible to use CRYPTO when the NSS power domain is OFF. During power-up sequence it is required to have the NSS power domain ON prior to turning the CRYPTO ON. Similarly during power down sequence it is required to power down CRYPTO prior to powering down the NSS domain.
3	CRYPTO	Crypto	OFF	
4	CBASS	CBASS, CPT_[2:14], MPU_[3:13, 15, 16]	ON	

Table 5-473. Power Domains (continued)

Power Domain Number (x)	Power Domain Name	Modules	PD Default	Notes
5	SYS_COMP	EDMA_0, EDMA_1, Semaphore, INTC, GIC, MPU_14, ARM_BOOTROM, TIMER_[0:5], I2C_[1:2], UART_[0:2], MCBSP, SPI[_0:3], QSPI, MMC_[0:1], GPMC, ELM, MLB, ePWM_[0:5], eQEP_[0:2], eCAP_[0:1], MCASP_[0:2]	ON	TIMER_0 and TIMER_5 are dedicated for DSP and ARMSS, respectively. When DSP or ARMSS is powered down, the corresponding TIMER is held in reset and not available for the system. Additionally, these timers are controlled by the DSP/ARMSS LPSCs.
6	Reserved	Reserved	ON	
7	MSMC	MSMC, CPT_[0:1]	ON	
8	DSP	DSP	ON	
9	ARMSS	ARMSS, A15 core	ON	
10	ASRC	ASRC	OFF	
11	ICSS	ICSS_0, ICSS_1	OFF	
12	DSS	DSS	OFF	
13	PCIE	PCIe	OFF	
14	USB	USB_0, USB_1	OFF	
15	DDR3	EMIF DDR3	ON	DDR3 EMIF controller does not support power down, but it supports clock-gating in case of No-DDR configuration.

NOTE: For details on power domain state transitions, please refer to Section 5.2.2.6, *Executing State Transitions*.

5.2.2.4 Clock Domains

Clock gating to each logic block is managed by the Local Power Sleep Controllers (LPSCs). For modules with a dedicated clock or multiple clocks, the LPSC communicates with the PLL controller to enable and disable that module's clock(s) at the source. For modules that share a clock with other modules, the LPSC controls the clock gating logic for each module.

Table 5-474 shows the clock domains in the device.

NOTE: ELM is not supported on this family of devices.

Table 5-474. Clock Domains

Module Domain Number (LPSCy)	Modules	Clock Default	Reset Isolation[1]	Notes
0	DCAN_[0:1], PSC, SERDES, TIMER_6, Message Manager, BOOT_CFG, I2C_0, MPU_[0:2], GPIO_[0:1]	ON	No	Always on power domain. Must never be clock-gated.
1	PMMC	ON	No	Always on power domain. Must never be clock-gated.
2	DEBUGSS, ETB (TETB), TBR (DBG, PMMC, ARMSS)	ON	Yes	
3	Network Coprocessor (NSS)	OFF	No	Even though NAVSS and EMAC have separate clock-stop interfaces, they are controlled by a single common LPSC (LPSC3) at the SoC level. Therefore, it is not possible to clock gate EMAC when only NAVSS is used.

(1) The reset isolation column only shows if the LPSC has reset isolation capability. The power-on default value is reset isolation disabled. The software can enable the reset isolation by writing to the respective LPSC register (MDCTLy, where y = module domain number).

Table 5-474. Clock Domains (continued)

Module Domain Number (LPSCy)	Modules	Clock Default	Reset Isolation[1]	Notes
4	Crypto	OFF	No	
5	CBASS, CPT_[2:14], MPU_[3:13, 15, 16]	ON	No	
6	EDMA_0, EDMA_1, Semaphore, INTC, GIC, MPU_14, ARM_BOOTROM, TIMER_[1:4], I2C_[1:2], UART_[0:2], MCBSP, SPI_[0:3]	ON	No	
7	QSPI	ON	No	
8	MMC_[0:1]	ON	No	
9	GPMC, ELM	ON	No	
10	Reserved	OFF	No	
11	MLB	OFF	No	
12	ePWM_[0:5]	OFF	No	
13	eQEP_[0:2]	OFF	No	
14	eCAP_[0:1]	OFF	No	
15	MCASP_[0:2]	OFF	No	
16	Reserved	ON	Yes	
17	MSMC, CPT_[0:1]	ON	No	MSMC does not support clock-gating.
18	DSP, TIMER_0	ON	No	Even though TIMER_0 and TIMER_5 are located in the SYS_COMP power domain, they are not controlled by the SYS_COMP LPSC. Instead, they are controlled by the corresponding core LPSC (that is LPSC18 for DSP timer and DPSC for A15 timer). DPSC is the Discrete PSC inside the ARMSS.
19	ARMSS, A15 core, TIMER_5 (DPSC)	ON	No	
20	ASRC	OFF	No	
21	ICSS_[0:1]	OFF	Yes	
22	Reserved	OFF	Yes	
23	DSS	OFF	No	
24	PCIe	OFF	No	
25	USB_0	OFF	No	
26	USB_1	OFF	No	
27	EMIF DDR3	ON	Yes	DDR3 EMIF controller does not support power down, but it supports clock-gating in case of No-DDR configuration.

NOTE: For details on module domain state transitions, please refer to Section 5.2.2.6, *Executing State Transitions*.

5.2.2.5 Power Domain and Module States Defined

The PSC module organizes modules into different power domains and into different modules (clock domains). Note that there is no relationship between a power domain and a module domain. These are completely separate entities and may be controlled separately.

5.2.2.5.1 Power Domain States

A power domain can be in only one of two states: ON or OFF, defined as follows:

- **ON:** Power to the power domain is on. Logic/memories are awake.
- **OFF:** Power to the power domain is off. Logic/memories are turned off.

The AlwaysOn power domain is always in the ON state when the device is powered on. The other power domains can be in either the ON or OFF state; that is, the logic/memory for a specific module can remain powered down if it is not used.

5.2.2.5.2 Module States

A module can be in one of two states: Enable or SwRstDisable. As shown in Table 5-475, these two states correspond to combinations of module reset asserted or de-asserted and module clock on or off. Note that module reset is defined to completely reset a given module, such that all hardware is put back into its default state.

Table 5-475. Module States

Module State	Module Reset	Module Clock
Enable	De-asserted	ON
SwRstDisable	Asserted	OFF

The module states are defined as follows:

- **Enable:** A module in the Enable state has its module reset de-asserted and its clock on. This is the normal run-time state for a given module.
- **SwRstDisable:** A module in the SwResetDisable state has its module reset asserted and its clock OFF.

5.2.2.5.3 Local Reset

In addition to module reset described in the previous section, the DSP core and PMMC can be reset using a special local reset. When local reset is asserted, the internal memories (L1P, L1D, and L2) for the core are still accessible. The local reset resets only the corresponding DSP core and PMMC, not the rest of the chip. Local reset is intended to be used by the watchdog timers to reset the DSP core in the event of an error. The procedures for asserting and de-asserting local reset are as follows (y denotes the module domain number):

1. Set MDCTLy[8] LRST to 0x0 to assert local reset.
2. Set MDCTLy[8] LRST to 0x1 to de-assert local reset. The DSP core immediately executes program instructions after reset is de-asserted. Note that the boot sequence does not re-occur unless there is a device-level reset. Execution of code previously in L2 begins execution.

5.2.2.6 Executing State Transitions

This section describes how to execute state transitions for power domains and modules. Examples show how to enable only power domains, only module domains, or a combination. Although the user has complete control of the sequencing, see Section 5.2.2.6.4, *Recommendations for Power Domain/Module Sequencing* for recommendations.

5.2.2.6.1 Power Domain State Transitions

This section describes the basic procedure for transitioning the state of a power domain.

NOTE: The AlwaysOn power domain is always in the ON state when the device is powered-on, and therefore it is not possible to transition this domain to the OFF state. Conversely, the other domains are in the OFF or ON state when the device is powered-on. Transitions between ON and OFF states need to follow the procedure below. See the device Data Manual and respective module TRM chapter for any other considerations especially while turning off the power domain.

The procedure for power domain state transitions follows (x denotes the power domain number, y denotes the module domain number):

1. Wait for PTSTAT[x] GOSTAT to clear to 0. Wait for any previously initiated transitions to finish before initiating a new transition.

2. Set PDCTLx[0] NEXT for an ON (1) or OFF (0) transition. NOTE: When PTCMD[x] GO is set to 1 in the next step, the PDCTLx[0] NEXT field of this power domain and the MDCTLy[4-0] NEXT field of the module in this power domain are evaluated. Therefore, the MDCTLy[4-0] NEXT field may be set for multiple modules before executing this step.
3. Set PTCMD[x] GO to 1 to initiate the state transition(s). The PSC turns on or off the logic/memory for that particular domain.
4. Wait for PTSTAT[x] GOSTAT to self-clear to 0. The domain is safely in the new state only after PTSTAT[x] GOSTAT is cleared to 0.

5.2.2.6.2 Module State Transitions

This section describes the procedure for transitioning the module state.

NOTE: The following procedure is applicable for all LPSC-controlled modules. To transition the module state, one must be aware of several system considerations. Transitions between Enable and Disable states need to follow the procedure below. See the device Data Manual and peripheral chapter for any other considerations especially while transitioning a module to Disable state. Also, before transitioning a module to Enable, if the logic/memories are not in the AlwaysOn power domain, they must be turned on before or in parallel with the transition. See Section 5.2.2.6.1, *Power Domain State Transitions*.

The procedure for module state transitions follows (x denotes the power domain number, y denotes the module domain number):

1. Wait for PTSTAT[x] GOSTAT to clear to 0x0. Wait for any previously initiated transitions to finish before initiating a new transition.
2. Set MDCTLy[4-0] NEXT to Enable (0x3) or Disable (0x0). Note that transitions in multiple MDCTLy[4-0] NEXT fields may be set in this step as long as the corresponding power domain is on.
3. Set PTCMD[x] GO to 1 to initiate the transition(s).
4. Wait for PTSTAT[x] GOSTAT to self-clear to 0. The module is safely in the new state only after PTSTAT[x] GOSTAT clears to 0.

5.2.2.6.3 Concurrent Power Domain/Module State Transitions

This section describes the basic procedure for transitioning the state of a power domain and module domain for modules that are not in the AlwaysOn domain. This may be done separately as described in the sections above, if desired.

The procedure for concurrent power domain/module state transitions follows (X denotes the power domain number, Y denotes the module domain number):

1. Wait for PTSTAT[x] GOSTAT to clear to 0. Wait for any previously initiated transitions to finish before initiating a new transition.
2. Set PDCTLx[0] NEXT for an ON (1) transition.
3. Set MDCTLy[4-0] NEXT to Enable (0x3). Note that transitions in multiple MDCTLy[4-0] NEXT fields may be set in this step as long as the corresponding power domain is on.
4. Set PTSTAT[x] GOSTAT to 1 to initiate the state transition(s). The PSC will turn on the logic/memory for that particular domain, starts the module clock, then de-asserts the module reset.
5. Wait for PTSTAT[x] GOSTAT to self-clear to 0. The module is safely in the new state only after PTSTAT[x] GOSTAT clears to 0.

5.2.2.6.4 Recommendations for Power Domain/Module Sequencing

As noted in Section 5.2.2.5, *Power Domain and Module States Defined* and Section 5.2.2.6, *Executing State Transitions*, a particular peripheral's power domain must be enabled before the module is enabled.

Unless there is a system-level reason to perform each function separately, the recommendation is to use the sequence listed in Section 5.2.2.6.3 rather than individual sequencing. Even though a single write to the appropriate GO bit starts the power domain and module transition, it is still sequenced such that the memory/logic is certain to turn on before the module is enabled. Thus, there is no ill effect in performing these together.

It is important to know that when a power domain is enabled, all logic/memories for that module are immediately turned on and moved to an active state. This means that there will be a spike in current consumption at that particular time. In other words, di/dt will be affected. To minimize the effect on di/dt of enabling power domains, it is recommended to power up modules one at a time. It is also recommended that all modules be initialized early in the application to avoid a large spike in di/dt during normal operation where the application may already be drawing a substantial amount of current from the supply.

5.2.2.7 Emulation Support in the PSC

The PSC supports commands that allow emulation tools to have some control over the state of power domains and modules.

In particular, the PSC supports the following emulation commands:

- Power on and enable features:
  - **Force Power:** Allows emulation to force the power domain into an ON state.
  - **Force Active:** Allows emulation to force the power domain into an ON state and force the module into the Enable state.
- Reset features:
  - **Assert Reset:** Allows emulation to assert the module's local reset.
  - **Wait Reset:** Allows emulation to keep local reset asserted for an extended period of time after software initiates local reset de-assert.
  - **Block Reset:** Allows emulation to block software initiated local and module resets.

Local reset applies only to the DSP core/PMMC domains; module reset applies to all other domains.

5.2.2.8 PSC Registers

5.2.2.8.1 PSC Register Memory Map

Table 5-477 shows the PSC Register memory map.

Table 5-476. PSC Instances

Instance	Base Address
PSC	0235 0000h

Table 5-477. PSC Registers

Offset	Acronym	Register Name	PSC Physical Address	Section
0h	PID	Peripheral Identification Register	0235 0000h	Section 5.2.2.8.2.1
120h	PTCMD	Power Domain Transition Command Register	0235 0120h	Section 5.2.2.8.2.2
128h	PTSTAT	Power Domain Transition Status Register	0235 0128h	Section 5.2.2.8.2.3
200h	PDSTAT0 to PDSTAT17	Power Domain Status Register	0235 0200h to 0235 0244h	Section 5.2.2.8.2.4
300h	PDCTL0 to PDCTL17	Power Domain Control Register 0	0235 0300h to 0235 0344h	Section 5.2.2.8.2.5
800h to 87Ch	MDSTAT0 to MDSTAT31	Module Status Register 0	0235 0800h to 0235 087Ch	Section 5.2.2.8.2.6
A00h to A7Ch	MDCTL0 to MDCTL31	Module Control Register (never gated)	0235 0A00h to 0235 0A7Ch	Section 5.2.2.8.2.7

5.2.2.8.2 Register Description

This section describes, in detail, the module registers with a register figure and a field description table for each. Each register figure identifies the bit field, the register name, read/write capability, and the default value.

5.2.2.8.2.1 PID Register (Offset = 0h) [reset = 44827A00h]

The peripheral identification register is shown in Figure 5-155 and described in Table 5-479.

Table 5-478. PID Instances

Instance	Physical Address
PSC	0235 0000h

Figure 5-155. PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-44827A00h																															

LEGEND: R/W = Read/Write; R = Read only; to clear bit; -*n* = value after reset

Table 5-479. PID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	44827A00h	TI internal data. Identifies revision of peripheral.

Table 5-480. Register Call Summary for PID

Power Sleep Controller (PSC) • PSC Register Memory Map: [0]

5.2.2.8.2.2 PTCMD Register (Offset = 120h) [reset = 0h]

The power domain transition command register is shown in Figure 5-156 and described in Table 5-482.

Table 5-481. PTCMD Instances

Instance	Physical Address
PSC	0235 0120h

Figure 5-156. PTCMD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GO																															
W-0h																															

LEGEND: W = Write only; *-n* = value after reset

Table 5-482. PTCMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GO	W	0h	Power domain GO transition command. One bit per domain. Refer to Table 5-473, Power Domains for the power domain assignment.). Write 1 to cause the state transition interrupt generation block to evaluate the new PTNEXT and MDCTL.NEXT states as the application desired states.

Table 5-483. Register Call Summary for PTCMD

Power Sleep Controller (PSC)
- Power Domain State Transitions: [0][1]
- Module State Transitions: [0]
- PSC Register Memory Map: [0]

5.2.2.8.2.3 PTSTAT Register (Offset = 128h) [reset = 0h]

The power domain transition status register is shown in Figure 5-157 and described in Table 5-485.

Table 5-484. PTSTAT Instances

Instance	Physical Address
PSC	0235 0128h

Figure 5-157. PTSTAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GOSTAT[X]																															
R-0h																															

LEGEND: R = Read only; -*n* = value after reset

Table 5-485. PTSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GOSTAT[X]	R	0h	Power domain transition status. One bit per domain. Refer to Table 5-473, Power Domains for the power domain assignment. • 0 = No transition in progress. • 1 = Power domain is transitioning (i.e., either the power domain is transitioning or modules in this power domain are transitioning).

Table 5-486. Register Call Summary for PTSTAT

Power Sleep Controller (PSC)
- Power Domain State Transitions: [0][1][2]
- Module State Transitions: [0][1][2]
- Concurrent Power Domain/Module State Transitions: [0][1][2][3]
- PSC Register Memory Map: [0]

5.2.2.8.2.4 PDSTAT0 to PDSTAT17 Register (Offset = 200h - 27Ch) [reset = 100h]

The power domain status register is shown in Figure 5-158 and described in Table 5-488.

Refer to Table 5-473, *Power Domains* for the power domain assignment.

Table 5-487. PDSTAT0 to PDSTAT17 Instances

Instance	Physical Address
PSC	0235 0200h - 0235 027Ch

Figure 5-158. PDSTAT0 to PDSTAT17 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						RESERVED	
R-0h						R-1h	
7	6	5	4	3	2	1	0
RESERVED						STATE	
R-0h						R-0h	

LEGEND: R = Read only; -*n* = value after reset

Table 5-488. PDSTAT0 to PDSTAT17 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved.
9-8	RESERVED	R	1h	Reserved.
7-2	RESERVED	R	0h	Reserved.
1-0	STATE	R	0h	Power domain status. Reset value per Table 5-473, Power Domains. • 0 = Power domain is in the OFF state. • 1 = Power domain is in the ON state.

Table 5-489. Register Call Summary for PDSTAT0 to PDSTAT17

5.2.2.8.2.5 PDCTL0 to PDCTL17 Register (Offset = 300h - 37Ch) [reset = 338000h]

The power domain control register is shown in Figure 5-159 and described in Table 5-491.

Refer to Table 5-473, *Power Domains* for the power domain assignment.

Table 5-490. PDCTL0 to PDCTL17 Instances

Instance	Physical Address
PSC	0235 0300h - 0235 037Ch

Figure 5-159. PDCTL0 to PDCTL17 Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED		RESERVED			
R/W-0h	R-0h	R/W-0h		R-0h			
23	22	21	20	19	18	17	16
RESERVED							
R/W-							
15	14	13	12	11	10	9	8
RESERVED				RESERVED		RESERVED	
R/W-				R-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED							NEXT
R-0h							R/W-0

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 5-491. PDCTL0 to PDCTL17 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved.
30	RESERVED	R	0h	Reserved.
29-28	RESERVED	R/W	0h	Reserved.
27-24	RESERVED	R	0h	Reserved.
23-16	RESERVED	R/W	0h	Default value after reset: On domains -0x01; OFF domains -0x33. Reserved.
15-12	RESERVED	R/W	0h	Default value after reset: On domains -0xB; OFF domains -0x8. Reserved.
11-10	RESERVED	R	0h	Reserved.
9-8	RESERVED	R/W	0h	Reserved.
7-1	RESERVED	R	0h	Reserved.
0	NEXT	R/W	0h	Default value after reset: ON domain -1; OFF domains -0. Power domain next state. • 0 = Power domain OFF. • 1 = Power domain ON.

Table 5-492. Register Call Summary for PDCTLx

Power Sleep Controller (PSC)
- Power Domain State Transitions: [0][1]
- Concurrent Power Domain/Module State Transitions: [0]

5.2.2.8.2.6 MDSTAT0 to MDSTAT31 Register (Offset = 800h - 87Ch) [reset = 100h]

The module status register is shown in Figure 5-160 and described in Table 5-494.

Refer to Table 5-474, *Clock Domains* for the clock domain assignment.

Table 5-493. MDSTAT0 to MDSTAT31 Instances

Instance	Physical Address
PSC	0235 0800h - 0235 087Ch

Figure 5-160. MDSTAT0 to MDSTAT31 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			MCKOUT	MRSTDONE	MRST	LRSTDONE	LRST
R-0h			R-	R-0h	R-0h	R-	R-0h
7	6	5	4	3	2	1	0
RESERVED			STATE				
R-0h			R-				

LEGEND: R = Read only; -*n* = value after reset

Table 5-494. MDSTAT0 to MDSTAT31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved.
12	MCKOUT	R	0h	Default value after reset: Per Table 5-474, Clock Domains. Module clock output status. Shows status of module clock; on/off. • 0 = Module clock is off. • 1 = Module clock is on.
11	MRSTDONE	R	0h	Module reset done. Software is responsible for checking that mode reset is done before accessing the module. • 0 = Module reset is not done. • 1 = Module reset is done.
10	MRST	R	0h	Module reset status. Reflects actual state of module reset. • 0 = Module reset is asserted. • 1 = Module reset is de-asserted.
9	LRSTDONE	R	0h	Default value after reset: DSP domain -0; other domains -1. Local reset done. Software is responsible for checking that local reset is done before accessing this module. • 0 = Local reset is not done. • 1 = Local reset is done.
8	LRST	R	0h	Module local reset status. • 0 = Local reset is asserted. • 1 = Local reset is de-asserted.
7-6	RESERVED	R	0h	Reserved
5-0	STATE	R	0h	Default value after reset: Per Table 5-474, Clock Domains. Module state status. Indicates current module status. • 0x0 = SwRstDisable state. • 0x3 = Enable state. • Others = Reserved.

Table 5-495. Register Call Summary for MDSTAT0 to MDSTAT31

5.2.2.8.2.7 MDCTL0 to MDCTL31 Register (Offset = A00h - A7Ch) [reset = 100h]

The module control register is shown in Figure 5-161 and described in Table 5-497.

Refer to Table 5-474, *Clock Domains* for the clock domain assignment.

Table 5-496. MDCTL0 to MDCTL31 Instances

Instance	Physical Address
PSC	0235 0A00h - 0235 0A7Ch

Figure 5-161. MDCTL0 to MDCTL31 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED			RESETISO	RESERVED			LRST
R/W-0h			R/W-0h	R/W-0h			R/W-1h
7	6	5	4	3	2	1	0
RESERVED			NEXT				
R-0h			R/W-0h				

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 5-497. MDCTL0 to MDCTL31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R/W	0h	Reserved.
12	RESETISO	R/W	0h	Reset isolation. • 0 = No reset isolation • 1 = Enable reset isolation. LPSC will block global device resets and will not pause clocks during reset transitions. Applicable only for modules that support reset isolation. See Table 5-474, Clock Domains.
11-9	RESERVED	R/W	0h	Reserved
8	LRST	R/W	1h	Module local reset control. • 0 = Assert local reset. • 1 = De-assert local reset.
7-5	RESERVED	R	0h	Reserved.
4-0	NEXT	R/W	0h	Default value after reset: Per Table 5-474, Clock Domains. Module next state. • 0x0 = SwRstDisable state. • 0x3 = Enable state. • Other = Reserved.

Table 5-498. Register Call Summary for MDCTL0 to MDCTL31

Power Sleep Controller (PSC)
- Local Reset: [0][1]
- Power Domain State Transitions: [0][1]
- Module State Transitions: [0][1]
- Concurrent Power Domain/Module State Transitions: [0][1]

5.3 Reset Management

This chapter describes the reset management in the device.

5.3.1 Reset Management Overview

Resets bring the device or part of the device to a known state after events such as power-up, hardware or software requests. There are numerous reset types on this device, however, all of these resets can be categorized into the following categories.

- Global resets:
  - Power-on reset
  - Hard and Soft resets
  - Emulation reset
  - Test reset
- Local resets:
  - DSP local reset
  - PMMC CPU local reset
  - Module resets.

Local reset is a reset that is applied to any given module in a module domain or to a CPU and/or its mega-module (any tightly coupled controllers delivered with the CPU) and their associated local watchdog timers.

Table 5-499 explains further the types of reset, the reset initiator, and the effects of each reset on the device. For more information on the effects of each reset on the PLL controller and its clocks, see Section 5.4.5.3, *PLL Controller*.

Table 5-499. Reset Types

Reset Type	Initiator	Effects	Blockable?	Global Reset? (1)
Power-on reset (POR)	PORn pin RESETFULLn pin	Total reset of the device. Boot configurations are latched. ROM boot process is initiated.	No	Yes
Hard reset	RESETn pin RSCTRL Register Watchdog Timers (via RSTMUX) Emulation	Resets everything except for test/emu logic and Reset Isolated modules. Asserts both Chip Reset 0 and Chip Reset 1. More details on chip resets 0 and 1 are provided in Section 5.3.3 and Section 5.3.4. This reset is also different from POR in that the PLLCTL assumes power and clocks are stable when Device Reset is asserted. From PSC point of view, Chip Reset 0 causes PSM state machine reset, so all the power domains except the always-on and reset-isolated domains will be turned off before it gets turned on again. Boot configurations are not re-latched. ROM boot process is initiated.	Yes - only WDT No - all other reset sources	Yes
Soft reset (2)	RESETn pin RSCTRL Register Watchdog Timers (via RSTMUX)	Soft Reset will behave like Hard Reset except that PCIe registers (sticky bits only), DDR EMIF registers (DDR is reset isolated) and External Memory contents are retained. Asserts Chip Reset 1. Also, from PSC point of view, Chip Reset 1 does not cause PSM state machine reset. Boot configurations are not re-latched. ROM boot process is initiated.	Yes - only WDT No - all other reset sources	Yes
Emulation reset	On-chip Emulation logic	Equivalent to Hard reset, non-blockable.	No	Yes
Test reset	TRSTn pin (JTAG)	Resets the Test and Emulation logic. For more information about Test reset, see Chapter 12, *On-chip Debug* .	No	No

(1) Global resets are indicated with RESETSTATn output pin

(2) The Reset config register (RSCFG) of the PLLCTL must be set to select the various initiators for Soft reset

Table 5-499. Reset Types (continued)

Reset Type	Initiator	Effects	Blockable?	Global Reset? (1)
DSP local reset	LRESETn pin Watchdog Timer 0 (via RSTMUX) LPSC Registers	Register bit in LPSC controls DSP Core local reset. Used by Watchdog Timers (in the event of a timeout) to reset DSP Core. Can also be initiated by LRESETn device pin. DSP Core memory system and Slave DMA port are still running when DSP Core is in local reset. Does not initiate ROM boot process.	No	No
ARMSS local resets	Watchdog Timer 5 (via RSTMUX) PSC Registers	Resets ARMSS components such as A15 core, memory subsystem, debug logic etc. ARMSS incorporates PSC to generate resets for it's internal modules.	No	No
PMMC CPU local reset	LPSC Registers	PMMC Core will be held in Reset upon any device reset. Master CPU is responsible of loading PMMC software and releasing PMMC from reset.	No	No
Module local reset	LPSC Registers	Module reset is initiated by LPSC and only resets the module controlled by that LPSC.	No	No

NOTE: Note that PORn/RESETFULLn, RESETn and LRESETn pins must not be tied together at the board level.

5.3.2 Power-on Reset

Power-on reset (POR) is a cold reset. That means that it is assumed that the device may not have been powered and is being powered up. Therefore, all logic on the device must be reset to a default state since it was in an unknown state.

Power-on reset is used to reset the entire device, including the test and emulation logic.

Power-on reset is initiated by the following:

1. PORn pin
2. RESETFULLn pin

All output pins except certain test and status pins must not cause contention at power-up. Outputs are disabled and internal pull-up or pull-down circuitry is enabled where appropriate to hold these pins at a stable level. These two requirements must be satisfied based only on the state of the PORn pin or RESETFULLn pin. HHV logic enforces this requirement when PORn is active (low). Asynchronous logic enforces this requirement when PORn is inactive (high) and RESETFULLn is still active (low). See the device Data Manual for more details on IO state during POR /HHV. The PLL reference clocks are required before PORn/RESETFULLn de-assertion.

Reset Controller accepts this reset, and steps through the following reset sequence. See the device Data Manual for detailed reset sequencing procedure and timing diagram.

1. PORn pin needs to be pulled low (active) before powering up the device. The reset signals flow to the entire chip, resetting anything that uses reset asynchronously, and sends a tristate signal to all the I/O pads except RESETSTATn and SERDES IOs, to prevent off-chip contention (transmitters of the IO buffers are disabled during HHV). This happens asynchronously during PORn assertion, even to module-controlled IO pins. See the device Data Manual for more details on IOs during power-on reset.
2. Clocks and resets are propagated throughout the chip to reset any logic that was using reset synchronously. All logic is now reset. All system clocks are propagated at reference clock frequencies, all dividers are bypassed.
3. PORn and/or RESETFULLn must be held active (low) at least for 150µs after all supplies on the board are stable. Reference clocks of all on-chip PLLs need to be running during this time. If on-chip oscillators are used, then PORn and/or RESETFULLn low period must be extended to meet oscillator start-up time (typically around 1 ms). Also, RESETn pin need to be held inactive, that is, high. PORn and/or RESETFULLn then can be de-asserted, once these conditions are satisfied.
4. Sampled-at-reset BOOTMODE pin values are latched as soon as PORn or RESETFULLn goes high. If both PORn and RESETFULLn are active at the same time then BOOTMODE pins will be latched on

the rising edge of signal that is de-asserted last. PORn and RESETFULLn are AND-ed together before connecting to the Reset Controller.

5. The Efuse Farm is sent an active low reset signal and its clocks are enabled.
6. Efuse Farm reset is released after holding it active for the minimum time specified. Efuse Farm starts autoload process, that is, loading all Efuse registers with their values from the Efuse ROM as soon as the reset is released.
7. When the Efuse autoload is complete the Efuse signals PSC to start its initialization sequence by asserting efuse_ready.
8. Similarly, when the PSC initialization is complete it signals Reset Controller. At this point all system clocks are paused.
9. The chip internal synchronous reset is released, and after waiting 8 Main PLL reference clock cycles, all the system clocks are started again.

Generally, PORn is asserted until all power supplies are ramped up and stable plus few additional milliseconds for internal oscillator to stabilize. When a power-management IC (PMIC) is used, this wait time will be configurable in PMIC. If a discrete power-supply solution is used, it is recommended to use an external RC circuit to extend PORn for at least 5–10 ms. Small and inexpensive power supervisors like the TPS3808 with delays built-in can also be used.

5.3.3 Hard Reset (CHIP_0_RST and CHIP_1_RST Asserted)

This reset type has similar effect but assumes that the device was already powered up and has already gone through a power-on reset sequence. This type of resets is referred to commonly as warm resets, device resets, and/or hard resets.

NOTE: This type of reset will be referred to as *Hard reset* in this TRM.

Do not to be confused with hardware resets. Both Hard and Soft resets can be initiated by either hardware or software.

Hard reset does not reset any test or emulation logic. An emulator session will continue unaffected.

Resets initiated through RSTMUX (watchdog timers) are blockable by setting RSCFG register in PLLCTL.

Reset isolation is supported for

- DDR EMIF
- ICSS
- TETB
- TBR

By default, the Main PLL goes to bypass mode and clock settings are reset to default values. In order to avoid this, Reset Isolation Register (RSISO) in the Reset Controller is enabled by Boot ROM code. By doing this, the state of the PLL and/or the dividers in the PLLCTL are retained. This requirement is met in order to retain the system clocks without pausing for all the Reset Isolation modules. Secondary PLLs, namely DDR PLL, ICSS PLL, UART PLL, NSS PLL are not reset on Hard Reset since both DDR EMIF and ICSS support reset isolation feature. The corresponding registers defined in BOOT_CFG which control these PLLs are reset isolated from Hard resets.

The BOOTMODE pin latching is not performed; boot ROM execution will proceed based on values from the previous POR. The Efuse scan autoload sequence is not initiated. PORn/RESETFULLn must remain de-asserted during this time. A Hard reset takes the PSC to its default state. From PSC point of view, hard reset causes PSM state machine reset, so all the power domains except the always-on and reset isolated domains will be turned off before it gets turned on again. The minimum pulse width of RESETn active low is 12 cycles of Main PLL input clock.

Hard reset is initiated by the following:

- RESETn pin
- RSCTRL Register in the PLL Controller
- Watchdog Timers (via BOOTCFG_RSTMUX)

- Emulation

By default, all the initiators listed above are configured to generate a hard reset. Except for emulation, all of the other three initiators can be configured in the RSCFG Register in the Reset Controller to generate soft resets instead.

The Reset Controller module accepts this reset, and steps through the following reset sequence. See the device Data Manual for detailed reset sequencing procedure and timing diagram.

1. RESETn pin is pulled low or internal requestor asserts reset. The reset signals flow to the modules reset, resetting anything that uses reset asynchronously, and sends a tri-state signal to most the I/O pads, to prevent off-chip contention. IOs associated with reset isolated modules would not be tri-stated. For a list of the IOs tri-stated, see the device Data Manual.
2. Any logic that was using reset synchronously is now reset.
3. RESETn pin is released or other internal requestor de-asserts reset (driven inactive).
4. Reset is stretched for an additional 16 SLOWSYSCLK cycles.
5. The chip synchronous reset is released, and after waiting 8 Main PLL clock cycles, all the system clocks that were allowed to pause are started again.
6. When clocks restart after the previously mentioned pause, the device now is out of reset, and starts execution.

NOTE: There is a minimum pulse width requirement and no dependency on power supplies and clocks.

5.3.4 Soft Reset (CHIP_1_RST Asserted)

This reset only resets a portion of the device, a subset of the Hard reset.

NOTE: This type of resets will be referred to as *Soft reset* in this TRM.

Do not to be confused with software resets. Both Hard and Soft resets can be initiated by either hardware or software.

Soft Reset will behave like Hard Reset (see Section 5.3.3, *Hard Reset (CHIP_0_RST and CHIP_1_RST)*) except that PCIe registers (sticky bits only), DDR EMIF registers (DDR is reset isolated) and External Memory contents are retained.

By default all reset sources are configured to initiate a Hard reset. The Reset config register (RSCFG) of the Reset Controler must be set to select the required sources for the Soft reset.

Similarly to Hard reset, Soft reset can be initiated by the following:

- RESETn pin
- RSCTRL Register in the Reset Controller
- Watchdog timers via BOOTCFG_RSTMUX.

5.3.5 DSP Local Resets

The local reset to the DSP resets DSP Core without resetting any other chip components. PSC coordinates this reset from the following sources:

- By Software (through PSC Register)
- By local reset pin (LRESETn)
- DSP Watchdog (WD) timer
- ICEPICK

The DSP Core memory system is not affected by assertion of local reset and thus there is no impact on the ongoing SDMA transactions to the DSP Core.

5.3.5.1 DSP Local Reset by Software

Local reset can be issued through the PSC register bit MDCTL18[8] LRST.

5.3.5.2 DSP Local Reset by pin input (LRESETn)

Local reset is initiated by hardware, by asserting the LRESETn pin (low) and then latching it with the rising edge of LRESETNMIENn pin. The Core is held in reset until LRESETn input is latched high by the rising LRESETNMIENn input.

5.3.5.3 Local Reset by Watchdog Timers

The device incorporates one dedicated TIMER module for DSP and one for ARMSS. These TIMERs can be configured to serve as a Watchdog timer or a General-purpose timer for that core. When configured as watchdog timer, it is expected that software would periodically write the appropriate value in the specified register so that WD timer does not time out. For details, see Section 11.17, *Timers*. The timeout event from the WD timers is routed to the respective core as described in Section 5.1.3.1.8, *BOOT_CFG Reset Multiplex Block*.

When configuring TIMER_0 in Watchdog timer mode, the software must setup the BOOTCFG_RSTMUX0 bits to route the timer events with the following options:

- local DSP core reset
- DSP non-maskable interrupt (NMI)
- NMI followed by a configurable delay and then a local reset
- device reset by requesting Reset Controler.

When configuring TIMER_5 in Watchdog timer mode, the software must setup the BOOTCFG_RSTMUX8 bits to route the timer events with the following options:

- interrupt to ARMSS through GIC
- interrupt to ARMSS through GIC followed by a configurable delay and then device reset request to Reset Controller
- device reset request to Reset Controller

NOTE: TIMER_0 and TIMER_5 are always reset when their corresponding core DSP and ARMSS, respectively, is in reset.

5.3.6 ARMSS Reset

The ARMSS uses a combination of power-on-reset and module-reset to reset its components, such as the Cortex-A15 processor, memory subsystem, debug logic, and so forth. The ARMSS incorporates the PSC to generate resets for its internal modules. Details of reset generation and distribution inside the ARMSS can be found in Section 6.1, *Arm Cortex-A15 Subsystem*.

5.3.7 Reset Priority

If any of the above reset sources occur simultaneously, the PLL Controller processes only the highest priority reset request. The reset request priorities are as follows (high to low):

1. Power-on reset
2. Hard/soft reset

Note that DSP Core local reset does not go through the Reset Controller.

5.3.8 Reset Indicators

The Reset status of the Device is notified in various ways:

- Device reset output pin — RESETSTATn pin for Global resets
- RSTYPE register in PLLCTL captures Global resets

- BOOTCFG_RESET_STAT register (BOOT_CFG) captures Local resets for (each) DSP core or Global device reset
- BOOTCFG_LRSTNMISTAT register captures LRESET/NMI status that are initiated by device pins LRESETn/NMIn.
- MDSTATy PSC register for each module domain (x) shows the actual local/module reset status indicating whether the reset is ongoing or reset is complete.

5.3.8.1 RESETSTATn Pin

There is single reset output pin on this device called RESETSTATn. It provides device reset status to the external world. This pin is driven active when any of the global resets of the device is asserted. It will remain asserted while reset is being propagated throughout the chip and will be de-asserted once reset sequence is complete (when CHIP_0_RST and CHIP_1_RST are de-asserted).

5.3.8.2 RSTYPE Register

The Reset Type (RSTYPE) register inside PLL Controller identifies the last global reset to occur on the device. Software (Boot ROM, bootloaders) can use this information to take different device initialization steps, if desired. This register notes whether the latest global reset was due to power-on reset (PORn/RESETFULLn), device reset via RESETn pin, Reset Controller register (RSCTRL), WD Timers (via RSTMUX), or Emulation reset.

5.3.8.3 RESET_STAT Register

The reset status register (RESET_STAT) captures the status of Local reset (LRx) for each of the DSP cores and also the Global device reset (GRESET). Software can use this information to take different device initialization steps, if desired.

- In case of Local reset: The LRx bits are written as 1 and GR bit is written as 0 only when the DSP Core receives an LRST without receiving a GRESET.
- In case of Global reset: The LRx bits are written as 0 and GR bit is written as 1 only when a GRESET is asserted.

The BOOTCFG_RESET_STAT bits can be cleared by writing 1 to the corresponding bit in the BOOTCFG_RESET_STAT_CLR register.

5.3.9 Reset Isolation

Reset isolation means some modules or subsystems are not reset during global device resets except on PORn and RESETFULLn. PORn and RESETFULLn will reset the entire device including the reset isolated modules. The device, in hardware, supports reset isolation for the following peripherals and its subsystems:

- DDR EMIF
- TETB (DSP Core) — Whenever there is an ownership change for TETB, its registers will be reset. Therefore, the TETB ownership is tied off on the chip level as specified as below:
  - Ownership application owned
  - Treat all accesses as application-initiated
- TBR (Debug_SS, PMMC and ARMSS) — Retain registers and memory contents

Some of the above peripherals and its subsystem allow devices to be daisy-chained. Hence they all must support Reset isolation that allows packet forwarding transactions to continue when the remainder of the device is being reset.

Reset isolation register (RSISO) in Reset Controller must be set for all the reset isolated modules to retain system clocks without pausing on Non-POR resets (that is, all global resets except PORn and RESETFULLn). Also, PSC registers (MDCTLy, where y is the respective module domain) must be configured for all the Reset-isolated modules. This setting is mandatory for reset isolation to work. Therefore, Boot ROM code handles this task during start-up from POR resets.

5.3.10 ICSS Reset Isolation

The device supports warm reset isolation (Hard/Soft Resets, Watchdog Resets) on ICSS. RESETn Pin isolation is not supported.

The ICSS subsystem itself does not support reset isolation natively. Hence reset isolation must be provided by system software to delay reset signals going to ICSS so software can take necessary actions to fence ICSS from these resets. The following needs to be done in order to support reset isolation on ICSS:

- System software must block any soft resets going to ICSS and must generate a early warning event to ICSS. If it is a watchdog reset, the RESETMUX timer must be configured to provide adequate delay before a watchdog reset is issued. Refer to BOOTCFG_RSTMUX0 and BOOTCFG_RSTMUX8 registers for RESETMUX settings. Based on this early warning event, ICSS software must initiate a sequence to complete/suspend any pending events, such as interrupts, bus transactions, etc.
- ICSS pinmux controls must be maintained across these warm resets.
- All corresponding PLLs providing clocks to ICSS must not reset on warm resets. These are ICSS PLL, UART PLL and NSS PLL.

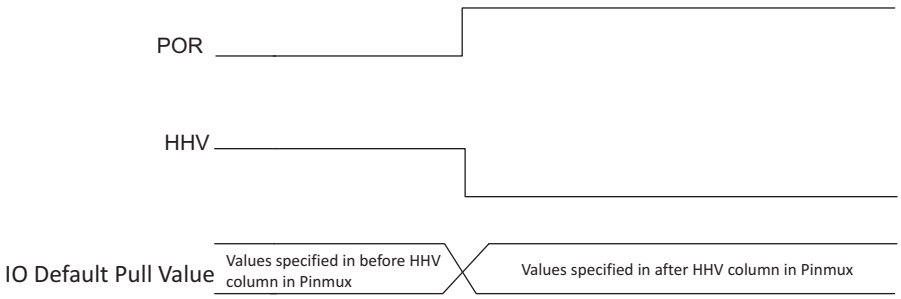
5.3.11 Device Pinmux

Pinmux configuration registers are only reset on PORn or RESETFULLn event. After reset de-assertion, the pins will default to the states defined in Section 5.1, *Control Module (BOOT_CFG)*. During boot-up, it is the responsibility of boot ROM code to configure necessary peripheral pins with correct settings to enable initial boot. Remaining pins must be configured by the secondary boot loader based on the application requirements.

5.3.12 HHV

IOs support HHV mode during power up. HHV is defined as a state when PORn signal is driven LOW. HHV allows IO cells to be tri-stated and is an IO feature. All IO cells have HHV which is asserted (driven) by HHV generated during PORn assertion (see Figure 5-162). There are default pull values during this HHV/PORn assertion specified in the device Data Manual for each pin. All HHV logic controlling the buffer high-impedance control and the associated default pull value will be asynchronous.

Figure 5-162. HHV Condition and Default Pull States on the Device IO



5.3.13 Reset Controller Integration

Pin resets (PORn, RESETFULLn and RESETn) are routed directly to the reset controller. The reset controller, then resets the appropriate device logic. To guarantee stable clocks and power rails, an external power-on reset circuit controls PORn pin of the device and releases it after each power-up. Refer to Table 5-500 for details on mapping of reset requestors to Reset Controller outputs. Here, X indicates that these signals are asserted. Note that when CHIP_0_RST is asserted, CHIP_1_RST is always asserted along with it.

Table 5-500. Mapping of Reset Requestors to Reset Controller Outputs

Requestor	Requestor Type	POR	POR_EARLY (unstretched)	CHIP_0_RST (stretched)	CHIP_0_EARLY_RST (unstretched)	CHIP_1_RST (stretched)	CHIP_1_EARLY_RST (unstretched)	FRESET
PORn	Pin	X	X	X	X	X	X	X
RESETFULLn	Pin	X	X	X	X	X	X	X
RESETn as Hard reset (default)	Pin			X	X	X	X	
RESETn as Soft reset (RSCFG[12] = 1)	Pin					X	X	
LRESETn (1)	Pin							
EMU Chip 0 reset input from Icepick	DCS			X	X	X	X	
RSCTRL as Hard reset (default)	Register			X	X	X	X	
RSCTRL as Soft reset (RSCFG[13] = 1)	Register					X	X	
WDRST (through RSTMUX) as Hard reset (default)	TIMER_0 or TIMER_5 Timeout			X	X	X	X	
WDRST (through RSTMUX) as Soft reset (RSCFG[1] = 1)	TIMER_0 or TIMER_5 Timeout					X	X	
WDRST0 (1) (through RSTMUX)	TIMER_0 Timeout							

(1) This will not request resets from the Reset Controller. They will be routed directly to the LPSC of the DSP Core to initiate a CPU reset.

Table 5-501 is high-level overview of chip-level resets going to the modules and subsystems. More details on reset mapping is shown in Table 5-502, *Reset Mapping and Description*.

Table 5-501. Mapping of Reset Controller Reset Outputs to Modules

Modules	POR	POR_EARLY	CHIP_0_RST	CHIP_0_EARLY_RST	CHIP_1_RST	CHIP_1_EARLY_RST	FRESET
DSP Core	X				X		
DSP Core TETB (POR)			X				
DSP Core TETB (VBUSP_RST)					X		
ADTF (GL_ATRESETN)			X				
Debug_SS (POR_RST)		X					
Debug_SS (CHIP_RST)					X		
Debug_SS (All MOD_G_RST)					X		
DFT_SS (1)		X			X		

(1) These modules also get TRSTn.

Table 5-501. Mapping of Reset Controller Reset Outputs to Modules (continued)

Modules	POR	POR_EARLY	CHIP_0_RST	CHIP_0_EARLY_RST	CHIP_1_RST	CHIP_1_EARLY_RST	FRESET
SEC_CTL	X				X		
KEY_MGR					X		
PSC	X	X	X	X	X	X	
EFUSE_SS (1)		X					X
BOOT_CFG		X			X		
CBA_SS					X		
CP_Tracers					X		
MSMC					X		
DDR3 EMIF Hard Reset			X				
DDR3 EMIF Soft Reset					X		
PMMC Power On Reset	X						
PMMC Main Reset			X				
Other Reset Isolation Modules					X		
All other Modules					X		

5.3.14 Reset Controller Registers

The reset controller registers are part of the PLL Controller register space. For more information, see Section 5.4.5.4, *PLL Controller Registers*.

5.3.15 Reset Mapping

Details on reset mapping to module reset inputs is shown in Table 5-502.

NOTE: ELM is not supported on this family of devices.

Table 5-502. Reset Mapping and Description

Module	Module Reset Port	Description	Source	Source Port	Comments
PMMC	POR_RST	This is the power on reset. Complete debug logic is on this reset. This includes DAP interface, ITM interface, ATB Upsizer module, CTI and CM3 debug logic.	LPSC1	POR	
	MAIN_RST	This is the warm reset for PMMC. Most of the remaining PMMC logic is on this reset. All PMMC Control MMRs are sensitive to this reset.	LPSC1	MOD_G_RST	
	CPU_LOCAL_RST	This is the local reset for PMMC CPU. Host CPU ensures that UMEM is loaded with right firmware before releasing this reset.	LPSC1	LRST	
MLB	MLB_SYS_MAIN_RST	Active Low, Async asserted, Sync deasserted reset signal to MLBSS, which resets all the logic except the AHB/APB portion of the MediaLB core.	LPSC11	MOD_G_RST	
	MLB_OCP_SPB_MAIN_RST	Active Low, Async asserted, Sync deasserted reset signal to MLBSS, which resets all the logic in the APB portion of the MediaLB core and the OCP2APB bridge.	LPSC11	MOD_G_RST	
	MLB_OCP_SHB_MAIN_RST	Active Low, Async asserted, Sync deasserted reset signal to MLBSS, which resets all the logic in the AHB portion of the MediaLB IP core and the AHB2OCP bridge.	LPSC11	MOD_G_RST	
DSS	DSS_RESET	Module Asynchronous Reset	LPSC23	MOD_G_RST	
McBSP	RESET	Module Asynchronous Reset	LPSC6	MOD_G_RST	
McASP_[0:2]	MCASP_VBUSP_ASYNC_RST	Module Asynchronous Reset	LPSC15	MOD_G_RST	
ASRC	MOD_G_RST	Module Asynchronous Reset	LPSC20	MOD_G_RST	
DCAN_[0:1]	VBUSP_RST	Module Asynchronous Reset	LPSC0	MOD_G_RST	
EMIF	CHIP_RST	Typically power-up reset. This resets the registers as well as the control logic in the EMIF.	Reset Controller or BOOT_CFG	CHIP_0_RST or DDR3A_PHY_RST bit	See BOOTCFG_DDR3A_PLL_CTL1 register
	MOD_G_RST	Typically warm reset. This only resets the control logic in the EMIF. It does not reset the internal registers.		LPSC27.MOD_G_RST or CHIP_1_RST	
	CTL_RST	Active low reset for the PUB logic and DDR PHY	Reset Controller or BOOT_CFG	CHIP_0_RST or DDR3A_PHY_RST bit	See BOOTCFG_DDR3A_PLL_CTL1 register
MMC[0:1]	RESETN	Asynchronous system reset	LPSC8	MOD_G_RST	
GPMC	GPMC_RST	Module Asynchronous Reset	LPSC9	MOD_G_RST	
ELM	RST	Module Asynchronous Reset	LPSC9	MOD_G_RST	
SPI[0:3]	RESET	Module Asynchronous Reset	LPSC6	MOD_G_RST	

Table 5-502. Reset Mapping and Description (continued)

Module	Module Reset Port	Description	Source	Source Port	Comments
ICSS_0/ICSS_1	MOD_G_RST	Module Asynchronous Reset	LPSC21	MOD_G_RST	
USB_0	MOD_G_RST	Module Asynchronous Reset	LPSC25	MOD_G_RST	
	POR	Module level power on reset (PHY Reset)	LPSC25	MOD_G_RST	
USB_1	MOD_G_RST	Module Asynchronous Reset	LPSC26	MOD_G_RST	
	POR	Module level power on reset (PHY Reset)	LPSC26	MOD_G_RST	
NSS	NAVSS_MOD_G_RST	Main SoC interface and NAVSS Reset. This reset controls the reset for all internal sub-modules of the NAVSS. It should always be de-asserted first or coincident with the resets for the SA and Ethernet. It should always be asserted last or coincident with the resets for SA and Ethernet.	LPSC3	MOD_G_RST	
	SA_UL_MOD_G_RST	SA Reset. This reset controls the reset for the SA. It should always be de-asserted after or coincident with the Main reset. It should always be asserted before or coincident with the Main reset.	LPSC4	MOD_G_RST	
	ESW_MOD_G_RST	EMAC Reset. This reset controls the reset for the EMAC.	LPSC3	MOD_G_RST	
PCIe	MOD_G_RST	Module Reset	LPSC24	MOD_G_RST	
	CHIP_RST	Power-on Reset		PSM13.MOD_POR	
SERDES	SERDES_VBUSP_RST	SerDes Reset	LPSC0	MOD_G_RST	
OTP	RESET	Module Reset	LPSC0	AON_RST	
GPIO	RST	Module Reset	LPSC0	AON_RST	
TIMER_0	VBUS_RST	TIMER_0 Module Reset (DSP WDT)	LPSC18	LRST	
TIMER_[1:4]	RST	TIMER_1-4 Module Resets (General Purpose Timers)	LPSC6	MOD_G_RST	
TIMER_5	RST	TIMER_5 Module Reset (ARMSS WDT)	ARM_DPSC	TIMER5_MOD_RST	
TIMER_6	RST	TIMER_6 Module Reset (wake-up timer for PMMC CPU)	LPSC0	AON_RST	
GTC	PWR_ON_RST	Module Reset	Reset Controller	POR	
SEC_MGR	POR	Power-on-Reset (POR)	Reset Controller	POR	
	EARLY_POR	Early Power-on-Reset	Reset Controller	POR_EARLY	
	MOD_G_RST	Main bus reset (warm reset)	LPSC0	MOD_G_RST	
	SEC_WRST	P1500 Reset		From Debugss Reset (corressponding Tap)	
	MOD_G_RST	EMIF register reset	LPSC27	MOD_G_RST	

Table 5-502. Reset Mapping and Description (continued)

Module	Module Reset Port	Description	Source	Source Port	Comments
MESSAGE MANAGER	MMGR_MOD_G_RST	Module Reset	LPSC0	MOD_G_RST	
BOOT_CFG	SYS_PORRST	Power On Reset	Reset Controller	POR_BOOT_CFG_RST	
	MOD_G_RST	Module Reset	Reset Controller	CHIP_1_RST	
DEBUGSS	JTAG_TRST	I/O Reset – JTAG TRSTn		TRSTn	Device Pin
	P1500STD_WRST	I/O Reset – P1500 WRSTn		TRSTn	Device Pin
	DSSRST_CHIP_RST	Stretched chip reset		LPSC2.MOD_G_RST or CHIP_0_RST	
	DSSRST_POR_EARLY	POR Early Reset		PSM1_LPSC2.POR	
	DSSRST_POR_BOOT_CFG_RST	Unstretched POR. Used to sample debug bootmodes.		PSM1_LPSC2.POR or POR_EARLY	In mission mode only PSM1_LPSC2.POR is used
	DBGCORE_DRST_[0:9]_DBGINRESET	Debug Reset Outputs		Connects to corresponding Module WRST	
	DBGCORE_DRST_[0:9]_DBGPOR	Debug Reset Outputs		Connects to corresponding Module WRST	
	DBGCORE_DRST_[0:9]_DBGUNNATURALRESET	Debug Reset Outputs		Connects to corresponding Module WRST	
UART_[0:2]	CBA_RST	Module Reset	LPSC6	MOD_G_RST	
ePWM_[0:5]	VBUS_RST	Module Reset	LPSC12	MOD_G_RST	
eQEP_[0:2]	VBUS_RST	Module Reset	LPSC13	MOD_G_RST	
eCAP_[0:1]	VBUS_RST	Module Reset	LPSC14	MOD_G_RST	
I2C_0	VBUS_RST	Module Reset	LPSC0	MOD_G_RST	
I2C_1, I2C_2	VBUS_RST	Module Reset	LPSC6	MOD_G_RST	
CP_TRACER (CPT_[0:14])	CP_TRACER_RST_MOD_G_RST[0:1]	Module Reset	LPSC17	MOD_G_RST	
	CP_TRACER_RST_MOD_G_RST[2:14]	Module Reset	LPSC5	MOD_G_RST	
MPU_[0:2] (Memory Protection Unit)	RST	Module Reset	LPSC0	MOD_G_RST	
MPU_[14]	RST	Module Reset	LPSC6	MOD_G_RST	
MPU_[3:13], MPU_[15:16]	RST	Module Reset	LPSC5	MOD_G_RST	
EDMA	TPTC_RESET	TPTC Reset	LPSC6	MOD_G_RST	
	TPCC_RESET	TPCC Reset	LPSC6	MOD_G_RST	
SEMAPHORE	MOD_G_RST	Module Reset	LPSC6	MOD_G_RST	
PSC	VBUS_RST	VBUS Reset	Reset Controller	CHIP_1_RST	

Table 5-502. Reset Mapping and Description (continued)

Module	Module Reset Port	Description	Source	Source Port	Comments
	CHIP_0_RST_EARLY	Chip 0 Early Reset. This signal synchronously resets everything in GPSC except registers. Reset Controller asserts this signal asynchronously whenever hard reset source is asserted, and de-asserts asynchronously when the hard reset source is de-asserted (at which point the Reset Controller is still stretching and holding CHIP_0_RST asserted). PSM synchronizes this input. For reset isolation power domains, POR_EARLY is used instead.	Reset Controller	CHIP_0_EARLY_RST	
	CHIP_1_RST_EARLY	Chip 1 Early Reset. Blockable by RSISO	Reset Controller	CHIP_1_EARLY_RST	
	CHIP_0_RST	Chip 0 Stretched Reset. This signal synchronously resets the registers to default values. Register default values are synchronously latched into the registers when this signal is de-asserted. Note that registers of Reset Isolation power domains are reset by POR_EARLY instead.	Reset Controller	CHIP_0_RST	
	CHIP_1_RST	Chip 1 Stretched Reset. Blockable by RSISO	Reset Controller	CHIP_1_RST	
	POR_EARLY	Power-On Reset Early. For Reset Isolation power domains, POR_EARLY resets most of the state machine and logic.	Reset Controller	POR_EARLY	
	POR	Power-On Reset Stretched. This signal is a stretched version of POR_EARLY. This signal is used to reset registers of Reset Isolation domains.	Reset Controller	POR	
INTC	MOD_G_RST	Module Reset	LPSC6	MOD_G_RST	
GIC	MOD_G_RST	Module Reset	LPSC6	MOD_G_RST	
QSPI	MOD_G_RST	Module Reset	LPSC7	MOD_G_RST	
ARMSS	POR_EARLY_RST	Early PORn reset for ARM_DPSC power state machine	Reset Controller	POR_EARLY	
	POR_RST	Power on reset for debug related logic in ARM_WRAP	Reset Controller	POR	
	CHIP_EARLY_RST	Reset for ARM_WRAP logic(vbus, non-debug bridges, and DPSC)	Reset Controller	POR_EARLY	
	CHIP_RST	Reset for EAGLENEST (MSMC side)	LPSC17	MOD_G_RST	MSMC Reset
	TETRIS_MOD_POR	Power on reset for ARMSS	PSM9	MOD_POR	
	TETRIS_MOD_RST	Warm reset for ARMSS	LPSC19	MOD_G_RST	
DSP	GRST	Global Reset	LPSC18	MOD_G_RST	
	LRST	Local Reset	LPSC18	LRST	

Table 5-502. Reset Mapping and Description (continued)

Module	Module Reset Port	Description	Source	Source Port	Comments
	POR_Z	Power on Reset	PSM8	POR	
	GEM_TRSTZ	Reset from Debugss		Reset from Debugss	
MSMC	MSMC_PARITY_RST_BLOCK	Boot config bit (block reset to Parity)	LPSC17	MOD_G_RST	Setting BOOTCFG_CHIP_MISC_CTL0[12] MSMC_BLOCK_PARITY_RST bit blocks this reset
	MOD_G_RST	Global Reset	LPSC17	MOD_G_RST	
DFT_SS	RESET	Module Reset	LPSC0	AON_RST	
CBASS	AON_CHIP_RST	Reset to always-on logic	LPSC0	MOD_G_RST	
	CHIP_RST	Module reset	LPSC5	MOD_G_RST	
	SCR and Bridge resets	Interface reset	LPSC5	MOD_G_RST	
EFUSE	FRESET	eFuse Reset	Reset Controller	FRESET	
	POR_EARLY_RST	Power On Reset	Reset Controller	POR_EARLY	
	VBUSP_RST	Module Reset	Reset Controller	AON_RST	
ETB (TETB)	EARLY_POR	Power On Reset	PSM1_LPSC2	POR	
	MOD_G_RST	Warm reset for vbusp interface only	LPSC2	MOD_G_RST	
TBR (DBG, PMMC, ARMSS)	POR	Power On Reset	PSM1_LPSC2	POR	
	VBUSP_RST	Warm reset for vbusp interface only	LPSC2/Reset Controller	LPSC2.MOD_G_RST or CHIP_0_RST	
ICEMELTER	PIDNTRST	TRSTn IO pin reset		TRSTn	TRSTn IO pin
	DBG_POR_RST	Power Domain Reset		PSM1_LPSC2.POR or POR_EARLY	In mission mode only PSM1_LPSC2.POR is used.
	POR_EARLY_RST	Power On Reset	Reset Controller	POR_EARLY	
	POR_BOOT_CFG_RST	POR Raw for EMU0/1 latching	Reset Controller	POR_BOOT_CFG_RST	

5.4 Clock Management

This chapter describes the clock architecture of the device.

NOTE: For a detailed visual representation of the distribution and management of the device clocks at the device level, see the clock interactive software (CTT) for the device.

See more about the CTT at http://www.ti.com/tool/clocktreetool

5.4.1 Overview

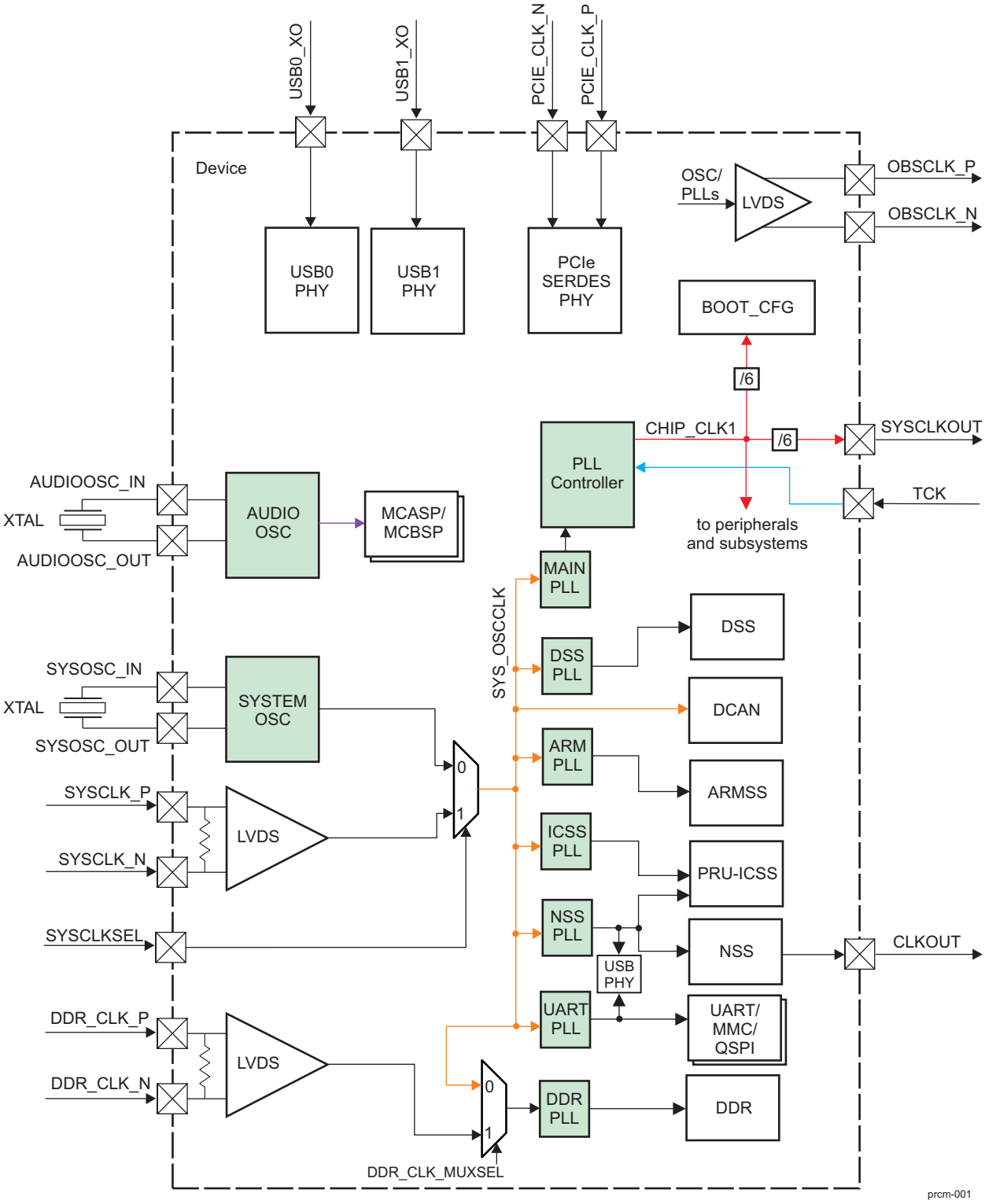
The device features multiple clock sources and clock generators (PLLs, PLL Controller, and dividers) to satisfy the various subsystems requirements. Most of the device is driven by the output from the MAIN PLL and MAIN PLL Controller except for the following modules and subsystems:

- Arm subsystem (ARMSS) has its own dedicated PLL (ARM PLL)
- DDR EMIF subsystem has its own dedicated PLL to drive EMIF and DDR PHY.
- PRU-ICSS receives clocks from several PLLs — UART PLL to generate constant 192-MHz clock, ICSS PLL to generate 225-MHz core clock and NSS/IEP PLL to generate 200-MHz ICSS core clock and 250-MHz Ethernet clocks.
- Display Subsystem (DSS) has its own dedicated PLL (DSS PLL), to generate the pixel clock
- PCIe requires separate external reference clock to drive SERDES PHYs
- USB PHYs support optional external reference clock input, additionally to UART PLL and NSS/IEP PLL
- MCASP and MCBSP modules support optional clock inputs to get required audio frequencies.

Figure 5-163 shows the device top-level block diagram including clock sources (pins and oscillators), clock outputs and PLLs.

Clock distribution is described in Table 5-557, *Module Clock Distribution*.

Figure 5-163. Top-Level Clock Diagram



5.4.2 Clock Inputs

Various external clock inputs are needed to drive the device. Summary of these input clock signals are:

- SYSOSC_IN/SYSOSC_OUT — external main crystal interface pins connected to internal oscillator which sources reference clock for all the PLLs. When internal oscillator is bypassed, reference clock input must be provided on SYSOSC_IN pin by an external clock source/generator.
- SYSCLK_P/N — optional pins to provide reference clock input to the PLLs.
- DDR_CLK_P/N — optional clock input to DDR3 PLL.
- AUDIOOSC_IN/AUDIOOSC_OUT — optional audio crystal interface pins connected to secondary internal oscillator dedicated for audio applications. See Section 5.4.4, *Audio Clocking*.
- PCIE_CLKP/N — SerDes reference clock for PCIe
- USB0_XO/USB1_XO — optional USB0/USB1 PHY reference clock input
- CPTS_REFCLKP/N — CPTS reference clock inputs

NOTE: Other clock inputs that are routed directly to the subsystems are described in the respective subsystem chapter.

The PLL reference clock (SYS_OSCCLK) can be either:

- internal high-frequency (HF) oscillator with external crystal: SYSOSC
- clock inputs provided by external clock sources

SYS_OSCCLK clock source is selected by means of SYSCLKSEL pin. The encoding of SYSCLKSEL is shown in Table 5-503.

Table 5-503. SYS_OSCCLK Clock Selection

SYSCLKSEL pin	SYS_OSCCLK Selection
0	HF OSC OUT (SYSOSC output)
1	SYSCLK_P/SYSCLK_N (LVDS clock input)

The status of SYSCLKSEL pin can be observed by software in the BOOT_CFG_PLLCLKSEL_STAT register.

Besides as a PLL reference clock, SYS_OSCCLK clock is used as the DCAN and GTC modules functional clock.

DDR PLL input reference clock is selected by another mux configured in the BOOTCFG_DDR_CLKCTL register (Table 5-504).

Table 5-504. DDR PLL Clock Selection

DDR_CLK_MUXSEL bit	DDR_CLK Selection
0 (default)	SYS_OSCCLK
1	DDR_CLK_P/DDR_CLK_N (LVDS clock input)

NOTE: SYSCLK_P/N and DDR_CLK_P/N are implemented with Low-Voltage Differential Signaling (LVDS) buffers. The system designer is responsible for using a compatible clock source/driver.

Device has the following clock oscillators built-in:

- internal high-frequency (HF) oscillator with an external crystal resonator, referred to as SYSOSC
- internal HF oscillator with an external crystal resonator, referred to as AUDIOOSC.

System oscillator clock (SYS_OSCCLK) clock frequencies supported by the device are:

- 19.2 MHz
- 24 MHz
- 25 MHz
- 26 MHz.

By default, SYSOSC is enabled and AUDIOOSC is disabled after POR. Boot Config module (BOOT_CFG) provides software means to enable/disable the two oscillators in the BOOTCFG_OSC_CTL register, as well as other oscillator settings.

NOTE: For input clock specifications, please refer to section *Clock Specifications* in the device specific Data Manual.

5.4.3 Clock Outputs

The device provides several system clock outputs. Summary of these output clock signals is as follows:

- OBSCLK_P/OBSCLK_N
- SYSCLKOUT
- CLKOUT

Other clock outputs, that are routed directly from subsystems to device pins, are described in the respective module chapter.

On the differential clock output OBSCLK_P/OBSCLK_N, oscillators and PLLs clocks can be observed for tests and debug. OBSCLK output is controlled by BOOTCFG_OBSCLKCTL register in the BOOT_CFG module.

NOTE: OBSCLK_P/OBSCLK_N output cannot be used to drive external circuitry. It is for debug only.

Table 5-505. OBS_CLK Clock Selection

BOOTCFG_OBSCLKCTL[3-0] PLL_OBSCLK_SEL	OBS_CLK Selection
0x0 (default)	Reserved
0x1	MAIN PLL
0x2	DSS PLL
0x3	ARM PLL
0x4	UART PLL
0x5	ICSS PLL
0x6	DDR PLL
0x7	PLL Controller OBSCLK
0x8	NSS PLL
0x9	SYSOSC clock

SYSCLKOUT provides SYSCLK1 coming from the DPLL Controller for test and debugging. It is divided by 6 inside the device and then sent out as a LVCMOS clock signal. This signal can be used to test if the main chip clock is running or not. Output of SYSCLKOUT may be disabled or enabled by user. See BOOTCFG_DEVCFG register.

CLKOUT provides an option to output 50-MHz or 25-MHz clock. This clock can be used as a reference clock for RMII or MII/GMII Ethernet mode. Refer to Section 11.13.4, *Gigabit Ethernet MAC (EMAC) Subsystem* and BOOTCFG_ETHERNET_CLKCTL register.

There is no dedicated ball for CLKOUT function on the device. Instead, it is available as a pin muxing option. See the device Data Manual for device pin mux details and BOOTCFG_PADCONFIG for pin mux registers.

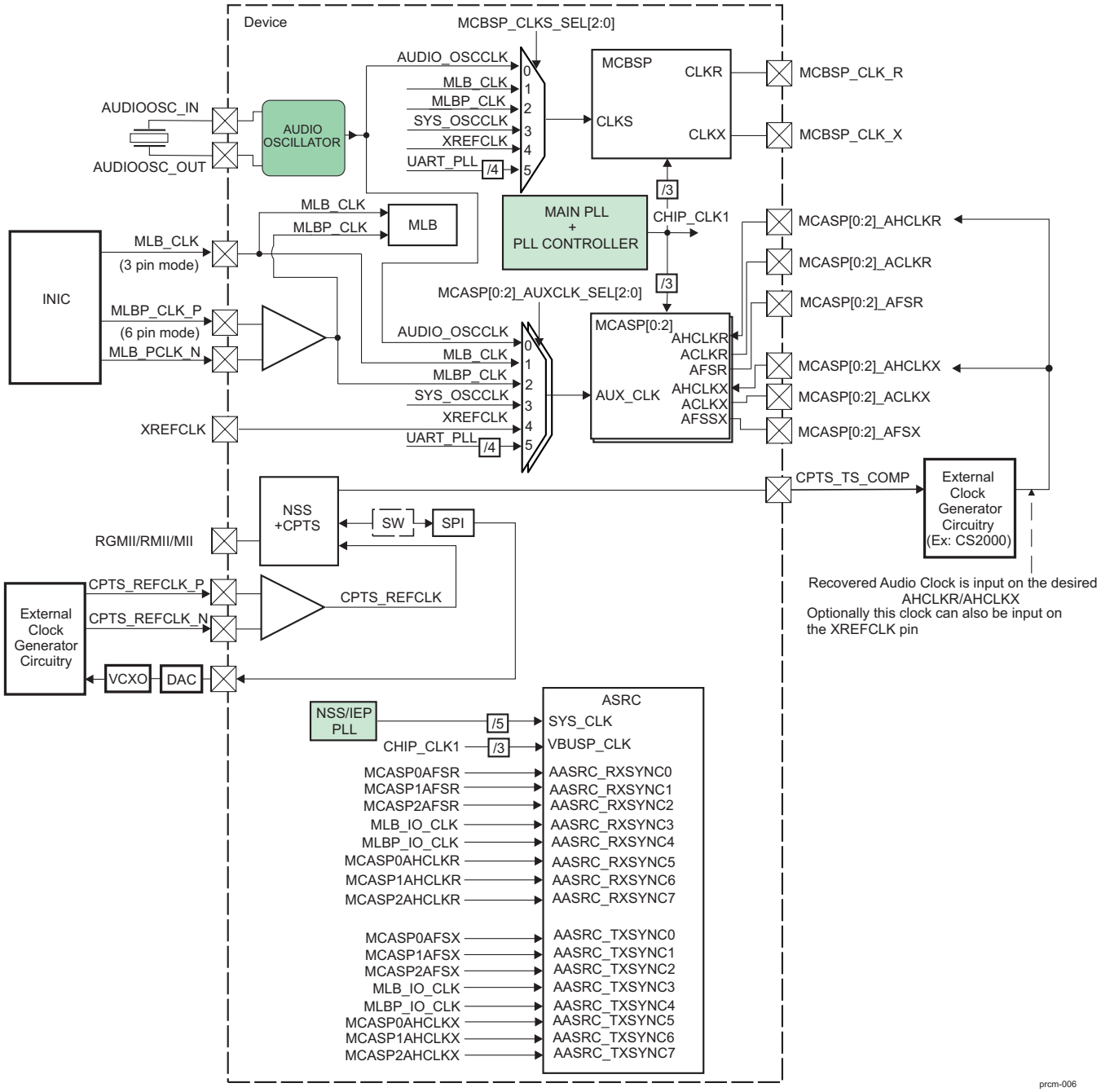
NOTE: For output clock specifications, please refer to section *Clock Specifications* in the device specific Data Manual.

5.4.4 Audio Clocking

Audio peripherals can be driven by dedicated HF audio oscillator (AUDIO_OSC). The supported frequency range for audio oscillator is 11.289 MHz to 49.152 MHz.

Figure 5-164 summarizes the audio clocking scheme of the device.

Figure 5-164. Device Audio Modules Clocking



Clock muxes on Figure 5-164 are controlled through the BOOTCFG_SERIALPORT_CLKCTL register.

NOTE: For audio peripherals description, see the respective peripheral chapter.

5.4.5 PLLs

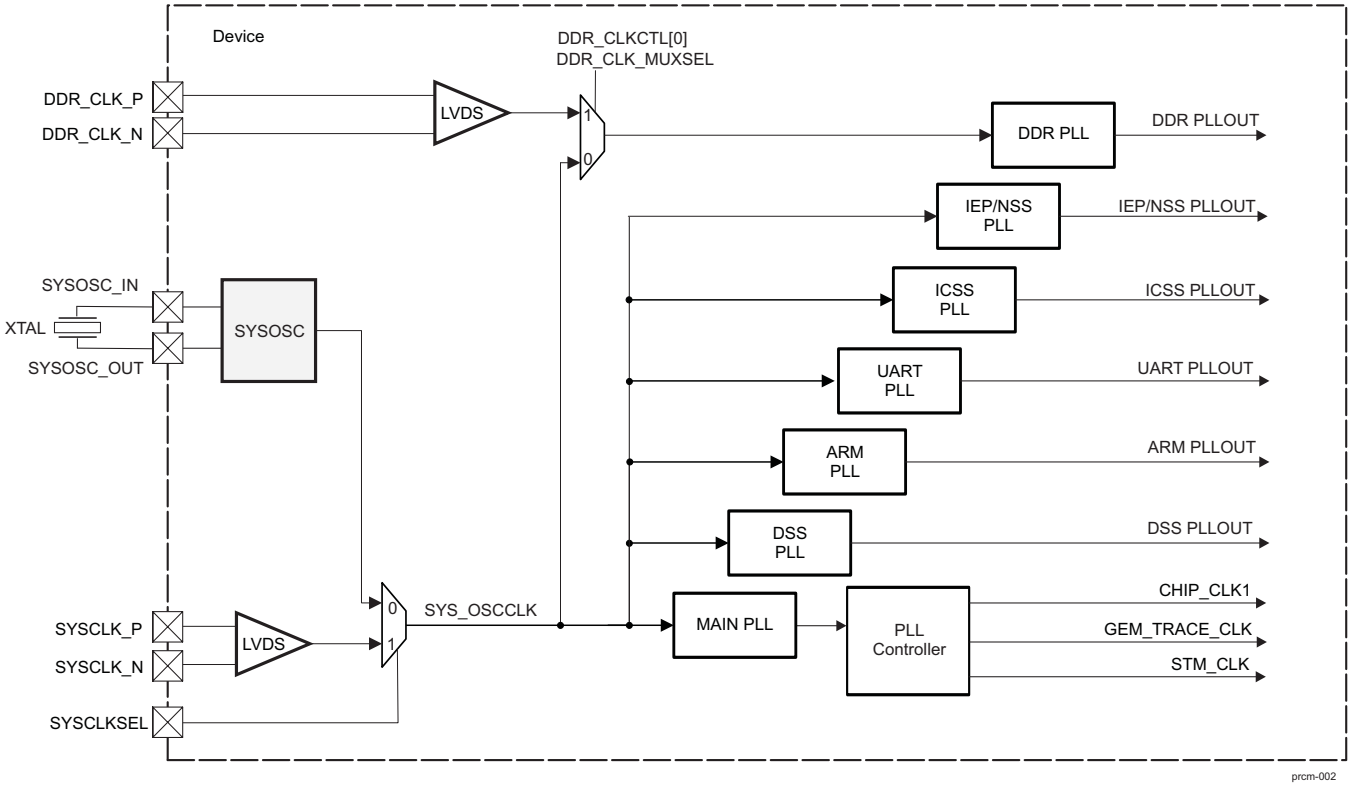
Phase-Locked Loop circuitries (PLLs) in the device are clock generator PLLs, which multiply the lower-frequency reference clock up to the operating frequency of the respective subsystem(s). There are total seven PLLs in the device:

- MAIN PLL with PLL_CONTROLLER

- ARM PLL
- DSS PLL
- UART PLL
- ICSS PLL
- NSS/IEP PLL
- DDR PLL.

Overview of the device PLLs with their reference clock options is shown on Figure 5-165.

Figure 5-165. PLLs Integration



The ARM PLL, DSS PLL, UART PLL, ICSS PLL, NSS PLL, and DDR PLL are used to provide dedicated clocks to:

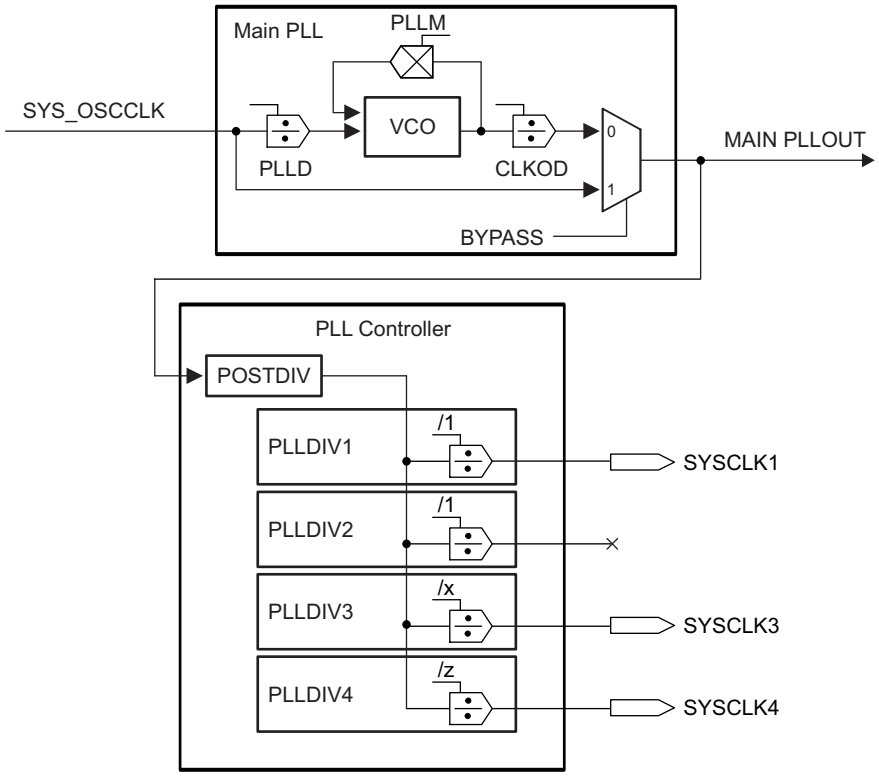
- ARM PLL: ARMSS
- DSS PLL: Display subsystem (DSS) pixel clock
- UART PLL: UARTs (including ICSS UARTs), MMC, USB PHY, QSPI
- ICSS PLL: ICSS PRU cores
- NSS/IEP PLL: NSS clocks, USB PHY, ICSS PRU cores, ICSS IEP
- DDR PLL: DDR3 EMIF and PHY

These chip level PLLs support a wide range of multiplier and divider values, which can be programmed through the chip level registers located in the BOOT_CFG and starting at BOOTCFG_NSS_PLL_CTL0 register address. The Boot ROM programs the multiplier values for MAIN PLL, ARM PLL, NSS PLL, UART PLL and DDR PLL based on the current boot mode. See Chapter 4, *Initialization* for more details on boot mode selection.

5.4.5.1 MAIN PLL and PLL Controller Overview

The MAIN PLL is controlled by the PLL Controller. The PLL Controller manages the clock ratios, alignment, and gating for the system clocks to the device. By default, the device powers up with the MAIN PLL bypassed. Figure 5-166 shows a block diagram of the MAIN PLL and the PLL Controller. For details on the operation of the PLL Controller module, see Section 5.4.5.3, *PLL Controller*.

Figure 5-166. MAIN PLL and PLL Controller



The multiplication and division ratios within the PLL and the post-division for each of the chip-level clocks are determined by a combination of this PLL and the PLL Controller. The PLLM[5-0] bits of the multiplier are controlled by the PLLM register inside the PLL Controller and the PLLM[12-6] bits are controlled by the chip-level BOOTCFG_MAIN_PLL_CTL0 register. The output divide and bypass logic of the PLL are controlled by fields in the SECCTL register in the PLL Controller. Only PLLDIV3, and PLLDIV4 are programmable on the device. See the Section 5.4.5.3.3, *PLL/PLL Controller Programming Guide* for more details on how to program the MAIN PLL and PLL controller.

NOTE: MAIN PLL Controller registers can be accessed by any master in the device.

The MAIN PLL Controller also controls reset propagation through the chip, clock alignment, and test points. The PLL Controller monitors the MAIN PLL status and provides an output signal indicating when the PLL is locked.

5.4.5.2 MAIN PLL Controller Device-Specific Information

5.4.5.2.1 Clocks Generated by PLL Controller

The MAIN PLL, used to drive the DSP CorePac, the switch fabric, and a majority of the peripheral clocks, requires a PLL Controller to manage the various clock divisions, gating, and synchronization. Output divider ratio is set in the SECCTL[22-19] OUTPUT_DIVIDE bitfield. Boot ROM sets this value during boot sequence based on MAINPLL_OD_SEL pin value. See BOOTCFG_DEVSTAT register in Section 5.1, *Control Module (BOOT_CFG)* for its description.

PLLM[5-0] input of the MAIN PLL is controlled by the PLL controller PLLM register.

The MAIN PLL Controller has four SYSCLK outputs that are listed below, along with the clock descriptions. Each SYSCLK has a corresponding divider that divides down the output clock of the PLL. Note that dividers only for SYSCLK3 and SYSCLK4 are programmable.

- **SYSCLK1:** Full-rate clock for C66x DSP CorePac. Using local dividers, SYSCLK1 is used to derive clocks required for the majority of peripherals. SYSCLK1 must be configured by software (ROM code) to be reset isolated to protect the modules that require reset isolation. That is, SYSCLK1 must maintain its frequency without pausing through non-POR resets.

  See Section 5.4.6, *Module Clock Distribution* for the complete list of peripherals fed by this clock.
- **SYSCLK2:** Unused
- **SYSCLK3:** 1/x-rate clock used to clock the C66x DSP CorePac emulation. The default rate for this clock is /3. This clock is programmable from /1 to /32, where this clock does not violate the maximum of 333 MHz. SYSCLK3 can be turned off by software.
- **SYSCLK4:** 1/z-rate clock for the system trace module only. The default rate for this clock is /5. This clock is configurable: the maximum configurable clock is 200 MHz and the minimum configuration clock is 32 MHz. SYSCLK4 can be turned off by software.

NOTE: SYSCLK names are defined as the PLL Controller outputs. Once leaving the PLL Controller, the following clock names apply at the device level:

- SYSCLK1: CHIP_CLK1, drives C66x DSP CorePac and most of the peripherals through local dividers
- SYSCLK3: GEM_TRACE_CLK, used for emulation
- SYSCLK4: STM_CLK, used for trace module.

5.4.5.2.2 MAIN PLL Controller Operating Modes

The MAIN PLL Controller has two modes of operation: bypass mode and PLL mode. The mode of operation is determined by the BYPASS bit of the PLL Secondary Control Register (SECCTL).

- In bypass mode, PLL input is fed directly out as SYSCLK1.
- In PLL mode, SYSCLK1 is generated from the PLL output using the values set in the PLLM and PLLD fields in the BOOTCFG_MAIN_PLL_CTL0 register.

External hosts must avoid access attempts to the DSP while the frequency of its internal clocks is changing. User software must implement a mechanism that causes the DSP to notify the host when the PLL configuration has completed.

5.4.5.2.3 MAIN PLL Stabilization, Lock, and Reset Times

The PLL stabilization time is the amount of time that must be allotted for the internal PLL regulators to become stable after device power-up. The device must not be taken out of reset until this stabilization time has elapsed.

The PLL reset time is the amount of wait time needed when resetting the PLL (writing PLLRST = 1), in order for the PLL to properly reset, before bringing the PLL out of reset (writing PLLRST = 0). For the MAIN PLL reset time value, see Table 5-506.

The PLL lock time is the amount of time needed from when the PLL is taken out of reset to when the PLL Controller can be switched to PLL mode. The MAIN PLL lock time is given in Table 5-506.

Table 5-506. MAIN PLL Stabilization, Lock, and Reset Times

PARAMETER	MIN	TYP	MAX	UNIT
PLL stabilization time	100			µs
PLL lock time			2000 × C[(1)]	
PLL reset time	1000			ns

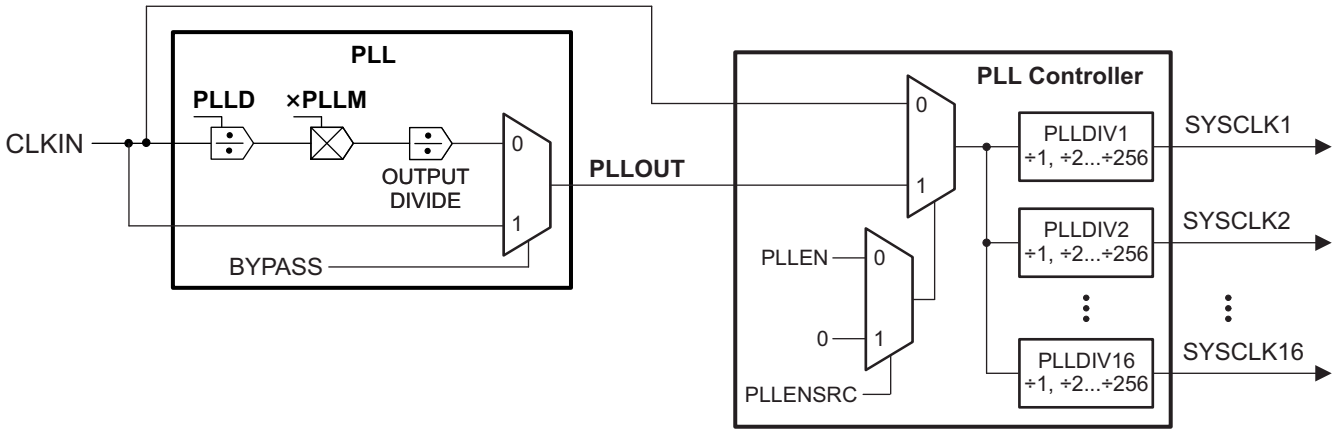
(1) C = SYSCLK1(N|P) cycle time in ns.

5.4.5.3 PLL Controller

5.4.5.3.1 PLL Controller Overview

Figure 5-167 shows the logical implementation of the PLL and the PLL Controller. The PLL Controller offers flexibility and convenience with software-configurable dividers (PLLDIV3 and PLLDIV4 in this device) to modify the input clock signal internally. The PLL Controller also contains registers (PLLM and SECCTL) that are used to drive the PLLM, OUTPUT DIVIDE, and BYPASS logic of the PLL (see Figure 5-167). The resulting clock outputs from the PLL Controller are passed to the DSP core, peripherals, and other modules inside the device.

Figure 5-167. PLL and PLL Controller Generic Block Diagram



The PLL Controller has the following input and output clocks:

- Input reference clock to the PLL Controller:
  - **PLLOUT:** Output signal from the PLL
  - **CLKIN:** Input reference clock to the PLL
- Resulting output clocks from the PLL Controller:
  - **SYSCLK1 to SYSCLK4:** System domain clocks, each output from its own divider (see Figure 5-167)

5.4.5.3.2 PLL Controller Functional Description

5.4.5.3.2.1 Dividers

The clock dividers (PLLDIV3 and PLLDIV4 in this device) are programmable in a range from ÷1 to ÷256 and may be disabled. When a clock divider is disabled, no clock is output from that clock divider. A divider outputs a clock only when it is enabled in the corresponding PLLDIVn register.

PLLD is a divider inside the PLL. BOOTCFG_MAIN_PLL_CTL0[5-0] register controls this divider.

5.4.5.3.2.2 Multiplier

The PLLM[5-0] bits of the multiplier are controlled by the PLLM register inside the PLL Controller. The PLLM[12-6] bits are controlled by the chip-level BOOTCFG_MAIN_PLL_CTL0 register. PLLM[12-6] must be written just before writing to PLLM[5-0] bits of the PLLM register to have the complete 13-bit value latched when the GO operation is initiated in the PLL controller.

5.4.5.3.2.3 PLL Control Register and Secondary Control Register

After device reset, the value of the PLL enable bit (PLLEN) in the PLL control register (PLLCTL) can be changed, but it will not have any effect on the function of the PLL Controller. To enable the PLLEN bit, the PLLENSRC bit (also in the PLLCTL register) must first be cleared to 0. Once the PLLEN bit has been enabled, it can be used to select the bypass mode or PLL mode of the PLL Controller as discussed in the next two sections. The PLLRST bit in PLLCTL is used to reset the PLL Controller.

The Secondary Control register (SECCTL) is used to drive the OUTPUT DIVIDE and BYPASS logic of the PLL.

The PLL can be operated in Bypass or PLL mode based on the status of the BYPASS, PLLENSRC, and PLLEN bits in the PLLCTL and SECCTL registers and is discussed in the next two sections.

5.4.5.3.2.4 Bypass Mode

When BYPASS = 1 (bypass enabled in the PLL Mux), that is, in bypass mode, the PLLM, PLLD, and OUTPUT DIVIDE logic of the PLL are bypassed and the input reference clock to the PLL (CLKIN) is input directly to the PLL Controller (see Figure 5-167). The PLL block is operating in Bypass Mode.

When PLLENSRC=0 and PLLEN=0 (bypass enabled in the PLL Controller mux), the entire PLL block is bypassed and the reference input from the PLL is fed as a direct input to the PLL Controller. The PLL Controller block is operating in Bypass Mode.

CAUTION

The PLL comes up in Bypass mode by default on powering up the device. Once the PLL is initialized in PLL mode, it must not be re-initialized back to Bypass mode unless the user intends to power down the device.

5.4.5.3.2.5 PLL Mode

When BYPASS = 0 (in PLL Mux), that is, in PLL mode, the PLLM, PLLD, and OUTPUT DIVIDE logic of the PLL are used (see Figure 5-167, *PLL and PLL Controller Block Diagram*). The output of the PLL (PLLOUT) is fed as an input to the PLL Controller. The PLL block is operating in PLL Mode.

When PLLENSRC=0 and PLLEN=1 (in the PLL Controller mux), the output of the PLL (PLLOUT) is fed as an input to the PLL Controller (see Figure 5-167). The PLL Controller block is operating in PLL Mode.

Further, when enabled (DnEN = 1), the system clock dividers (D1-D16) divide down by the RATIO value in PLLDIVn, the output clock of the PLL. The system clock dividers generate a 50% duty cycle output clock SYSCLKn.

5.4.5.3.3 PLL/PLL Controller Programming Guide

5.4.5.3.3.1 MAIN PLL Initialization

The MAIN PLL and PLL Controller are initialized by Boot ROM/software after reset. The PLL Controller registers must be modified only by processor or emulation. External masters, for example PCIe, must not be used to access the PLL Controller registers directly. For this purpose, a Boot ROM parameter table is provided to set the PLLs. See Chapter 4, *Initialization* for details.

The initialization of the PLL Controller must be performed as soon as possible at the beginning of the program, before initializing any peripherals. Upon device reset, the following software initialization procedures must be done to properly set up the PLL and PLL Controller.

NOTE: PLL configuration registers (BOOTCFG_MAIN_PLL_CTL0 and BOOTCFG_MAIN_PLL_CTL1) are write-protected at power-up. Software must first un-lock the BOOTCFG_KICK0 and BOOTCFG_KICK1 registers prior to writing to any chip-level registers.

5.4.5.3.3.1.1 MAIN PLL Initialization to PLL Mode

The MAIN PLL and PLL Controller must always be initialized prior to initializing the secondary PLLs. This section shows the initialization sequence if the user intends to use the PLL in PLL mode. The steps below also show when the multipliers, divider, and system clock dividers must be programmed, if required.

The following generic equation describes the MAIN PLL output when initialized to PLL mode:

PLLOUT = CLKIN × ((PLLM+1) ÷ ((OUTPUT_DIVIDE+1) × (PLLD+1)))

NOTE: For validated set of parameters that can be programmed to PLLs, please refer to the device-specific Data Manual.

NOTE: Once the ROM boot is complete, it is highly recommended that initial software reconfigures the MAIN PLL to the desired frequency, even if it is already achieved by the ROM settings. To minimize the overall output jitter, the PLLs must be operated as close as possible to the maximum operating frequency. To maximize the VCO frequency within the PLL, the PLL must be clocked to 2× the intended frequency and the PLL Output Divider must be set to divide-by-2 as described in the below sequence.

Perform each step in sequence unless directed to jump directly to another step.

1. If executing this sequence immediately after device power-up, user must wait for the PLL to become stable. PLL stabilization time is 100 µs.
2. Check the status of BYPASS bit in SECCTL register, execute following steps if BYPASS = 1 (bypass enabled); if BYPASS = 0 then **Jump to Step 3**
   a. In MAIN_PLL_CTL1, write ENSAT = 1 (for optimal PLL operation)
   b. In PLLCTL, write PLLEN = 0 (bypass enabled in PLL Controller mux)
   c. In PLLCTL, write PLLENSRC = 0 (enable PLLEN to control PLL Controller mux)
   d. Wait 4 cycles of the reference clock CLKIN (to make sure the PLL Controller mux switches properly to the bypass)
   e. In SECCTL, write BYPASS = 1 (bypass enabled in PLL mux)
   f. In PLLCTL, write PLLPWRDN = 1 (power down the PLL)
   g. Wait for at least 5 µs based on the reference clock CLKIN (PLL power down toggling time)
   h. In PLLCTL, write PLLPWRDN = 0 (power up the PLL).
3. In PLLCTL, write PLLRST = 1 (PLL is reset)
4. PLLM is split in two different registers. Program PLLM[5-0] in PLL multiplier control register (PLLM) and PLLM[12-6] in BOOTCFG_MAIN_PLL_CTL0 register
5. BWADJ is split in two different registers. Program BWADJ[7-0] in BOOTCFG_MAIN_PLL_CTL0 and BWADJ[11-8] in BOOTCFG_MAIN_PLL_CTL1 register. BWADJ value should be set to: *BWADJ = ((PLLM+1) >> 1) - 1*.
6. Program PLLD in BOOTCFG_MAIN_PLL_CTL0 register
7. In SECCTL, write OD (Output Divide) = 1 (that is divide-by-2)
8. This step is needed only if default ratio values in PLLDIVn registers (control SYSCLKn clocks) need to be changed. Otherwise **Jump to Step 9**. Note that in this device, only SYSCLK2 and SYSCLK3 are programmable.
   a. Check that the GOSTAT bit in PLLSTAT is cleared to show that no GO operation is currently in progress.
   b. Program the RATIO field in PLLDIVn to the desired new divide-down rate. If the RATIO field changed, the PLL controller will flag the change in the corresponding bit of DCHANGE.
   c. Set the respective ALNn bits in ALNCTL to align any SYSCLKs after the GO operation.
   d. Set the GOSET bit in PLLCMD to initiate the GO operation to change the divide values and align the SYSCLKs as programmed.
   e. Read the GOSTAT bit in PLLSTAT to make sure the bit returns to 0 to indicate that the GO

operation has completed.

9. Wait for at least 7 µs based on the reference clock CLKIN (PLL reset time)
10. In PLLCTL, write PLLRST = 0 (PLL reset is de-asserted)
11. Wait for at least *500 × CLKIN cycles × (PLLD + 1)* (PLL lock time)
12. In SECCTL, write BYPASS = 0 (enable PLL mux to switch to PLL mode)
13. In PLLCTL, write PLLEN = 1 (enable PLL Controller mux to switch to PLL mode)
14. At this point, the MAIN PLL is properly initialized.

CAUTION

Software must always perform read-modify-write to any register in the PLL and PLL Controller. This is to ensure that only the relevant bits in the register are modified and the rest of the bits including the reserved bits are not affected.

5.4.5.3.3.1.2 MAIN PLL Initialization to Bypass Mode

CAUTION

The PLL comes up in bypass mode by default on powering up the device. Once the PLL is initialized in PLL mode, it must not be re-initialized back to bypass mode unless the user intends to power down the device.

5.4.5.3.3.2 MAIN PLL Power Down

The PLL may be powered down, in which case the PLL is in bypass mode and the DSP runs from a divided-down version of the input reference clock. The DSP is able to respond to events because it is still being clocked by the bypass clock (directly from CLKIN), although at a lower frequency.

Perform the following procedure to power down the PLL:

1. In SECCTL, write BYPASS = 1 (bypass mode).
2. Wait 4 cycles of the slower of PLLOUT or CLKIN.
3. In PLLCTL, write PLLPWRDN = 1 to power down the PLL.

The above sequence assumes that the device has been powered up long enough that the PLL stabilization time has been met. If executing this sequence immediately after device power-up, user must allow time for the PLL to become stable before performing these steps. For PLL stabilization time, see Table 5-506.

5.4.5.3.3.3 MAIN PLL Wake Up

Perform the following procedure to wake up the PLL from its power-down mode.

1. In SECCTL, write BYPASS = 1 (bypass mode).
2. In PLLCTL, write PLLPWRDN = 0 to wake up the PLL.
3. Follow the PLL reset sequence in Section 5.4.5.3.3.1.1, *MAIN PLL Initialization to PLL Mode* (steps 3 to 13) to reset the PLL. Wait for the PLL to lock and to switch from bypass to PLL mode.

5.4.5.3.3.4 Secondary PLLs Initialization

Secondary PLLs (NSS/IEP PLL, DSS PLL, ARM PLL, UART PLL, ICSS PLL and DDR PLL) do not have their PLL Controllers. They are controlled through BOOT_CFG registers starting at BOOTCFG_NSS_PLL_CTL0 address.

NOTE: For validated set of parameters that can be programmed to PLLs, please refer to the device-specific Data Manual.

Initialization steps for secondary PLLs:

1. At power-up, all secondary PLLs operate in bypass mode.
2. Then, PLLs are properly initialized by Boot ROM software depending on the selected boot mode. Once the PLLs are stabilized with correct frequency settings the PLL BYPASS Mux control is updated by Boot ROM to pass through the PLL clocks to the respective modules. See Section 5.4.5.3.3.4.1, *Secondary PLLs Initialization in PLL Mode* for PLL initialization sequence.

Modules using non-MAIN PLL clocks treat these clocks as asynchronous. For example, ICSS uses clocks from UART PLL, ICSS PLL, NSS PLL and MAIN PLL. All these clocks are asynchronous to each other. Note that PLL sequence must always be executed before powering up their module's power domains.

CAUTION

The DDR interface needs to reset every time the DDR PLL is re-programmed.

5.4.5.3.3.4.1 Secondary PLLs Initialization in PLL Mode

The MAIN PLL and PLL Controller must always be initialized prior to initializing the secondary PLLs. The sequence shown below must be followed to initialize the secondary PLLs.

NOTE: Once the ROM boot is complete, reconfiguring the PLL with the intention of optimizing and reducing output jitter may optionally be done, even if the PLL is already configured by the ROM code. To minimize the overall output jitter, the PLL VCO output must be operated close to, but not above, the maximum operating frequency of 4.2 GHz. Maximizing the VCO frequency within the PLL is done by setting the PLL Output Divider as close as possible to 15 (OUTPUT_DIVIDE+1 must be an even number only). The PLLD value must be kept at 0 or as low as possible. The target PLLOUT frequency is therefore achieved by adjusting PLLM.

PLL VCO Frequency = CLKIN / (PLLD+1) × (PLLM+1)

PLL OUT Frequency = VCO / (OUTPUT_DIVIDE+1)

NOTE: In the below sequence, only the generic names of PLL_CTL0 and PLL_CTL1 registers are used. User must add the appropriate prefix as follows:

- BOOTCFG_NSS_
- BOOTCFG_DDR3A_
- BOOTCFG_ARM_
- BOOTCFG_DSS_
- BOOTCFG_ICSS_
- BOOTCFG_UART_

Secondary PLL registers start at BOOTCFG_NSS_PLL_CTL0 address.

1. In PLL_CTL1, write ENSAT = 1 (for optimal PLL operation).
2. In PLL_CTL0, write BYPASS = 1 (set the PLL in Bypass).
3. Program PLLM and PLLD in the PLL_CTL0 register.
4. Program BWADJ[7-0] in PLL_CTL0 and BWADJ[11-8] in the PLL_CTL1 register. BWADJ[11-0] must be programmed to a value related to PLLM[12-0] value based on the equation:
   BWADJ = ((PLLM+1) >> 1) - 1
5. In PLL_CTL1, write PLLRST = 1 (PLL reset is asserted).

6. Wait for at least 5 µs based on the reference clock (PLL reset time).
7. In PLL_CTL1, write PLLRST = 0 (PLL reset is de-asserted).
8. Wait for at least *500 × REFCLK cycles × (PLLD + 1)* (PLL lock time).
9. In PLL_CTL0, write BYPASS = 0 (switch to PLL mode).
10. PLL is now initialized.

5.4.5.3.3.4.2 USB Core Clock

USB module supports various CLKCORE frequencies: 12-, 19.2-, 24-, and 50-MHz. To generate these frequencies, clock outputs from UART PLL or NSS PLL can be selected. This is done by programming USB0/1_CLKCORE_SEL bit in the BOOTCFG_USB0_CLKCTL and BOOTCFG_USB1_CLKCTL registers. There is a programmable 5-bit divider at the output of this mux to generate the correct USB CLKCORE frequency. After setting the appropriate divide value, USB0/1_CLKCORE_LOAD_DIV bit must be set to 1 in to load the divider value into the clock divider. USB Core clock dividers must be configured prior to releasing USB out of reset.

The sequence to configure USB clock core divider is the following:

1. USB_0/1 module must be held in reset
2. Wait till NSS PLL or UART PLL is out of bypass mode
3. USB0/1_CLKCORE_SEL bit must be configured to select the appropriate source (NSS PLL or UART PLL). Reset default is UART PLL clock
4. Set the USB0/1_CLKCORE_DIV field to appropriate value to get (12/19.2/24/50 MHz)
5. Program the USB0/1_CLKCORE_LOAD_DIV bit to 0
6. Program the USB0/1_CLKCORE_LOAD_DIV bit to 1
7. Release USB_0/1 from reset.

5.4.5.3.3.5 Divider n and GO Operation

The GO operation is required to change the divider ratios of the PLLDIVn registers. Section 5.4.5.3.3.5.1, *GO Operation* discusses the GO operation. Section 5.4.5.3.3.5.2, *Software Steps to Modify PLLDIVn Ratios* provides the software steps required to change the divider ratios.

5.4.5.3.3.5.1 GO Operation

The GO operation writes to the RATIO field in the PLLDIVn. Registers do not change the dividers' divide ratios immediately. The PLLDIV dividers change to the new RATIO rates only during a GO operation. This section discusses the GO operation and alignment of the SYSCLKs.

The PLL Controller clock align control register (ALNCTL) determines which SYSCLKs must be aligned. Before a GO operation, program ALNCTL so that the appropriate clocks are aligned during the GO operation.

A GO operation is initiated by setting the GOSET bit in PLLCMD to 1. During a GO operation:

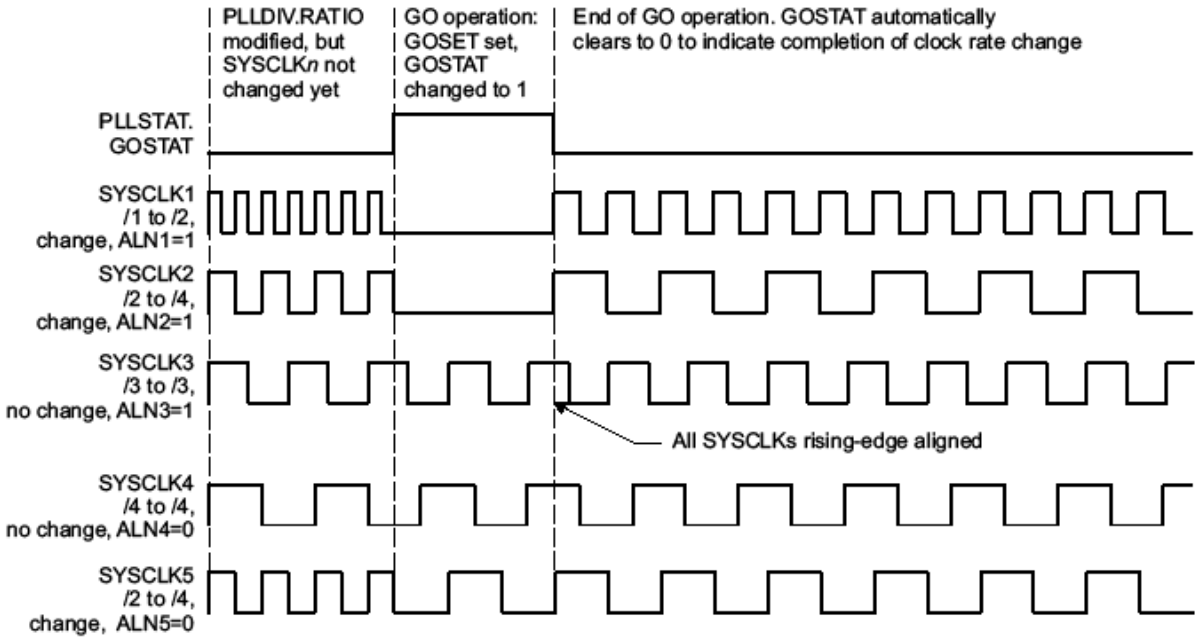
- Any SYSCLK *n* with the corresponding ALN *n* bit in ALNCTL and SYS *n* bit in DCHANGE set to 1 is paused at the low edge. Then the PLL Controller restarts all these SYSCLKs simultaneously, aligned at the rising edge. When the SYSCLKs are restarted, SYSCLK *n* toggles at the rate programmed in the RATIO field in PLLDIVn.
- Any SYSCLK *n* with the corresponding ALN *n* bit in ALNCTL cleared and the SYS *n* bit in DCHANGE set immediately changes to the new rate programmed in the RATIO field.
- The GOSTAT bit in PLLSTAT is set throughout the duration of a GO operation.

CAUTION

To help prevent errors, all device operation must be stopped before the GO operation.

Figure 5-168 shows an example of how the clocks are rising-edge aligned during a GO operation. Notice that because SYSCLK5 does not need to be aligned with the other clocks, it immediately switches to its new ratio during the GO operation.

Figure 5-168. Example Clock Ratio Change and Alignment with GO Operation



5.4.5.3.3.5.2 Software Steps to Modify PLLDIV Ratios

Perform the following steps to modify PLLDIV.

1. Check that the GOSTAT bit in PLLSTAT is cleared to show that no GO operation is currently in progress.
2. Program the RATIO field in PLLDIVn to the desired new divide-down rate. If the RATIO field changed, the PLL Controller will flag the change in the corresponding bit of DCHANGE.
3. Set the respective ALN *n* bits in ALNCTL to align any SYSCLKs after the GO operation.
4. Set the GOSET bit in PLLCMD to initiate the GO operation to change the divide values and align the SYSCLKs as programmed.
5. Read the GOSTAT bit in PLLSTAT to make sure the bit returns to 0 to indicate that the GO operation has completed.

5.4.5.4 PLL Controller Registers

The PLL Controller registers are listed in Table 5-508. All other register offset addresses not listed in Table 5-508 must be considered as reserved locations and the register contents must not be modified.

Table 5-507. PLL Controller Instances

Instance	Base Address
PLL	0231 0000h

Table 5-508. PLL Controller Registers

Offset	Acronym	Register Name	PLL Physical Address	Section
E4h	RSTYPE	Reset Type Status Register	0231 00E4h	Section 5.4.5.4.13
E8h	RSCTRL	Reset Control Register	0231 00E8h	Section 5.4.5.4.14
ECh	RSCFG	Reset Configuration Register	0231 00ECh	Section 5.4.5.4.15
F0h	RSISO	Reset Isolation Register	0231 00F0h	Section 5.4.5.4.16
100h	PLLCTL	PLL Control Register	0231 0100h	Section 5.4.5.4.1
108h	SECCTL	PLL Secondary Control Register	0231 0108h	Section 5.4.5.4.2
110h	PLLM	PLL Multiplier Control Register	0231 0110h	Section 5.4.5.4.3
118h	PLLDIV1	PLL Controller Divider 1 Register	0231 0118h	Section 5.4.5.4.4
11Ch	PLLDIV2	PLL Controller Divider 2 Register	0231 011Ch	Section 5.4.5.4.5
120h	PLLDIV3	PLL Controller Divider 3 Register	0231 0120h	Section 5.4.5.4.6
138h	PLLCMD	PLL Controller Command Register	0231 0138h	Section 5.4.5.4.8
13Ch	PLLSTAT	PLL Controller Status Register	0231 013Ch	Section 5.4.5.4.9
140h	ALNCTL	PLL Controller Clock Align Control Register	0231 0140h	Section 5.4.5.4.10
144h	DCHANGE	PLLDIV Divider Ratio Change Status Register	0231 0144h	Section 5.4.5.4.11
150h	SYSTAT	Sysclk Status Register	0231 0150h	Section 5.4.5.4.12
160h	PLLDIV4	PLL Controller Divider 4 Register	0231 0160h	Section 5.4.5.4.7

5.4.5.4.1 PLLCTL Register (Offset = 100h) [reset = 0h]

The PLL control register (PLLCTL) is shown in Figure 5-169 and described in Table 5-510.

Table 5-509. PLLCTL Instances

Instance	Physical Address
PLL	0231 0100h

Figure 5-169. PLLCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0							
15	14	13	12	11	10	9	8
RESERVED							
R/W-1							
7	6	5	4	3	2	1	0
RESERVED		PLLENSRC	RESERVED	PLLRST	RESERVED	PLLPWRDN	PLLEN
R/W-1		R-1	R-0	R/W-1	R-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 5-510. PLLCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R/W	1	Reserved.
5	PLLENSRC	R	1	PLL enable source bit. • 0 = PLLEN bit is enabled. The value of the PLLEN bit will affect the operation of the PLL Controller. • 1 = PLLEN bit is disabled. The value of the PLLEN bit has no effect on the operation of the PLL Controller.
4	RESERVED	R	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
3	PLLRST	R/W	1	PLL reset bit. • 0 = PLL reset is released. • 1 = PLL reset is asserted.
2	RESERVED	R	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
1	PLLPWRDN	R/W	0	PLL power-down mode select bit. • 0 = PLL is operational. • 1 = PLL is placed in a power-down state, that is, all analog circuitry in the PLL is turned off.
0	PLLEN	R/W	0	PLL enable bit. • 0 = Bypass mode. • 1 = PLL mode.

Table 5-511. Register Call Summary for PLLCTL

Power Management Overview
PLLs • RSISO Register (Offset = F0h) [reset = 0h]: [0] • DCHANGE Register (Offset = 144h) [reset = 0h]: [0] • MAIN PLL Initialization: [0][1][2][3][4][5][6] • PLLCTL Register (Offset = 100h) [reset = 0h]: [0] • PLL Control Register and Secondary Control Register: [0][1][2][3] • PLL Controller Registers: [0] • MAIN PLL Wake Up: [0] • MAIN PLL Power Down: [0]

5.4.5.4.2 SECCTL Register (Offset = 108h) [reset = 0h]

The PLL Secondary Control Register is shown in Figure 5-170 and described in Table 5-513.

Table 5-512. SECCTL Instances

Instance	Physical Address
PLL	0231 0108h

Figure 5-170. SECCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0							
23	22	21	20	19	18	17	16
BYPASS	OUTPUT_DIVIDE				RESERVED		
RW-1	RW-0				RW-10000h		
15	14	13	12	11	10	9	8
RESERVED							
RW-10000h							
7	6	5	4	3	2	1	0
RESERVED							
RW-10000h							

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 5-513. SECCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0	Reserved.
23	BYPASS	RW	1	PLL Bypass Enable. • 0 = PLL Bypass disabled. • 1 = PLL Bypass enabled.
22-19	OUTPUT_DIVIDE	RW	0	Output Divider ratio bits. • 0h = ÷1. Divide frequency by 1. • 1h = ÷2. Divide frequency by 2. • 2h = ÷3. Divide frequency by 3. • 3h = ÷4. Divide frequency by 4. • 4h-Fh = ÷5 to ÷16. Divide frequency by 5 to divide frequency by 16.
18-0	RESERVED	RW	10000h	Reserved.

Table 5-514. Register Call Summary for SECCTL

PLLs
- MAIN PLL and PLL Controller Overview: [0]
- PLL Controller Registers: [0]
- Clocks Generated by PLL Controller: [0]
- MAIN PLL Controller Operating Modes: [0]
- PLL Controller Overview: [0]
- PLL Control Register and Secondary Control Register: [0][1]
- MAIN PLL Power Down: [0]
- MAIN PLL Wake Up: [0]
- MAIN PLL Initialization: [0][1][2][3]

5.4.5.4.3 PLLM Register (Offset = 110h) [reset = 0h]

The PLL multiplier control register (PLLM) is shown in Figure 5-171 and described in Table 5-516. The MSB bits PLLM[12-6] come from the chip-level register (BOOTCFG_MAIN_PLL_CTL0).

NOTE: Table 5-516 lists all the possible values for the PLL multiplier bits (PLLM). However, some of these values may not be valid for production. For a list of valid values for PLLM, see the device Data Manual.

Table 5-515. PLLM Instances

Instance	Physical Address
PLL	0231 0110h

Figure 5-171. PLLM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										PLLM					
R-0																										R/W-1					

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 5-516. PLLM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
5-0	PLLM	R/W	1h	PLL multiplier bits. Defines the frequency multiplier of the input reference clock. • 0h = ×1 multiplier rate. • 1h = ×2 multiplier rate. • 2h = ×3 multiplier rate. • 3h = ×4 multiplier rate. • 4h-3Fh = ×5 multiplier rate to ×64 multiplier rate.

Table 5-517. Register Call Summary for PLLM

PLLs
- MAIN PLL and PLL Controller Overview: [0]
- Clocks Generated by PLL Controller: [0]
- MAIN PLL Controller Operating Modes: [0]
- PLL Controller Overview: [0]
- PLLM Register (Offset = 110h) [reset = 0h]: [0][1]
- PLL Controller Registers: [0]
- Secondary PLLs Initialization: [0][1][2]
- Multiplier: [0][1][2]
- MAIN PLL Initialization: [0]

5.4.5.4.4 PLLDIV1 Register (Offset = 118h) [reset = 0h]

The PLL Controller divider register (PLLDIV1-PLLDIV4) is shown in Figure 5-172 and described in Table 5-519. The PLLDIVn dividers generate a 50% duty cycle output clock SYSCLK1 when enabled. The PLLDIVn registers contain the default divider values on power up. Software needs to re-program a new value only if desires to change the default values. Otherwise, this register can be left unchanged.

Table 5-518. PLLDIV1 Instances

Instance	Physical Address
PLL	0231 0118h

Figure 5-172. PLLDIV1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0							
23	22	21	20	19	18	17	16
RESERVED							
R-0							
15	14	13	12	11	10	9	8
EN	RESERVED						
R/W-1	R-0						
7	6	5	4	3	2	1	0
RATIO							
R/W-0							

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 5-519. PLLDIV1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
15	EN	R/W	1	Divider D *n* enable bit. • 0 = Divider *n* is disabled. • 1 = Divider *n* is enabled.
14-8	RESERVED	R	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect
7-0	RATIO	R/W	0	Divider ratio bits. • 0h = ÷1. Divide frequency by 1 • 1h = ÷2. Divide frequency by 2 • 2h = ÷3. Divide frequency by 3 • 3h = ÷4. Divide frequency by 4 • 4h-FFh = ÷5 to ÷256 Divide frequency by 5 to divide frequency by 256.

Table 5-520. Register Call Summary for PLLDIV1

PLLs
- PLLDIV1-PLLDIV4 Register (Offset = 118h) [reset = 0h]: [0]

5.4.5.4.5 PLLDIV2 Register (Offset = 11Ch) [reset = 0h]

The PLL Controller divider register (PLLDIV2) is shown in Figure 5-173 and described in Table 5-522. The PLLDIV dividers generate a 50% duty cycle output clock SYSCLK2 when enabled. The PLLDIVn registers contain the default divider values on power up. Software needs to re-program a new value only if desires to change the default values. Otherwise, this register can be left unchanged.

Table 5-521. PLLDIV2 Instances

Instance	Physical Address
PLL	0231 011Ch

Figure 5-173. PLLDIV2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0							
23	22	21	20	19	18	17	16
RESERVED							
R-0							
15	14	13	12	11	10	9	8
EN	RESERVED						
R/W-1	R-0						
7	6	5	4	3	2	1	0
RATIO							
R/W-0							

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 5-522. PLLDIV2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
15	EN	R/W	1	Divider D *n* enable bit. • 0 = Divider *n* is disabled. • 1 = Divider *n* is enabled.
14-8	RESERVED	R	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect
7-0	RATIO	R/W	0	Divider ratio bits. • 0h = ÷1. Divide frequency by 1 • 1h = ÷2. Divide frequency by 2 • 2h = ÷3. Divide frequency by 3 • 3h = ÷4. Divide frequency by 4 • 4h-FFh = ÷5 to ÷256 Divide frequency by 5 to divide frequency by 256.

Table 5-523. Register Call Summary for PLLDIV2

PLLs
- PLLDIV1-PLLDIV4 Register (Offset = 118h) [reset = 0h]: [0]

5.4.5.4.6 PLLDIV3 Register (Offset = 120h) [reset = 0h]

The PLL Controller divider registers (PLLDIV3) are shown in Figure 5-174 and described in Table 5-525. The PLLDIVn dividers generate a 50% duty cycle output clock SYSCLK3 when enabled. The PLLDIV registers contain the default divider values on power up. Software needs to re-program a new value only if desires to change the default values. Otherwise, this register can be left unchanged.

Table 5-524. PLLDIV3 Instances

Instance	Physical Address
PLL	0231 0120h

Figure 5-174. PLLDIV3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0							
23	22	21	20	19	18	17	16
RESERVED							
R-0							
15	14	13	12	11	10	9	8
EN	RESERVED						
R/W-1	R-0						
7	6	5	4	3	2	1	0
RATIO							
R/W-2							

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 5-525. PLLDIV3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
15	EN	R/W	1	Divider 3 enable bit. • 0 = Divider 3 is disabled. • 1 = Divider 3 is enabled.
14-8	RESERVED	R	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect
7-0	RATIO	R/W	2	Divider ratio bits. • 0h = ÷1. Divide frequency by 1 • 1h = ÷2. Divide frequency by 2 • 2h = ÷3. Divide frequency by 3 • 3h = ÷4. Divide frequency by 4 • 4h-FFh = ÷5 to ÷256 Divide frequency by 5 to divide frequency by 256.

Table 5-526. Register Call Summary for PLLDIV3

PLLs
- PLLDIV1-PLLDIV4 Register (Offset = 118h) [reset = 0h]: [0]

5.4.5.4.7 PLLDIV4 Register (Offset = 160h) [reset = 0h]

The PLL Controller divider register (PLLDIV4) is shown in Figure 5-175 and described in Table 5-528. The PLLDIVn dividers generate a 50% duty cycle output clock SYSCLK4 when enabled. The PLLDIV registers contain the default divider values on power up. Software needs to re-program a new value only if desires to change the default values. Otherwise, this register can be left unchanged.

Table 5-527. PLLDIV4 Instances

Instance	Physical Address
PLL	0231 0160h

Figure 5-175. PLLDIV4 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0							
23	22	21	20	19	18	17	16
RESERVED							
R-0							
15	14	13	12	11	10	9	8
EN	RESERVED						
R/W-1	R-0						
7	6	5	4	3	2	1	0
RATIO							
R/W-4							

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 5-528. PLLDIV4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
15	EN	R/W	1	Divider 4 enable bit. • 0 = Divider 4 s disabled. • 1 = Divider 4 is enabled.
14-8	RESERVED	R	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect
7-0	RATIO	R/W	4	Divider ratio bits. • 0h = ÷1. Divide frequency by 1 • 1h = ÷2. Divide frequency by 2 • 2h = ÷3. Divide frequency by 3 • 3h = ÷4. Divide frequency by 4 • 4h-FFh = ÷5 to ÷256 Divide frequency by 5 to divide frequency by 256.

Table 5-529. Register Call Summary for PLLDIV4

PLLs
- PLLDIV1-PLLDIV4 Register (Offset = 118h) [reset = 0h]: [0]

5.4.5.4.8 PLLCMD Register (Offset = 138h) [reset = 0h]

The PLL Controller command register (PLLCMD) contains the command bit for the GO operation. PLLCMD is shown in Figure 5-176 and described in Table 5-531.

Table 5-530. PLLCMD Instances

Instance	Physical Address
PLL	0231 0138h

Figure 5-176. PLLCMD Register

31	30	29	28	27	26	25	24
RESERVED							
R-0							
23	22	21	20	19	18	17	16
RESERVED							
R-0							
15	14	13	12	11	10	9	8
RESERVED							
R-0							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	GOSET
R-0						R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 5-531. PLLCMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
1	RESERVED	R/W	0	Reserved.
0	GOSET	R/W	0	GO operation command for SYSCLK rate change and phase alignment. Before setting this bit to initiate a GO operation, check the GOSTAT bit in the PLLSTAT register to ensure all previous GO operations have completed. • 0 = No effect. Write of 0 clears bit. • 1 = Initiates GO operation. Write of 1 initiates GO operation. Once set, GOSET remains set but further writes of 1 can reinitiate the GO operation.

Table 5-532. Register Call Summary for PLLCMD

PLLs
- Divider n and GO Operation: [0][1]
- PLL Controller Registers: [0]
- ALNCTL Register (Offset = 140h) [reset = 0h]: [0][1][2][3][4][5][6][7]
- MAIN PLL Initialization: [0]
- PLLCMD Register (Offset = 138h) [reset = 0h]: [0][1]

5.4.5.4.9 PLLSTAT Register (Offset = 13Ch) [reset = 0h]

The PLL Controller status register (PLLSTAT) shows the PLL Controller status. PLLSTAT is shown in Figure 5-177 and described in Table 5-534.

Table 5-533. PLLSTAT Instances

Instance	Physical Address
PLL	0231 013Ch

Figure 5-177. PLLSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0							
23	22	21	20	19	18	17	16
RESERVED							
R-0							
15	14	13	12	11	10	9	8
RESERVED							
R-0							
7	6	5	4	3	2	1	0
RESERVED							GOSTAT
R-0							R-0

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 5-534. PLLSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
0	GOSTAT	R	0	GO operation status. • 0 = GO operation is not in progress. SYSCLK divide ratios are not being changed. • 1 = GO operation is in progress. SYSCLK divide ratios are being changed.

Table 5-535. Register Call Summary for PLLSTAT

PLLs
- Divider n and GO Operation: [0][1][2]
- PLL Controller Registers: [0]
- PLLSTAT Register (Offset = 13Ch) [reset = 0h]: [0][1]
- MAIN PLL Initialization: [0][1]
- PLLCMD Register (Offset = 138h) [reset = 0h]: [0]

5.4.5.4.10 ALNCTL Register (Offset = 140h) [reset = 0h]

The PLL Controller clock align control register (ALNCTL) is shown in Figure 5-178 and described in Table 5-537.

Table 5-536. ALNCTL Instances

Instance	Physical Address
PLL	0231 0140h

Figure 5-178. ALNCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0							
23	22	21	20	19	18	17	16
RESERVED							
R-0							
15	14	13	12	11	10	9	8
RESERVED							
R-0							
7	6	5	4	3	2	1	0
RESERVED				ALN4	ALN3	ALN2	ALN1
R-0				R/W-1	R/W-1	R/W-1	R/W-1

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 5-537. ALNCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0	Reserved. This reserved bit location is always read as 0. A value written to this field has no effect.
3	ALN4	R/W	1	SYSCLK4 alignment. • 0 = Do not align SYSCLK4 to other SYSCLKs during GO operation. If SYS 4 in DCHANGE is set, SYSCLK4 switches to the new ratio immediately after the GOSET bit in PLLCMD is set. • 1 = Align SYSCLK4 to other SYSCLKs selected in ALNCTL when the GOSET bit in PLLCMD is set and SYS 4 in DCHANGE is 1. The SYSCLK4 rate is set to the ratio programmed in the RATIO bit in PLLDIV 4.
2	ALN3	R/W	1	SYSCLK3 alignment. • 0 = Do not align SYSCLK3 to other SYSCLKs during GO operation. If SYS 3 in DCHANGE is set, SYSCLK3 switches to the new ratio immediately after the GOSET bit in PLLCMD is set. • 1 = Align SYSCLK3 to other SYSCLKs selected in ALNCTL when the GOSET bit in PLLCMD is set and SYS 3 in DCHANGE is 1. The SYSCLK3 rate is set to the ratio programmed in the RATIO bit in PLLDIV 3.
1	ALN2	R/W	1	SYSCLK2 alignment. • 0 = Do not align SYSCLK2 to other SYSCLKs during GO operation. If SYS 2 in DCHANGE is set, SYSCLK2 switches to the new ratio immediately after the GOSET bit in PLLCMD is set. • 1 = Align SYSCLK2 to other SYSCLKs selected in ALNCTL when the GOSET bit in PLLCMD is set and SYS 2 in DCHANGE is 1. The SYSCLK2 rate is set to the ratio programmed in the RATIO bit in PLLDIV 2.

Table 5-537. ALNCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	ALN1	R/W	1	SYSCLK1 alignment. • 0 = Do not align SYSCLK1 to other SYSCLKs during GO operation. If SYS 1 in DCHANGE is set, SYSCLK1 switches to the new ratio immediately after the GOSET bit in PLLCMD is set. • 1 = Align SYSCLK1 to other SYSCLKs selected in ALNCTL when the GOSET bit in PLLCMD is set and SYS 1 in DCHANGE is 1. The SYSCLK16 rate is set to the ratio programmed in the RATIO bit in PLLDIV 1.

Table 5-538. Register Call Summary for ALNCTL

PLLs
- Divider n and GO Operation: [0][1][2][3][4]
- PLL Controller Registers: [0]
- MAIN PLL Initialization: [0]
- DCHANGE Register (Offset = 144h) [reset = 0h]: [0]
- ALNCTL Register (Offset = 140h) [reset = 0h]: [0][1][2][3][4]

5.4.5.4.11 DCHANGE Register (Offset = 144h) [reset = 0h]

When a different ratio is written to the PLLDIVn registers, the PLLCTL flags the change in the DCHANGE status register. During the GO operation, the PLL Controller changes only the divide ratio of the SYSCLKs with the bit set in DCHANGE. Note that the ALNCTL register determines if that clock also needs to be aligned to other clocks. The PLLDIV divider ratio change status register is shown in Figure 5-179 and described in Table 5-540.

Table 5-539. DCHANGE Instances

Instance	Physical Address
PLL	0231 0144h

Figure 5-179. DCHANGE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0							
23	22	21	20	19	18	17	16
RESERVED							
R-0							
15	14	13	12	11	10	9	8
RESERVED							
R-0							
7	6	5	4	3	2	1	0
RESERVED				SYS4	SYS3	SYS2	SYS1
R-0				R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 5-540. DCHANGE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0	Reserved. This reserved bit location is always read as 0. A value written to this field has no effect.
3	SYS4	R/W	0	Identifies when the SYSCLK4 divide ratio has been modified. • 0 = SYSCLK4 ratio has not been modified. When GOSET is set, SYSCLK4 will not be affected. • 1 = SYSCLK4 ratio has been modified. When GOSET is set, SYSCLK4 will change to the new ratio.
2	SYS3	R/W	0	Identifies when the SYSCLK3 divide ratio has been modified. • 0 = SYSCLK3 ratio has not been modified. When GOSET is set, SYSCLK3 will not be affected. • 1 = SYSCLK3 ratio has been modified. When GOSET is set, SYSCLK3 will change to the new ratio.
1	SYS2	R/W	0	Identifies when the SYSCLK2 divide ratio has been modified. • 0 = SYSCLK2 ratio has not been modified. When GOSET is set, SYSCLK2 will not be affected. • 1 = SYSCLK2 ratio has been modified. When GOSET is set, SYSCLK2 will change to the new ratio.
0	SYS1	R/W	0	Identifies when the SYSCLK1 divide ratio has been modified. • 0 = SYSCLK1 ratio has not been modified. When GOSET is set, SYSCLK1will not be affected. • 1 = SYSCLK 1 ratio has been modified. When GOSET is set, SYSCLK1 will change to the new ratio.

Table 5-541. Register Call Summary for DCHANGE

PLLs
• Divider n and GO Operation: [0][1][2] • PLL Controller Registers: [0] • MAIN PLL Initialization: [0] • DCHANGE Register (Offset = 144h) [reset = 0h]: [0][1] • ALNCTL Register (Offset = 140h) [reset = 0h]: [0][1][2][3][4][5][6][7]

5.4.5.4.12 SYSTAT Register (Offset = 150h) [reset = 0h]

The SYSCLK status register (SYSTAT) shows the status of SYSCLKn. SYSTAT is shown in Figure 5-180 and described in Table 5-543.

Table 5-542. SYSTAT Instances

Instance	Physical Address
PLL	0231 0150h

Figure 5-180. SYSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0							
23	22	21	20	19	18	17	16
RESERVED							
R-0							
15	14	13	12	11	10	9	8
RESERVED							
R-0							
7	6	5	4	3	2	1	0
RESERVED				SYS4ON	SYS3ON	SYS2ON	SYS1ON
R-0				R/W-	R/W-	R/W-	R/W-

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 5-543. SYSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
3	SYS4ON	R	-	SYSCLK4 on status. • 0 = SYSCLK4 is gated. • 1 = SYSCLK4 is on.
2	SYS3ON	R	-	SYSCLK3 on status. • 0 = SYSCLK3 is gated. • 1 = SYSCLK3 is on.
1	SYS2ON	R	-	SYSCLK2 on status. • 0 = SYSCLK2 is gated. • 1 = SYSCLK2 is on.
0	SYS1ON	R	-	SYSCLK1 on status. • 0 = SYSCLK1 is gated. • 1 = SYSCLK1 is on.

Table 5-544. Register Call Summary for SYSTAT

PLLs
- SYSTAT Register (Offset = 150h) [reset = 0h]: [0][1]
- PLL Controller Registers: [0]

5.4.5.4.13 RSTYPE Register (Offset = E4h) [reset = 0h]

The reset type status (RSTYPE) register latches the cause of the last reset. If multiple reset sources occur simultaneously, this register latches the highest priority reset source. The Reset Type Status register is shown in Figure 5-181 and described in Table 5-546.

Table 5-545. RSTYPE Instances

Instance	Physical Address
PLL	0231 00E4h

Figure 5-181. RSTYPE Register

31	30	29	28	27	26	25	24
RESERVED			EMU-RST	RESERVED			
R-0h			R-	R-0h			
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						WDRST	RESERVED
R-0h						R-	R-0h
7	6	5	4	3	2	1	0
RESERVED					PLLCTRLRST	RESET	POR
R-0h					R-	R-	R-

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 5-546. RSTYPE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved. Read only. Always reads as 0. Writes have no effect.
28	EMU-RST	R	-	Reset initiated by emulation. • 0 = Not the last reset to occur. • 1 = The last reset to occur.
27-10	RESERVED	R	0h	Reserved. Read only. Always reads as 0. Writes have no effect.
9	WDRST	R	-	Reset initiated by Watchdog Timer(s). • 0 = Not the last reset to occur. • 1 = The last reset to occur.
8-3	RESERVED	R	0h	Reserved. Read only. Always reads as 0. Writes have no effect.
2	PLLCTRLRST	R	-	Reset initiated by PLL Controller (software reset). • 0 = Not the last reset to occur. • 1 = The last reset to occur.
1	RESET	R	-	RESET pin reset. • 0 = Not the last reset to occur. • 1 = The last reset to occur.
0	POR	R	-	Power-on reset. • 0 = Not the last reset to occur. • 1 = The last reset to occur.

Table 5-547. Register Call Summary for RSTYPE

PLLs
- PLL Controller Registers: [0]
- RSTYPE Register (Offset = E4h) [reset = 0h]: [0]

5.4.5.4.14 RSCTRL Register (Offset = E8h) [reset = 3h]

This register contains a key that enables writes to the MSB of this register and the RSCFG register. The key value is 0x5A69. A valid key will be stored as 0x000C, any other key value is invalid. When the RSCTRL or the RSCFG is written, the key is invalidated. Every write must be set up with a valid key. The Software Reset Control register (RSCTRL) is shown in Figure 5-182 and described in Table 5-549.

Table 5-548. RSCTRL Instances

Instance	Physical Address
PLL	0231 00E8h

Figure 5-182. RSCTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							SWRST
R-0h							R/W-0h(1)
15	14	13	12	11	10	9	8
KEY							
R/W-3h							
7	6	5	4	3	2	1	0
KEY							
R/W-3h							

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

(1) Writes are conditionalyl based on a valid key.

Table 5-549. RSCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved.
16	SWRST	R/W	1h	Software reset. • 0 = Reset. • 1 = Not reset.
15-0	KEY	R/W	3h	Key used to enable writes to RSCTRL and RSCFG.

Table 5-550. Register Call Summary for RSCTRL

PLLs
- RSISO Register (Offset = F0h) [reset = 0h]: [0]
- RSCTRL Register (Offset = E8h) [reset = 3h]: [0][1][2]
- PLL Controller Registers: [0]
- RSCFG Register (Offset = ECh) [reset = 0h]: [0]

5.4.5.4.15 RSCFG Register (Offset = ECh) [reset = 0h]

This register is used to configure the type of reset initiated by RESET pin, the watchdog timers

, and the PLL Controller's RSCTRL register: a hard reset or a soft reset. By default, these resets are hard resets. The Reset Configuration Register (RSCFG) is shown in Figure 5-183 and described in Table 5-552.

Table 5-551. RSCFG Instances

Instance	Physical Address
PLL	0231 00ECh

Figure 5-183. RSCFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						WDBLOCK	RESERVED
R-0h						R/W-0[1]	R-0h
15	14	13	12	11	10	9	8
RESERVED		PLLCTLRSTTYPE	RESETTYPE	RESERVED			
R-0h		R/W-0h[1]	R/W-0h[1]	R-0h			
7	6	5	4	3	2	1	0
RESERVED						WDTYPE	RESERVED
R-0h						R/W-0[1]	R-0

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

(1) Writes are conditionally based on a valid key. For details, see Section 5.4.5.4.14

Table 5-552. RSCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved.
17	WDBLOCK	RW	0h	Block Watchdog timers reset: • 0 = Not blocked (default) • 1 = Blocked
16-14	RESERVED	R	0h	Reserved.
13	PLLCTLRSTTYPE	R/W	0h	PLL Controller initiates a software driven reset of type: • 0 = Hard reset (default) • 1 = Soft reset
12	RESETTYPE	R/W	0h	RESET pin initiates a reset of type: • 0 = Hard reset (default) • 1 = Soft reset
11-2	RESERVED	RW	0h	Reserved.
1	WDTYPE	R/W	0h	Watchdog Timers initiate a reset of type: • 0 = Hard reset (default) • 1 = Soft reset
0	RESERVED	RW	0h	Reserved.

Table 5-553. Register Call Summary for RSCFG

PLLs • RSCTRL Register (Offset = E8h) [reset = 3h]: [0][1][2] • PLL Controller Registers: [0] • RSCFG Register (Offset = ECh) [reset = 0h]: [0]

5.4.5.4.16 RSISO Register (Offset = F0h) [reset = 0h]

This register is used to select the module clocks that must maintain their clocking without pausing through non power-on reset. Setting any of these bits effectively blocks reset to all PLLCTL registers in order to maintain current values of PLL multiplier ratios, divide ratios, and other settings. The Reset Isolation register (RSCTRL) is shown in Figure 5-184 and described in Table 5-555.

NOTE: For a list of modules that can be reset-isolated, see Section 5.3.9, *Reset Isolation* and Section 5.2.2.4, *Clock Domains*.

Table 5-554. RSISO Instances

Instance	Physical Address
PLL	0231 00F0h

Figure 5-184. RSISO Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				MOD_ISO[N]																											
R-0h				R/W-0h																											

LEGEND: R/W = Read/Write; R = Read only; *-n* = value after reset

Table 5-555. RSISO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved.
27-0	MOD_ISO[N]	R/W	0h	Isolate Module domain[N]. • 0 = Not reset isolated. • 1 = Reset Isolated.

Table 5-556. Register Call Summary for RSISO

PLLs • PLL Controller Registers: [0]

5.4.6 Module Clock Distribution

The clock signals from the MAIN PLL Controller are routed to various modules and peripherals on the device. Some modules and peripherals have one or more internal clock dividers. Other modules and peripherals have no internal clock dividers, but are grouped together and receive clock signals from a shared local clock divider. Internal and shared local clock dividers have fixed division ratios.

Some modules need clocks from dedicated PLLs as described below:

- ARM PLL: ARMSS
- DSS PLL: Display subsystem (DSS) pixel clock
- UART PLL: UARTs (including ICSS UARTs), MMC, USB PHY, QSPI
- ICSS PLL: ICSS PRU cores
- NSS/IEP PLL: NSS clocks, USB PHY, ICSS PRU cores, ICSS IEP
- DDR PLL: DDR3 EMIF and PHY.

Table 5-557 describes the clock distribution to device modules and subsystems.

Table 5-557. Module Clock Distribution

Module	Module Clock Port	Description	Source	Comments
PMMC	MPM_VBUS_CLK	Bus interface clock	CHIP_CLK1/6	
	MPM_FUNC_32K_CLK	32 kHz free running Clock	CHIP_CLK1/6	
	MPM_FUNC_OSC_CLK	Main Oscillator Clock	CHIP_CLK1/6	
	MPM_DAP_CLK	Debug Clock	CHIP_CLK1/6	
MLB	MLB_SYS _CLK	Min Clk = 2048Fs (Max 220 MHz)	CHIP_CLK1/3	Fs is Frame Sync Rate. This is same as Audio Sampling Rate.
	MLB_SHB_OCP_CLK	Must be less than or equal to SYS_CLK	CHIP_CLK1/3	
	MLB_SPB_OCP_CLK	Must be less than or equal to SYS_CLK	CHIP_CLK1/3	
	MLB_IO_CLK	Single IO Clock (3-pin config)	IO	Max Frequency is 49.152 MHz (1024Fs)
	MLBP_IO_CLK	Differential IO Clock (6-pin config)	IO	Max Frequency is 98.304 MHz (2048Fs)
DSS	PI_DSS_OCP_CLK	250 MHz interface clock	CHIP_CLK1/4	When CHIP_CLK1 frequency is set to 400MHz, CHIP_CLK1/4 is 100MHz. In this case only 720p display resolution can be supported.
	PI_DSS_VP_CLK	Output Pixel clock is derived from this clock. This clock needs to support various frequencies to satisfy different display resolutions. Hence a dedicated PLL is instantiated.	DSS PLL	Needs at least 150 MHz to support a resolution of 1920×1080×60fps (1080p). K2G1x supports a 4-bit divider for DSS_VP_CLK with register control in BOOT_CFG: BOOTCFG_SPARE1
	DSS_PCLK	Output on Device Pin	IO	
MCBSP	VBUS_CLK	Also known as CPU clock. This clock must be 2× the rate of IO clock. Since IO clock target is 50 MHz, vbus_clk must be at least 100 MHz.	CHIP_CLK1/3	
	CLKX	Transmit Clock (IO)	IO	
	CLKR	Receive Clock (IO)	IO	
	CLKS	External Clock Input (I)	AUDIO_OSCIN or MLB IO Clocks or SYS_OSCCLK or XREFCLK IO or UART PLL/4	Selection is made in BOOTCFG_SERIALPORT_CLKCTL register

Table 5-557. Module Clock Distribution (continued)

Module	Module Clock Port	Description	Source	Comments
MCASP	VBUS_CLK	VBUS and Functional Clock. This clock must 2× the rate of IO clock. Since the max target for IO clock rate is 75MHz. This clock must be at least 150MHz.	CHIP_CLK1/3	
	AUX_CLK	Auxillary clock input.	AUDIO_OSCIN or MLB IO Clocks or SYS_OSCCLK or XREFCLK IO or UART PLL/4	Selection is made in BOOTCFG_SERIALPORT_CLKCTL register
	ACLKX	Transmit Audio Data clock	IO	
	ACLKR	Receive Audio Data clock	IO	
	AHCLKX	Transmit Audio High Frequency Clock	IO	
	AHCLKR	Receive Audio High Frequency Clock	IO	
ASRC	SYS_CLK	System Clock (Fixed 200-MHz clock)	NSS/IEP PLL/5	
	VBUSP_CLK	VBUS clock	CHIP_CLK1/3	
	AASRC_RXSYNC0	Receive Frame Sync Channel 0	MCASP0AFSR	
	AASRC_RXSYNC1	Receive Frame Sync Channel 1	MCASP1AFSR	
	AASRC_RXSYNC1	Receive Frame Sync Channel 2	MCASP2AFSR	
	AASRC_RXSYNC3	Receive Frame Sync Channel 3	MLB_IO_CLK (3-Pin Mode)	
	AASRC_RXSYNC4	Receive Frame Sync Channel 4	MLBP_IO_CLK (6-Pin Mode)	
	AASRC_RXSYNC5	Receive Frame Sync Channel 5	MCASP0AHCLKR	
	AASRC_RXSYNC6	Receive Frame Sync Channel 6	MCASP1AHCLKR	
	AASRC_RXSYNC7	Receive Frame Sync Channel 7	MCASP2AHCLKR	
	AASRC_TXSYNC0	Transmit Frame Sync Channel 0	MCASP0AFSX	
	AASRC_TXSYNC1	Transmit Frame Sync Channel 1	MCASP1AFSX	
	AASRC_TXSYNC2	Transmit Frame Sync Channel 2	MCASP2AFSX	
	AASRC_TXSYNC3	Transmit Frame Sync Channel 3	MLB_IO_CLK (3-Pin Mode)	
	AASRC_TXSYNC4	Transmit Frame Sync Channel 4	MLBP_IO_CLK (6-Pin Mode)	
	AASRC_TXSYNC5	Transmit Frame Sync Channel 5	MCASP0AHCLKX	
	AASRC_TXSYNC6	Transmit Frame Sync Channel 6	MCASP1AHCLKX	
	AASRC_TXSYNC7	Transmit Frame Sync Channel 7	MCASP2AHCLKX	
DCAN	VBUS_CLK	VBUS clock	CHIP_CLK1/6	
	CAN_CLK	CAN functional clock used for data transmit and receive.	SYS_OSCCLK	This clock is sourced directly from System Oscillator output avoiding PLLs to achieve the lowest possible jitter.
EMIF_DDR	V_CLK	VBUSM Clock (EMIF)	CHIP_CLK1/2	
	M_CLK	Memory Clock (EMIF) = 1/2 of IO Clock ddr_clkn/p	DDR PLL CLK	
	PUB_CTL_CLK	PHY Utility Block Control Clock	DDR PLL CLK	
	PHY_CTL_CLK	PHY Control Clock	DDR PLL CLK	
	VBUSP_CLK	EMIF Config Clock (VBUSP). Must be less than or equal to pub_ctl_clk	CHIP_CLK1/6	
	DDR_CLKN/P	Complementary DDR CLK OUTPUT (PIN)	IO	
MMC	VBUS_CLK	VBUSP Master Clock	CHIP_CLK1/3	
	CLK_ADPI	Functional Clock (96 MHz)	UART PLL/4	
	CLK_ADPO/CLKFDBKI	Functional Clock Looped Back through IO Buffer	CLK_ADPI	

Table 5-557. Module Clock Distribution (continued)

Module	Module Clock Port	Description	Source	Comments
	CLK32K	32-kHz Debounce Clock	UART PLL/12000	
GPMC	GPMC_FCLK	Functional clock and VBUS clock.	CHIP_CLK1/3	
	GPMC_CLK	IO CLOCK. Max rate 133 MHz	IO	
ELM	CLK	Functional clock and VBUS clock.	CHIP_CLK1/3	
ADTF	CLK	Functional clock and VBUS clock.	CHIP_CLK1/6	
SPI	VBUSP_CLK	Functional and vbus clock	CHIP_CLK1/6	
	SPICLK	IO	IO	
PRU-ICSS_0, PRU-ICSS_1	VCLK_CLK	VBUS Clock	CHIP_CLK1/3	
	CORE0_CLK	ICSS_0 Clock	ICSS PLL or NSS/IEP PLL/5	ICSS core clock is selected based on protocols. ICSS_0 and ICSS_1 core clocks can be set independently. See BOOTCFG_ICSS_CLKCTL register.
	CORE1_CLK	ICSS_1 Clock	ICSS PLL or NSS/IEP PLL/5	
	UCLK_CLK	UART Clock (192 MHz)	UART PLL/2	UART clock rate of 192 MHz is needed to support 12 Mbps Baud rate for Profibus protocol
	IEPCLK_CLK	Ethernet Clock	NSS/IEP PLL/5	IEP Clock needs to be at a constant clock rate of 200 MHz
USB	BUS_CLK	VBUS Clock	CHIP_CLK1/6	
	PHYMMR_CLK	PHY Configuration Clock	CHIP_CLK1/24	
	SUSP_CLK	Suspend Clock	CHIP_CLK1/24	
	REF_CLK	PHY Ref clock	CHIP_CLK1/24	
	CLKCORE	Core clock	Divided version of NSS/IEP PLL or UART PLL output.	12/19.2/24/50 MHz. The clock source and speed is set in registers BOOTCFG_USB0_CLKCTL and BOOTCFG_USB1_CLKCTL
	XO	External Reference Clock Input	IO	XO clock is used to provide external reference clock input to USB. When external reference clock is not needed, internal Core clock canl be used.
Networking Subsystem (NSS)	VCLK	VBUS Clock	CHIP_CLK1/3	
	SA_UL_CLK	Core Clock for SA_UL	CHIP_CLK1/3	
	SA_UL_X1_CLK	Half Rate Clock for SA_UL	CHIP_CLK1/6	
	ESW_CLK	Core Clock for Ethernet	CHIP_CLK1/3	
	CPTS_RFTCLK	CPTS Reference Clock	CPTS_REFCLK Input	See Section 11.13.4.4.7, *Common Platform Time Sync (CPTS).*
	GMII_RFTCLK	125MHz GMII Gigabit Mode Clock	NSS/IEP PLL/8	
	GMII_MRCLK	GMII Receive Clock (Device Pin)	IO	
	GMII_MTCLK	GMII 10/100 Transmit Clock (Device Pin)	IO	
	GMII_GMTCLK	GMII Clock output (Device Pin)	IO	
	RGMII_MHZ_5_CLK	5 MHz RGMII Clock	NSS/IEP PLL/200	
	RGMII_MHZ_50_CLK	50 MHz RGMII Clock	NSS/IEP PLL/20	
	RGMII_MHZ_250_CLK	250 MHz RGMII Clock	NSS/IEP PLL/4	
	RGMII_TXC_I_CLK	RGMII Transmit Clock Input (MII Mode). Device Pin	IO	
	RGMII_RXC_CLK	RGMII Receive Clock (Device Pin)	IO	

Table 5-557. Module Clock Distribution (continued)

Module	Module Clock Port	Description	Source	Comments
	RMII_MHZ_50_CLK	RMII Clock (Device Pin and Internal)	IO (CLKOUT looped back to RMIIREFCLK pin on PCB)	
	MDCLK	MDIO Clock (input or output)	IO	
PCIe	PCI_REFCLK	Differential IO. 100 MHz	IO	
	VBUS_CLK	VBUS clock for Master and Slave interfaces	CHIP_CLK1/3	
	Functional clocks	Other functional clocks	CHIP_CLK1/2, CHIP_CLK1/4, CHIP_CLK1/6	
GPIO	VBUS_CLK	VBUS and Functional clock	CHIP_CLK1/6	
TIMER	VBUS_CLK	VBUS and Functional clock	CHIP_CLK1/6	
	Timer Inputs (TINP) used as clock	This clock must be 1/4 the rate of VBUS_CLK	IO	
MESSAGE MANAGER	VBUS_CLK	VBUS and Functional clock	CHIP_CLK1/6	
BOOT_CFG	VBUS_CLK	VBUS and Functional clock	CHIP_CLK1/6	
ARM BOOTROM	VBUS_CLK	VBUS and Functional clock	CHIP_CLK1/6	
DEBUGSS	VBUSP_CTTBRCLK_CLK	VBUSP TBR Clock	CHIP_CLK1/3	
	VBUSP_STMD0_CLK	VBUSP STM Clock	CHIP_CLK1/3	
	VBUSP_SLAVE_CLK	VBUSP Config Slave Clock	CHIP_CLK1/6	
	VBUSP_MASTER_CLK	VBUSP DAP Master Clock	CHIP_CLK1/6	
	TCK	JTAG Clock Input (Device Pin)	IO	
	CS_TRCEXPT_CLK	CS Trace (TPIU) export (XPT) Clock	PLL controller SYSCLK3	Up to 333 MHz
	DSP_TRACECLK	DSP Trace export Clock	PLL controller SYSCLK3	Up to 333 MHz
	STMXPT_CLK	STM PTI export (XPT) Clock	PLL controller SYSCLK4	Up to 200 MHz
UART	CBA_CLK_PI	Functional and VBUS clock.	UART PLL/2	
ePWM	VBUS_CLK	Functional and VBUS clock.	CHIP_CLK1/6	
eQEP	VBUS_CLK	Functional and VBUS clock.	CHIP_CLK1/6	
eCAP	VBUS_CLK	Functional and VBUS clock.	CHIP_CLK1/6	
I2C	VBUS_CLK	Functional and VBUS clock.	CHIP_CLK1/6	
CP_TRACER (CPT_[0:1])	CP_TRACER_CLK	Functional and VBUS clock.	CHIP_CLK1	
CP_TRACER (CPT_[2:14])	CP_TRACER_CLK	Functional and VBUS clock.	CHIP_CLK1/3	
EDMA	TPTC_CLK	Functional and VBUS clock (Transfer Controller)	CHIP_CLK1/3	
	TPCC_CLK	Functional and VBUS clock (Channel Controller)	CHIP_CLK1/3	
SEMAPHORE	VBUS_CLK	Functional and VBUS clock.	CHIP_CLK1/3	
PSC	VBUS_CLK	Functional and VBUS clock.	CHIP_CLK1/6	
	SLOW_SYSCLK		CHIP_CLK1/24	
INTC	VBUS_CLK	Functional and VBUS clock.	CHIP_CLK1/6	
GIC	VBUS_CLK	Functional and VBUS clock.	CHIP_CLK1/3	
QSPI	QSPI_REF_CLK	QSPI Functional Reference clock	UART PLL	
	DATA_BUS_CLK	Data VBUS clock	CHIP_CLK1/6	
	CFG_BUS_CLK	Config VBUS Clock	CHIP_CLK1/6	

Table 5-557. Module Clock Distribution (continued)

Module	Module Clock Port	Description	Source	Comments
	QSPI_CLK_O	IO Clock (output)	IO	
	QSPI_RCLK_I	Return clock (looped back clock of qspi_clk_o)	IO	
GTC	CLK INPUT	Global Timebase Counter clock	SYS_OSCCLK	
	VBUSP_CLK	Bus Clock	CHIP_CLK1/6	
ARMSS	Various clocks		ARM PLL and MAIN PLL (CHIP_CLK1/1, CHIP_CLK1/2, CHIP_CLK1/3, CHIP_CLK1/6)	
DSP	Various clocks		MAIN PLL (CHIP_CLK1/1, CHIP_CLK1/2, CHIP_CLK1/3, CHIP_CLK1/4)	
MSMC	Various clocks		MAIN PLL (CHIP_CLK1/1, CHIP_CLK1/2)	
DFT_SS	Various clocks		MAIN PLL (CHIP_CLK1/2, CHIP_CLK1/3, CHIP_CLK1/4, CHIP_CLK1/6, CHIP_CLK1/8) and IO	
CBASS	Various clocks		MAIN PLL	

Processors and Accelerators

Topic **Page**

6.1 Arm Cortex-A15 Subsystem

This section describes the Arm® Cortex®-A15 Subsystem for the device.

6.1.1 Arm Subsystem Overview

6.1.1.1 Introduction

The Arm Subsystem (ARMSS) of the SoC integrates a single Cortex-A15 processor with additional logic for bus protocol conversion, local power management, and various debug and trace enhancements.

The Cortex-A15 processor is an Arm®v7A-compatible, multi-issue out-of-order superscalar execution engine with integrated L1 caches.

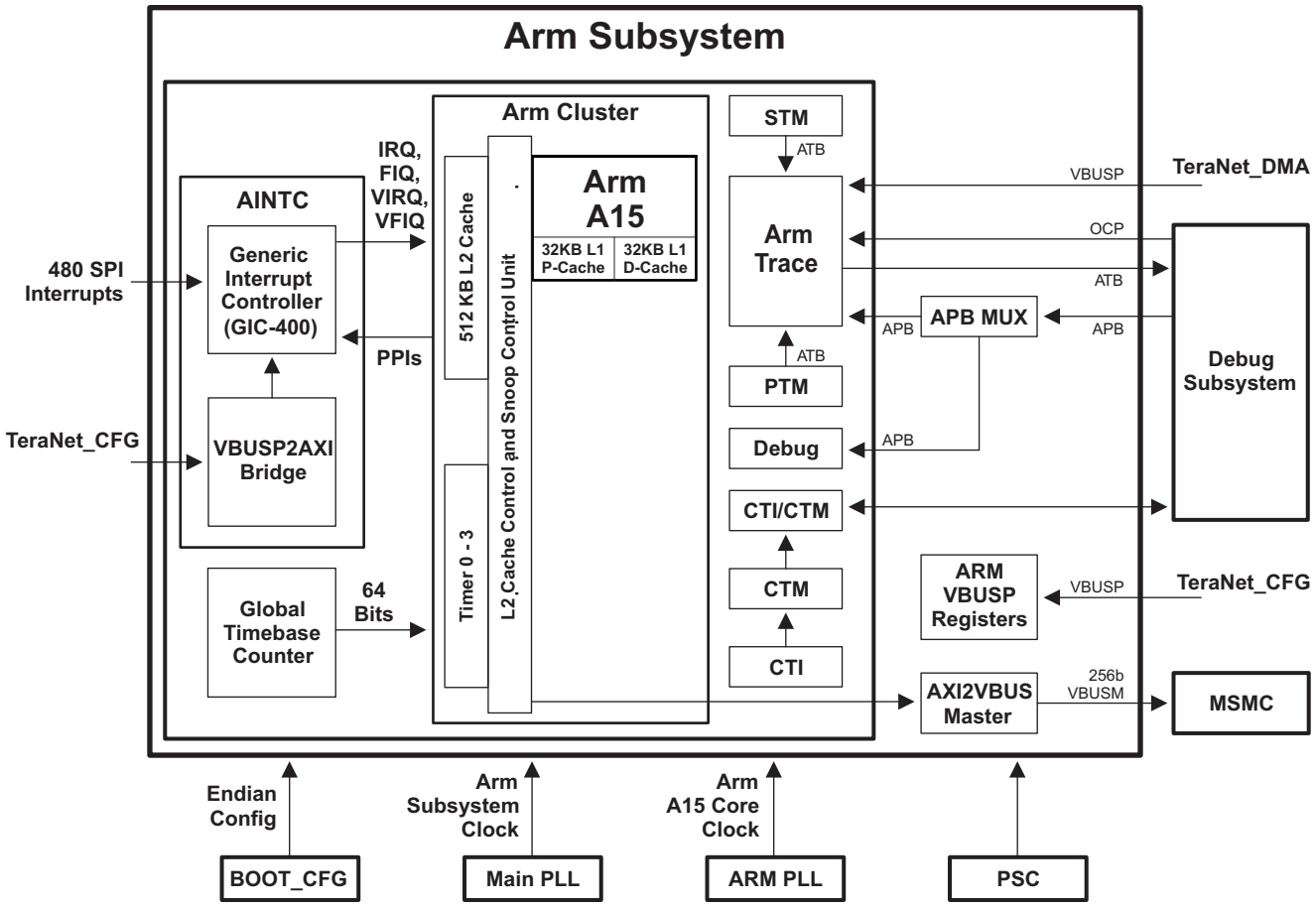
The implementation also supports advanced SIMDv2 (NEON™ technology) and VFPv4 (vector floating point) architecture extensions, security, virtualization, LPAE (large physical address extension), and multiprocessing extensions.

The Arm Subsystem includes a 512KB L2 cache and support for AMBA4 AXI and AXI coherence extension (ACE) protocols.

NOTE: The **Arm Subsystem** is also referred to as **Arm CorePac**.

Figure 6-1 shows an overview of the Arm Subsystem in the SoC.

Figure 6-1. Arm Subsystem Overview



6.1.1.2 Features

The Arm subsystem supports the following key features:

- Arm Cortex-A15 processor, full implementation of Arm®v7-A architecture instruction set
- 32KB L1 instruction (L1I) and data (L1D) caches
- 512KB L2 cache
- Super scalar, variable-length, out-of-order pipeline (12 stage in-order, 3-12 stage out-of-order)
- 128-bit instruction fetch
- 3-wide instruction decode
- 3-wide instruction dispatch
- 8-wide instruction issue
- Dynamic branch prediction with Branch Target Buffer (BTB) and Global History Buffer (GHB), a return stack, and an indirect predictor
- Integrated Neon and VFP (Vector Floating Point unit)
- Support for security and virtualization extensions
- Error Correction Code (ECC) protection for L1 data cache and L2 cache, parity protection for L1 instruction cache
- 32-entry fully-associative L1 Translation Look-aside Buffers (TLBs), for instruction fetch, data loads, and data stores
- 512-entry 4-way set-associative L2 TLB
- AMBA 4.0 AXI Coherency Extension (ACE) master port which is directly connected to MSMC (Multicore Shared Memory Controller) for low-latency access to shared MSMC SRAM
- Dedicated Arm clocking (ARM_PLL) for full flexibility in performance trade-offs
- Support for four integrated generic timers, in addition to 1 dedicated SoC-level watchdog timer (TIMER_5)
- Support for invasive (stop-mode) and non-invasive (tracing, performance monitoring) debug modes and cross triggering for multiprocessor debugging
- Support for processor instruction trace using Program Trace Macrocell (PTM) and data trace (printf style debug) using System Trace Macrocell (STM)
- Support for up to 480 interrupt requests via the Arm Interrupt Controller (AINTC) module
- Support for little-endian operation only

The Arm subsystem does not support the following features:

- ACP (Accelarator Coherancy Port) Slave
- Native AXI Master interface (only MSMC option is used)

6.1.2 Arm Subsystem Functional Description

6.1.2.1 ARMSS Block Diagram

Figure 6-1 shows a high-level block diagram of the Arm Subsystem along with its connectivity to other SoC subsystems and modules.

The Arm Subsystem integrates the following major blocks:

- Single-core Arm Cluster (Section 6.1.2.1.1)
- AXI2VBUS_MASTER (Section 6.1.2.1.2)
- Debug and Trace components (see *On-chip Debug* chapter)
- ARM_VBUSP registers
- AINTC
- Global Timebase Counter (GTC)
- Various interfaces for interaction with other SoC subsystems and modules

6.1.2.1.1 Arm A15 Cluster

The Arm A15 Cluster (Cortex-A15 MPCore) includes a single Cortex-A15 processor core and debug components, as well as 512KB L2 cache.

Table 6-1 shows the features supported by the Cortex-A15 processor core.

Table 6-1. Cortex-A15 Processor Core Supported Features

Features	Description
Arm version 7-A ISA	Standard Cortex-A15 processor instruction set + Thumb2, ThumbEE, JazelleX Java accelerator, and media extensions
	Backward compatible with previous Arm ISA versions
Cortex-A15 processor version	R2P2
Integer core	Main core for processing integer instructions
NEON core	Gives greatly enhanced throughput for media workloads and VFP-Lite support
Architecture Extensions	Security, virtualization and LPAE (40-bit physical address) extensions
L1 Lcache and Dcache	32KB, 2-way, 16-word line, 128-bit interface
L2 cache	512KB, 16-way, 16-word line, 128-bit interface to L1, ECC/Parity is supported shared between cores
	L2 valid bits cleared by software loop or by hardware
Cache Coherency	Support for coherent memory accesses between A15 core and other non-core master peripherals (Ex: EDMA) in the DDR3A and MSMC SRAM space.
Branch target address cache	Dynamic branch prediction with Branch Target Buffer (BTB) and Global History Buffer (GHB), a return stack, and an indirect predictor
Enhanced memory management unit	Mapping sizes are 4KB, 64KB, 1MB, and 16MB
Buses	128b AXI4 internal bus from Cortex-A15 converted to a 256b VBUSM to interface (through the MSMC) with MSMC SRAM, DDR EMIF, ROM, Interrupt controller and other system peripherals
Non-invasive Debug Support	Processor instruction trace using Program Trace Macrocell (Coresight™ PTM), Data trace (print-f style debug) using System Trace Macrocell (Coresight™ STM) and Performance Monitoring Unit (PMU)
Misc Debug Support	JTAG-based debug and Cross-triggering
Clocking	Dedicated ARM PLL for flexible clocking scenarios

The Arm A15 core has the following interfaces with other modules in the Arm subsystem:

- AXI (Advanced eXtensible Interface)
- APB (AMBA Advanced Peripheral Bus)
- ATB (AMBA Trace Bus)
- Cross trigger

For more information about:

- General Arm architecture, see the *Arm Architecture Reference Manual*
- Arm A15 core, see the *Arm Cortex-A15 MPCore Processor Technical Reference Manual*.

6.1.2.1.2 AXI2VBUS Master

6.1.2.1.2.1 AXI2VBUS_MASTER Overview

The AXI2VBUS_MASTER is a custom block for interfacing the A15 Cluster AXI bus to the System Interconnect through the Multicore Shared Memory Controller (MSMC). The MSMC is the global memory management unit that provides an interface between processor masters (such as DSP and ARMSS) and system memory. The AXI2VBUS_MASTER serves as a bridge between the ARMSS protocols and the MSMC, and also provides bus width and clock conversion.

The AXI2VBUS_MASTER serves the following purposes in the Arm subsystem:

- AXI-to-VBUSM protocol conversion
- Coherency extensions translation between ACE and MSMC custom protocols
- A 1:1 async boundary between the A15 Cluster and MSMC clock domains while providing 128-bit to 256-bit data width conversion, leading to no loss in overall throughput

6.1.2.1.2.2 Cache Line Size

Both MSMC and A15 core use a cache line size of 64 bytes.

6.1.2.1.3 Arm Debug Components

The Arm Subsystem integrates a number of Coresight debug components around the A15 Cluster for supporting embedded debug and trace, as well as cross-triggers. These are in addition to the PTMs inside the A15 cores and the CTM and CTI modules at the A15 core L2 level. For more details on Arm debug components, see Chapter 12, *On-chip Debug*.

6.1.2.2 ARMSS Interrupt Controller (AINTC)

The Arm interrupt controller (AINTC) is responsible for prioritizing all service requests from the system peripherals and the secondary interrupt controller (CIC) and then generating either nIRQ or nFIQ to the Cortex-A15 processor. The type of the interrupt (nIRQ or nFIQ) and the priority of the interrupt inputs are programmable.

The AINTC has the capability to handle up to 480 requests, which can be steered/prioritized as A15 nFIQ or nIRQ interrupt requests.

The general features of the AINTC are:

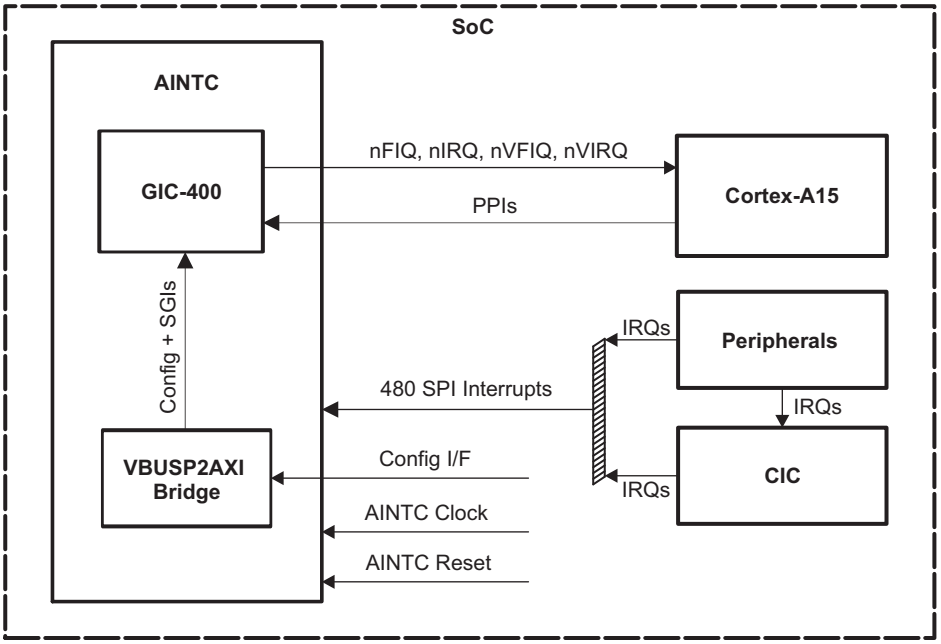
- Up to 480 shared peripheral interrupts (SPI) inputs
- Individual priority for each interrupt input
- Each interrupt can be steered to nFIQ or nIRQ
- Independent priority sorting for nFIQ and nIRQ
- Secure mask flag.

6.1.2.2.1 AINTC Block Diagram

The "core" of the AINTC is an IP block which is called Generic Interrupt Controller (GIC-400) and is provided by Arm. Additionally, the AINTC implements a VBUSP2AXI bridge to provide R/W access to its internal MMRs.

Figure 6-2 shows the block diagram of the AINTC.

Figure 6-2. AINTC Block Diagram



6.1.2.2.1.1 Generic Interrupt Controller (GIC-400)

The GIC-400 is a high-performance, area-optimized interrupt controller with an AMBA AXI interface. It detects, manages, and distributes system interrupts to the Cortex-A15 core. It also supports the separation of two groups of interrupts, typically secure and non-secure interrupts. In addition, the GIC-400 provides hardware acceleration for managing virtualized interrupts.

The GIC-400 implements the following interrupt types:

- 16 *Software Generated Interrupts* (SGIs)
- 7 *Private Peripheral Interrupts* (PPIs)
- 480 *Shared Peripheral Interrupts* (SPIs)

The GIC-400 can assert the following signals to indicate pending interrupts to the A15 CPU:

- Physical interrupts:
  - nFIQCPU[0]
  - nIRQCPU[0]
- Virtual interrupts:
  - nVFIQCPU[0]
  - nVIRQCPU[0]

The GIC-400 has a configuration register that allows software to configure each interrupt line to be level-sensitive or edge-sensitive. Because the hardware assumes all shared peripheral interrupts to be pulses, it is software responsibility to ensure the GICD_ICFGR bits are set-up correctly after reset.

NOTE: The original Arm GIC Security Extension supports both secure interrupts (Group 0) and non-secure interrupts (Group 1) but all transaction made to the GIC are treated as secure transaction in AINTC, so the user has to follow the Secure mode procedure irrespective of Arm A15 Core mode.

Detailed information about the GIC-400 is provided in the *CoreLink GIC-400 Generic Interrupt Controller Technical Reference Manual* and *Arm Generic Interrupt Controller Architecture Specification*.

6.1.2.2.1.2 VBUSP2AXI Bridge

The AINTC uses the AXI interface to provide R/W access to its internal registers. This means that it is necessary to translate VBUSP transactions into AXI transactions. This protocol conversion is handled by the VBUSP2AXI bridge instantiated inside the AINTC. The bridge handles only the protocol conversion of VBUSP into AXI transactions. It does not perform any clock rate adaptation or data width conversion.

6.1.2.2.2 AINTC Interrupt Mapping

Table 6-2 lists the interrupt sources for the AINTC/GIC-400. These interrupt sources correspond to the 480 Shared Peripheral Interrupts (SPIs) supported by the GIC-400. The association between the SPI event numbers and the corresponding GIC-400 ID numbers is also shown in this table.

NOTE: ELM is not supported on this family of devices.

Table 6-2. AINTC Interrupt Sources

SPI Event No.	GIC-400 ID	Event Name	Description
0	ID32	RSTMUX_INT8	Interrupt generated by TIMER_5 when used as WD Timer for ARMSS.
1	ID33	IPC_GR8	BOOT_CFG interprocessor communication register 8 interrupt
2	ID32	SEM_INT8	Semaphore Master 8 grant interrupt
3	ID32	SEM_ERR8	Semaphore Master 8 error interrupt
4	ID32	ARM_NPMUIRQ0	ARMSS PMU interrupt
5	ID32	ARM_NINTERRIRQ	ARMSS internal error interrupt
6	ID32	ARM_NAXIERRIRQ	ARMSS AXI error interrupt
7	ID32	ARM_NCTIIRQ0	ARMSS CTI interrupt
8	ID40	ARM_TBR_DMA	ARMSS trace buffer DMA event
9	ID41	ARM_TBR_ACQ	ARMSS trace buffer acquisition complete interrupt
10	ID42	BOOTCFG_INT	BOOT_CFG boot error interrupt
[15:11]	[ID47:ID43]	RESERVED	Reserved
16	ID48	TIMER_0_INTL	TIMER_0 low interrupt
17	ID49	TIMER_0_INTH	TIMER_0 high interrupt
18	ID50	TIMER_1_INTL	TIMER_1 low interrupt
19	ID51	TIMER_1_INTH	TIMER_1 high interrupt
20	ID52	TIMER_2_INTL	TIMER_2 low interrupt
21	ID53	TIMER_2_INTH	TIMER_2 high interrupt
22	ID54	TIMER_3_INTL	TIMER_3 low interrupt
23	ID55	TIMER_3_INTH	TIMER_3 high interrupt
24	ID56	TIMER_4_INTL	TIMER_4 low interrupt
25	ID57	TIMER_4_INTH	TIMER_4 high interrupt
26	ID58	TIMER_5_INTL	TIMER_5 low interrupt
27	ID59	TIMER_5_INTH	TIMER_5 high interrupt
28	ID60	TIMER_6_INTL	TIMER_6 low interrupt
29	ID61	TIMER_6_INTH	TIMER_6 high interrupt
[47:30]	[ID79:ID62]	RESERVED	Reserved
48	ID80	PCIE_INT0	PCIE interrupt 0
49	ID81	PCIE_INT1	PCIE interrupt 1
50	ID82	PCIE_INT2	PCIE interrupt 2
51	ID83	PCIE_INT3	PCIE interrupt 3
52	ID84	PCIE_INT4	PCIE interrupt 4
53	ID85	PCIE_INT5	PCIE interrupt 5
54	ID86	PCIE_INT6	PCIE interrupt 6

Table 6-2. AINTC Interrupt Sources (continued)

SPI Event No.	GIC-400 ID	Event Name	Description
55	ID87	PCIE_INT7	PCIE interrupt 7
56	ID87	PCIE_INT8	PCIE interrupt 8
57	ID89	PCIE_INT9	PCIE interrupt 9
58	ID90	PCIE_INT10	PCIE interrupt 10
59	ID91	PCIE_INT11	PCIE interrupt 11
60	ID92	PCIE_INT12	PCIE interrupt 12
61	ID93	PCIE_INT13	PCIE interrupt 13
62	ID94	PCIE_ECC_ERROR	PCIe ECC error interrupt
63	ID95	RESERVED	Reserved
64	ID96	SPI_0_INT0	SPI_0 Level 0 interrupt
65	ID97	SPI_0_INT1	SPI_0 Level 1 interrupt
66	ID98	SPI_1_INT0	SPI_1 Level 0 interrupt
67	ID99	SPI_1_INT1	SPI_1 Level 1 interrupt
68	ID100	SPI_2_INT0	SPI_2 Level 0 interrupt
69	ID101	SPI_2_INT1	SPI_2 Level 1 interrupt
70	ID102	SPI_3_INT0	SPI_3 Level 0 interrupt
71	ID103	SPI_3_INT1	SPI_3 Level 1 interrupt
[79:72]	[ID111:ID104]	RESERVED	Reserved
80	ID112	McASP_0_XINT	McASP_0 transmit interrupt
81	ID113	McASP_0_RINT	McASP_0 receive interrupt
82	ID114	McASP_1_XINT	McASP_1 transmit nterrupt
83	ID115	McASP_1_RINT	McASP_1 receive interrupt
84	ID116	McASP_2_XINT	McASP_2 transmit interrupt
85	ID117	McASP_2_RINT	McASP_2 receive interrupt
86	ID118	RESERVED	Reserved
87	ID119	RESERVED	Reserved
88	ID120	I2C_0_INT	I2C_0 interrupt
89	ID121	I2C_1_INT	I2C_1 interrupt
90	ID122	I2C_2_INT	I2C_2 interrupt
[95:91]	[ID127:ID123]	RESERVED	Reserved
96	ID128	MMC_SD_0_INT	MMC/SD_0 interrupt
97	ID129	MMC_SD_1_INT	MMC/SD_1 interrupt
98	ID130	RESERVED	Reserved
99	ID131	RESERVED	Reserved
100	ID132	McBSP_XINT	McBSP transmit interrupt
101	ID133	McBSP_RINT	McBSP receive interrupt
102	ID134	RESERVED	Reserved
103	ID135	RESERVED	Reserved
104	ID136	NSS_CDMA_STARVE_INT	NSS CDMA buffer starvation interrupt
105	ID137	NSS_SWITCH_STAT_INT0	NSS status pending interrupt 0
106	ID138	NSS_SWITCH_STAT_INT1	NSS status pending interrupt 1
107	ID139	NSS_NAVSS_ECC_INT	NSS NAVSS ECC error interrupt
108	ID140	NSS_SA_UL_ECC_INT	NSS SA_UL ECC error interrupt
109	ID141	NSS_SWITCH_ECC_INT	NSS ENET switch ECC error interrupt
110	ID142	NSS_MDIO_LINK_INT0	NSS MDIO link interrupt 0
111	ID143	NSS_CPTS_EVENT_INT	NSS CPTS event pending interrupt
112	ID144	NSS_MDIO_USER_INT0	NSS MDIO user interrupt 0

Table 6-2. AINTC Interrupt Sources (continued)

SPI Event No.	GIC-400 ID	Event Name	Description
113	ID145	NSS_MDIO_USER_INT1	NSS MDIO user interrupt 1
114	ID146	PKA_INT	PKA interrupt
115	ID147	RNG_INT	RNG interrupt
116	ID148	RESERVED	Reserved
117	ID149	RESERVED	Reserved
118	ID150	RESERVED	Reserved
119	ID151	RESERVED	Reserved
120	ID152	MLB_DMA_CH_INT0	MLB DMA channel interrupt 0
121	ID153	MLB_DMA_CH_INT1	MLB DMA channel interrupt 1
122	ID154	MLB_SYS_INT	MLB system interrupt
123	ID155	DDR3_ERR	DDR_EMIF error interrupt
124	ID156	SR_0_SR_I2CINT	SRSS I2C interrupt
125	ID157	RESERVED	Reserved
126	ID158	RESERVED	Reserved
127	ID159	RESERVED	Reserved
128	ID160	USB_0_INT00	USBSS_0 event ring 0 interrupt
129	ID161	USB_0_INT01	USBSS_0 event ring 1 interrupt
130	ID162	USB_0_INT02	USBSS_0 event ring 2 interrupt
131	ID163	USB_0_INT03	USBSS_0 event ring 3 interrupt
132	ID164	USB_0_INT04	USBSS_0 event ring 4 interrupt
133	ID165	USB_0_INT05	USBSS_0 event ring 5 interrupt
134	ID166	USB_0_INT06	USBSS_0 event ring 6 interrupt
135	ID167	USB_0_INT07	USBSS_0 event ring 7 interrupt
136	ID168	USB_0_INT08	USBSS_0 event ring 8 interrupt
137	ID169	USB_0_INT09	USBSS_0 event ring 9 interrupt
138	ID170	USB_0_INT10	USBSS_0 event ring 10 interrupt
139	ID171	USB_0_INT11	USBSS_0 event ring 11 interrupt
140	ID172	USB_0_INT12	USBSS_0 event ring 12 interrupt
141	ID173	USB_0_INT13	USBSS_0 event ring 13 interrupt
142	ID174	USB_0_INT14	USBSS_0 event ring 14 interrupt
143	ID175	USB_0_INT15	USBSS_0 event ring 15 interrupt
144	ID176	USB_1_INT00	USBSS_1 event ring 0 interrupt
145	ID177	USB_1_INT01	USBSS_1 event ring 1 interrupt
146	ID178	USB_1_INT02	USBSS_1 event ring 2 interrupt
147	ID179	USB_1_INT03	USBSS_1 event ring 3 interrupt
148	ID180	USB_1_INT04	USBSS_1 event ring 4 interrupt
149	ID181	USB_1_INT05	USBSS_1 event ring 5 interrupt
150	ID182	USB_1_INT06	USBSS_1 event ring 6 interrupt
151	ID183	USB_1_INT07	USBSS_1 event ring 7 interrupt
152	ID184	USB_1_INT08	USBSS_1 event ring 8 interrupt
153	ID185	USB_1_INT09	USBSS_1 event ring 9 interrupt
154	ID186	USB_1_INT10	USBSS_1 event ring 10 interrupt
155	ID187	USB_1_INT11	USBSS_1 event ring 11 interrupt
156	ID188	USB_1_INT12	USBSS_1 event ring 12 interrupt
157	ID189	USB_1_INT13	USBSS_1 event ring 13 interrupt
158	ID190	USB_1_INT14	USBSS_1 event ring 14 interrupt
159	ID191	USB_1_INT15	USBSS_1 event ring 15 interrupt

Table 6-2. AINTC Interrupt Sources (continued)

SPI Event No.	GIC-400 ID	Event Name	Description
160	ID192	USB_0_OABSINT	USBSS_0 OABS interrupt
161	ID193	USB_0_MISCINT	USBSS_0 miscellaneous interrupt
162	ID194	USB_1_OABSINT	USBSS_1 OABS interrupt
163	ID195	USB_1_MISCINT	USBSS_1 miscellaneous interrupt
164	ID196	UART_0_UARTINT	UART_0 interrupt
165	ID197	UART_1_UARTINT	UART_1 interrupt
166	ID198	UART_2_UARTINT	UART_2 interrupt
[171:167]	[ID203:ID199]	RESERVED	Reserved
172	ID204	EPWM_0_INT	ePWM_0 event trigger (ET) interrupt
173	ID205	EPWM_1_INT	ePWM_1 event trigger (ET) interrupt
174	ID206	EPWM_2_INT	ePWM_2 event trigger (ET) interrupt
175	ID207	EPWM_3_INT	ePWM_3 event trigger (ET) interrupt
176	ID208	EPWM_4_INT	ePWM_4 event trigger (ET) interrupt
177	ID209	EPWM_5_INT	ePWM_5 event trigger (ET) interrupt
178	ID210	EPWM_0_TRIP_ZONE	ePWM_0 tripzone (TZ) interrupt
179	ID211	EPWM_1_TRIP_ZONE	ePWM_1 tripzone (TZ) interrupt
180	ID212	EPWM_2_TRIP_ZONE	ePWM_2 tripzone (TZ) interrupt
181	ID213	EPWM_3_TRIP_ZONE	ePWM_3 tripzone (TZ) interrupt
182	ID214	EPWM_4_TRIP_ZONE	ePWM_4 tripzone (TZ) interrupt
183	ID215	EPWM_5_TRIP_ZONE	ePWM_5 tripzone (TZ) interrupt
184	ID216	EQEP_0_INT	eQEP_0 interrupt
185	ID217	EQEP_1_INT	eQEP_1 interrupt
186	ID218	EQEP_2_INT	eQEP_2 interrupt
187	ID219	ECAP_0_INT	eCAP_0 interrupt
188	ID220	ECAP_1_INT	eCAP_1 interrupt
189	ID221	RESERVED	Reserved
190	ID222	DCAN_0_INT0	DCAN_0 error/data/MO interrupt
191	ID223	DCAN_0_INT1	DCAN_0 message object (MO) interrupt
192	ID224	DCAN_0_ERR_INT	DCAN_0 parity error interrupt
193	ID225	DCAN_1_INT0	DCAN_1 error/data/MO interrupt
194	ID226	DCAN_1_INT1	DCAN_1 message object (MO) interrupt
195	ID227	DCAN_1_ERR_INT	DCAN_1 parity error interrupt
196	ID228	GPMC_INT	GPMC interrupt
197	ID229	ELM_INT	ELM error location process complete interrupt
198	ID230	QSPI_INT	QSPI interrupt
199	ID231	QSPI_ECC_ERR	QSPI ECC error interrupt
200	ID232	EDMACC_0_REGION_0_INT	EDMACC_0 region 0 DMA completion interrupt
201	ID233	EDMACC_0_REGION_1_INT	EDMACC_0 region 1 DMA completion interrupt
202	ID234	EDMACC_0_REGION_2_INT	EDMACC_0 region 2 DMA completion interrupt
203	ID235	EDMACC_0_REGION_3_INT	EDMACC_0 region 3 DMA completion interrupt
204	ID236	EDMACC_0_REGION_4_INT	EDMACC_0 region 4 DMA completion interrupt
205	ID237	EDMACC_0_REGION_5_INT	EDMACC_0 region 5 DMA completion interrupt
206	ID238	EDMACC_0_REGION_6_INT	EDMACC_0 region 6 DMA completion interrupt
207	ID239	EDMACC_0_REGION_7_INT	EDMACC_0 region 7 DMA completion interrupt
208	ID240	EDMACC_1_REGION_0_INT	EDMACC_1 region 0 DMA completion interrupt
209	ID241	EDMACC_1_REGION_1_INT	EDMACC_1 region 1 DMA completion interrupt
210	ID242	EDMACC_1_REGION_2_INT	EDMACC_1 region 2 DMA completion interrupt

Table 6-2. AINTC Interrupt Sources (continued)

SPI Event No.	GIC-400 ID	Event Name	Description
211	ID243	EDMACC_1_REGION_3_INT	EDMACC_1 region 3 DMA completion interrupt
212	ID244	EDMACC_1_REGION_4_INT	EDMACC_1 region 4 DMA completion interrupt
213	ID245	EDMACC_1_REGION_5_INT	EDMACC_1 region 5 DMA completion interrupt
214	ID246	EDMACC_1_REGION_6_INT	EDMACC_1 region 6 DMA completion interrupt
215	ID247	EDMACC_1_REGION_7_INT	EDMACC_1 region 7 DMA completion interrupt
216	ID248	EDMACC_0_MPINT	EDMACC_0 memory protection interrupt
217	ID249	EDMACC_0_ERRINT	EDMACC_0 error interrupt
218	ID250	EDMACC_0_GINT	EDMACC_0 global completion interrupt
219	ID251	EDMACC_1_MPINT	EDMACC_1 memory protection interrupt
220	ID252	EDMACC_1_ERRINT	EDMACC_1 error interrupt
221	ID253	EDMACC_1_GINT	EDMACC_1 global completion interrupt
222	ID254	RESERVED	Reserved
223	ID255	RESERVED	Reserved
224	ID256	PRU-ICSS_0_HOST_INT0	PRU-ICSS_0 host interrupt 0
225	ID257	PRU-ICSS_0_HOST_INT1	PRU-ICSS_0 host interrupt 1
226	ID258	PRU-ICSS_0_HOST_INT2	PRU-ICSS_0 host interrupt 2
227	ID259	PRU-ICSS_0_HOST_INT3	PRU-ICSS_0 host interrupt 3
228	ID260	PRU-ICSS_0_HOST_INT4	PRU-ICSS_0 host interrupt 4
229	ID261	RESERVED	Reserved
230	ID262	PRU-ICSS_0_HOST_INT6	PRU-ICSS_0 host interrupt 6
231	ID263	PRU-ICSS_0_HOST_INT7	PRU-ICSS_0 host interrupt 7
232	ID264	PRU-ICSS_1_HOST_INT0	PRU-ICSS_1 host interrupt 0
233	ID265	PRU-ICSS_1_HOST_INT1	PRU-ICSS_1 host interrupt 1
234	ID266	PRU-ICSS_1_HOST_INT2	PRU-ICSS_1 host interrupt 2
235	ID267	PRU-ICSS_1_HOST_INT3	PRU-ICSS_1 host interrupt 3
236	ID268	PRU-ICSS_1_HOST_INT4	PRU-ICSS_1 host interrupt 4
237	ID269	RESERVED	Reserved
238	ID270	PRU-ICSS_1_HOST_INT6	PRU-ICSS_1 host interrupt 6
239	ID271	PRU-ICSS_1_HOST_INT7	PRU-ICSS_1 host interrupt 7
240	ID272	ASRC_STREAM_IN_INT	ASRC input FIFO interrupt
241	ID273	ASRC_STREAM_OUT_INT	ASRC output FIFO interrupt
242	ID274	ASRC_STREAM_ERR_INT	ASRC error interrupt
243	ID275	ASRC_GROUP_IN_INT	ASRC Group input FIFO interrupt
244	ID276	ASRC_GROUP_OUT_INT	ASRC Group output FIFO interrupt
245	ID277	RESERVED	Reserved
246	ID278	RESERVED	Reserved
247	ID279	DSS_INT	DSS interrupt
248	ID280	DBGTBR_DMAINT	System trace buffer DMA event
249	ID281	DBGTBR_ACQCOMP	System trace buffer acquisition complete interrupt
250	ID282	MSMC_MPF_ERROR1	MSMC memory protection fault indicator for system master PrivID = 1 (Arm CorePac)
251	ID283	PMMC_INT0	PMMC interrupt 0
252	ID284	PMMC_FAULTDET_INT	PMMC safety interrupt
253	ID285	SECMGRINT	Security manager interrupt
254	ID286	PMMCTBR_ACQCOMP	PMMC trace buffer acquisition complete interrupt
255	ID287	PMMCTBR_DMAINT	PMMC trace buffer DMA event
[271:256]	[ID303:ID288]	RESERVED	Reserved
272	ID304	NSS_QPEND0	NSS transmit queue 0 pending interrupt

Table 6-2. AINTC Interrupt Sources (continued)

SPI Event No.	GIC-400 ID	Event Name	Description
273	ID305	NSS_QPEND1	NSS transmit queue 1 pending interrupt
274	ID306	NSS_QPEND2	NSS transmit queue 2 pending interrupt
275	ID307	NSS_QPEND3	NSS transmit queue 3 pending interrupt
276	ID308	NSS_QPEND4	NSS transmit queue 4 pending interrupt
277	ID309	NSS_QPEND5	NSS transmit queue 5 pending interrupt
278	ID310	NSS_QPEND6	NSS transmit queue 6 pending interrupt
279	ID311	NSS_QPEND7	NSS transmit queue 7 pending interrupt
280	ID312	NSS_QPEND8	NSS transmit queue 8 pending interrupt
281	ID313	NSS_QPEND9	NSS transmit queue 9 pending interrupt
282	ID314	NSS_QPEND10	NSS transmit queue 10 pending interrupt
283	ID315	NSS_QPEND11	NSS transmit queue 11 pending interrupt
284	ID316	NSS_QPEND12	NSS transmit queue 12 pending interrupt
285	ID317	NSS_QPEND13	NSS transmit queue 13 pending interrupt
286	ID318	NSS_QPEND14	NSS transmit queue 14 pending interrupt
287	ID319	NSS_QPEND15	NSS transmit queue 15 pending interrupt
288	ID320	NSS_QPEND16	NSS transmit queue 16 pending interrupt
289	ID321	NSS_QPEND17	NSS transmit queue 17 pending interrupt
290	ID322	NSS_QPEND18	NSS transmit queue 18 pending interrupt
291	ID323	NSS_QPEND19	NSS transmit queue 19 pending interrupt
292	ID324	NSS_QPEND20	NSS transmit queue 20 pending interrupt
293	ID325	NSS_QPEND21	NSS transmit queue 21 pending interrupt
294	ID326	NSS_QPEND22	NSS transmit queue 22 pending interrupt
295	ID327	NSS_QPEND23	NSS transmit queue 23 pending interrupt
296	ID328	NSS_QPEND24	NSS transmit queue 24 pending interrupt
297	ID329	NSS_QPEND25	NSS transmit queue 25 pending interrupt
298	ID330	NSS_QPEND26	NSS transmit queue 26 pending interrupt
299	ID331	NSS_QPEND27	NSS transmit queue 27 pending interrupt
300	ID332	NSS_QPEND28	NSS transmit queue 28 pending interrupt
301	ID333	NSS_QPEND29	NSS transmit queue 29 pending interrupt
302	ID334	NSS_QPEND30	NSS transmit queue 30 pending interrupt
303	ID335	NSS_QPEND31	NSS transmit queue 31 pending interrupt
304	ID336	NSS_QPEND32	NSS transmit queue 32 pending interrupt
305	ID337	NSS_QPEND33	NSS transmit queue 33 pending interrupt
306	ID338	NSS_QPEND34	NSS transmit queue 34 pending interrupt
307	ID339	NSS_QPEND35	NSS transmit queue 35 pending interrupt
308	ID340	NSS_QPEND56	NSS transmit queue 56 pending interrupt
309	ID341	NSS_QPEND57	NSS transmit queue 57 pending interrupt
310	ID342	NSS_QPEND58	NSS transmit queue 58 pending interrupt
311	ID343	NSS_QPEND59	NSS transmit queue 59 pending interrupt
312	ID344	NSS_QPEND60	NSS transmit queue 60 pending interrupt
313	ID345	NSS_QPEND61	NSS transmit queue 61 pending interrupt
314	ID346	NSS_QPEND62	NSS transmit queue 62 pending interrupt
315	ID347	NSS_QPEND63	NSS transmit queue 63 pending interrupt
316	ID348	RESERVED	Reserved
317	ID349	RESERVED	Reserved
318	ID350	RESERVED	Reserved
319	ID351	RESERVED	Reserved

Table 6-2. AINTC Interrupt Sources (continued)

SPI Event No.	GIC-400 ID	Event Name	Description
320	ID352	MSGMGR_PROXY_ERROR	Message Manager proxy error interrupt
321	ID353	MSGMGR_FREEINDEX_ERROR	Message Manager no free index available interrupt
322	ID354	MSGMGR_ECC_ERROR	Message Manager ECC error interrupt
323	ID355	RESERVED	Reserved
324	ID356	MSGMGR_QUE_PEND_5	Message Manager queue 5 pending interrupt
325	ID357	MSGMGR_QUE_PEND_37	Message Manager queue 37 pending interrupt
326	ID358	MSGMGR_QUE_PEND_49	Message Manager queue 49 pending interrupt
327	ID359	MSGMGR_QUE_PEND_57	Message Manager queue 57 pending interrupt
[335:328]	[ID367:ID360]	RESERVED	Reserved
336	ID368	CIC_0_OUT0	CIC output 0 host interrupt
337	ID369	CIC_0_OUT1	CIC output 1 host interrupt
338	ID370	CIC_0_OUT2	CIC output 2 host interrupt
339	ID371	CIC_0_OUT3	CIC output 3 host interrupt
340	ID372	CIC_0_OUT4	CIC output 4 host interrupt
341	ID373	CIC_0_OUT5	CIC output 5 host interrupt
342	ID374	CIC_0_OUT6	CIC output 6 host interrupt
343	ID375	CIC_0_OUT7	CIC output 7 host interrupt
344	ID376	CIC_0_OUT8	CIC output 8 host interrupt
345	ID377	CIC_0_OUT9	CIC output 9 host interrupt
346	ID378	CIC_0_OUT10	CIC output 10 host interrupt
347	ID379	CIC_0_OUT11	CIC output 11 host interrupt
348	ID380	CIC_0_OUT12	CIC output 12 host interrupt
349	ID381	CIC_0_OUT13	CIC output 13 host interrupt
350	ID382	CIC_0_OUT14	CIC output 14 host interrupt
351	ID383	CIC_0_OUT15	CIC output 15 host interrupt
352	ID384	CIC_0_OUT16	CIC output 16 host interrupt
353	ID385	CIC_0_OUT17	CIC output 17 host interrupt
354	ID386	CIC_0_OUT18	CIC output 18 host interrupt
355	ID387	CIC_0_OUT19	CIC output 19 host interrupt
356	ID388	CIC_0_OUT20	CIC output 20 host interrupt
357	ID389	CIC_0_OUT21	CIC output 21 host interrupt
358	ID390	CIC_0_OUT22	CIC output 22 host interrupt
359	ID391	CIC_0_OUT23	CIC output 23 host interrupt
360	ID392	CIC_0_OUT24	CIC output 24 host interrupt
361	ID393	CIC_0_OUT25	CIC output 25 host interrupt
362	ID394	CIC_0_OUT26	CIC output 26 host interrupt
363	ID395	CIC_0_OUT27	CIC output 27 host interrupt
364	ID396	CIC_0_OUT28	CIC output 28 host interrupt
365	ID397	CIC_0_OUT29	CIC output 29 host interrupt
366	ID398	CIC_0_OUT30	CIC output 30 host interrupt
367	ID399	CIC_0_OUT31	CIC output 31 host interrupt
368	ID400	CIC_0_OUT32	CIC output 32 host interrupt
369	ID401	CIC_0_OUT33	CIC output 33 host interrupt
370	ID402	CIC_0_OUT34	CIC output 34 host interrupt
371	ID403	CIC_0_OUT35	CIC output 35 host interrupt
372	ID404	CIC_0_OUT36	CIC output 36 host interrupt
373	ID405	CIC_0_OUT37	CIC output 37 host interrupt

Table 6-2. AINTC Interrupt Sources (continued)

SPI Event No.	GIC-400 ID	Event Name	Description
374	ID406	CIC_0_OUT38	CIC output 38 host interrupt
375	ID407	CIC_0_OUT39	CIC output 39 host interrupt
376	ID408	CIC_0_OUT80	CIC output 80 host interrupt
377	ID409	CIC_0_OUT81	CIC output 81 host interrupt
378	ID410	CIC_0_OUT82	CIC output 82 host interrupt
379	ID411	CIC_0_OUT83	CIC output 83 host interrupt
380	ID412	CIC_0_OUT84	CIC output 84 host interrupt
381	ID413	CIC_0_OUT85	CIC output 85 host interrupt
382	ID414	CIC_0_OUT86	CIC output 86 host interrupt
383	ID415	CIC_0_OUT87	CIC output 87 host interrupt
[431:384]	[ID463:ID416]	RESERVED	Reserved
432	ID464	GPIO_0_BANK0_INT	GPIO_0 bank 0 interrupt
433	ID465	GPIO_0_BANK1_INT	GPIO_0 bank 1 interrupt
434	ID466	GPIO_0_BANK2_INT	GPIO_0 bank 2 interrupt
435	ID467	GPIO_0_BANK3_INT	GPIO_0 bank 3 interrupt
436	ID468	GPIO_0_BANK4_INT	GPIO_0 bank 4 interrupt
437	ID469	GPIO_0_BANK5_INT	GPIO_0 bank 5 interrupt
438	ID470	GPIO_0_BANK6_INT	GPIO_0 bank 6 interrupt
439	ID471	GPIO_0_BANK7_INT	GPIO_0 bank 7 interrupt
440	ID472	GPIO_0_BANK8_INT	GPIO_0 bank 8 interrupt
441	ID473	RESERVED	Reserved
442	ID474	GPIO_1_BANK0_INT	GPIO_1 bank 0 interrupt
443	ID475	GPIO_1_BANK1_INT	GPIO_1 bank 1 interrupt
444	ID476	GPIO_1_BANK2_INT	GPIO_1 bank 2 interrupt
445	ID477	GPIO_1_BANK3_INT	GPIO_1 bank 3 interrupt
446	ID478	GPIO_1_BANK4_INT	GPIO_1 bank 4 interrupt
447	ID479	RESERVED	Reserved
448	ID480	GPIOMUX_INT0	GPIOMUX0 output
449	ID481	GPIOMUX_INT1	GPIOMUX1 output
450	ID482	GPIOMUX_INT2	GPIOMUX2 output
451	ID483	GPIOMUX_INT3	GPIOMUX3 output
452	ID484	GPIOMUX_INT4	GPIOMUX4 output
453	ID485	GPIOMUX_INT5	GPIOMUX5 output
454	ID486	GPIOMUX_INT6	GPIOMUX6 output
455	ID487	GPIOMUX_INT7	GPIOMUX7 output
456	ID488	GPIOMUX_INT8	GPIOMUX8 output
457	ID489	GPIOMUX_INT9	GPIOMUX9 output
458	ID490	GPIOMUX_INT10	GPIOMUX10 output
459	ID491	GPIOMUX_INT11	GPIOMUX11 output
460	ID492	GPIOMUX_INT12	GPIOMUX12 output
461	ID493	GPIOMUX_INT13	GPIOMUX13 output
462	ID494	GPIOMUX_INT14	GPIOMUX14 output
463	ID495	GPIOMUX_INT15	GPIOMUX15 output
464	ID496	GPIOMUX_INT16	GPIOMUX16 output
465	ID497	GPIOMUX_INT17	GPIOMUX17 output
466	ID498	GPIOMUX_INT18	GPIOMUX18 output
467	ID499	GPIOMUX_INT19	GPIOMUX19 output

Table 6-2. AINTC Interrupt Sources (continued)

SPI Event No.	GIC-400 ID	Event Name	Description
468	ID500	GPIOMUX_INT20	GPIOMUX20 output
469	ID501	GPIOMUX_INT21	GPIOMUX21 output
470	ID502	GPIOMUX_INT22	GPIOMUX22 output
471	ID503	GPIOMUX_INT23	GPIOMUX23 output
472	ID504	GPIOMUX_INT24	GPIOMUX24 output
473	ID505	GPIOMUX_INT25	GPIOMUX25 output
474	ID506	GPIOMUX_INT26	GPIOMUX26 output
475	ID507	GPIOMUX_INT27	GPIOMUX27 output
476	ID508	GPIOMUX_INT28	GPIOMUX28 output
477	ID509	GPIOMUX_INT29	GPIOMUX29 output
478	ID510	GPIOMUX_INT30	GPIOMUX30 output
479	ID511	GPIOMUX_INT31	GPIOMUX31 output

6.1.2.2.3 AINTC Clock

The AINTC has a single clock domain (CHIP_CLK1/3).

6.1.2.2.4 AINTC Reset

There is a single reset coming into the AINTC module (LPSC6.MOD_G_RST). The GIC-400 is reset asynchronously whereas the VBUSPAXI bridge and other miscellaneous logic is reset synchronously.

6.1.2.3 ARMSS Timers and Watchdog Timer

The following ARMSS-dedicated timer modules are supported in the SoC:

- 4 timers integrated inside the Arm Cluster; for general-purpose use
- Additional timer integrated at SoC level – TIMER_5; its primary purpose is to serve as a Watchdog Timer but it can also be used in general-purpose mode
- Global Timebase Counter (GTC); it is used for synchronizing the ARMSS internal timers.

The four ARMSS internal timers are described in detail in the *Arm Cortex-A15 MPCore Processor Technical Reference Manual*.

The need for a SoC-level watchdog timer (TIMER_5) that is closely-coupled to the Arm processor is driven by the fact that the ARMSS internal timers are not suspendable on CPU halt. TIMER_5 operation is controlled by the DPSC module which is integrated inside the Arm Subsystem. For detailed description of TIMER_5 functionality, see Section 11.17, *Timers*.

The GTC is a simple non-configurable 64-bit counter. The only time this counter gets reset is during power-on. The initialization value of the counter during power-on-reset is all zeros. After reset, the counter keeps incrementing by 1'b1 on each rising edge of the GTC clock. Before the counter value is sent out to the timers inside the A15 core, its value is transcoded from 64-bit graycode (output from the GTC) into 64-bit binary code – **CNTVALUEB[63:0]**, required by the Arm core.

6.1.2.4 ARMSS Boot

The Arm Subsystem has a dedicated 256KB boot ROM (ARM_BOOTROM) starting at address location 0x0000_0000. It can be divided into a secure and non-secure portion in a flexible way. The A15 core always boots in *Secure Monitor* mode and always from this location. This is true irrespective of the device type (secure or non-secure). After the initialization of the *Secure Monitor* mode, the ARM_BOOTROM will execute from secure or non-secure portion based on the device type.

There is also a bootmode pin to define whether ARMSS or DSP is the boot master in the system. For more information about selecting the boot master and a detailed boot flow diagram, see Chapter 4, *Initialization*.

6.1.2.5 ARMSS Power Management, Clocking and Reset

6.1.2.5.1 ARMSS Power

The Arm Subsystem supports a hierarchical power down/power up mechanism based on the interaction between the chip-level (Global) Power Sleep Controller (GPSC) and the Discrete PSC (DPSC) which is local to the Arm Subsystem. The handshaking between the GPSC and the DPSC is built in hardware state machine, and no software intervention is needed.

In this SoC, both Arm Subsystem and A15 core reside in the same power domain (PD9) which is completely under GPSC's control only. Although the DPSC has some registers for A15 individual power state control, they are reserved and software should not modify them.

By default (after reset), the ARMSS power domain is ON.

CAUTION

The DPSC registers dedicated to A15 power management should not be used and can cause undefined behavior (for example, hang) if used.

6.1.2.5.1.1 Powering Down the ARMSS

Because the A15 core and the Arm Subsystem share the same power domain, they are powered OFF at the same time (once such request is initiated). For ARMSS power down, it is software responsibility to do the following:

1. Make sure that A15 executes WFI. The CPU is expected to reach WFI state following a series of steps in software, as follows:
    1. In software, block the cache allocation (by clearing the SCTLR.C or HSCTLR.C bit)
    2. Clean and invalidate all data from the L1 data cache
    3. Take the CPU out of coherency (by clearing the ACTLR.SMP bit which switches the CPU from SMP mode to Asymmetric mode)
    4. Save the architectural state, if required
    5. Execute an instruction barrier ISB instruction to commit all CP15 changes from previous steps
    6. Execute a data barrier DSB instruction to commit all coherency operations from previous steps
    7. Execute a WFI instruction
2. Initiate ARMSS power down request by programming the following GPSC register settings:
    1. Write '0' to the PDCTL9[0] NEXT bit
    2. Write ‘1’ to the PTCMD[9] GO9 bit

The ARMSS power domain state can be verified by reading the PDSTAT9[1-0] STATE bit field in the GPSC.

For detailed description of the A15 SCTLR, HSCTLR, and ACTLR registers, see the *Arm Cortex-A15 MPCore Processor Technical Reference Manual*.

For detailed description of the GPSC PDCTL, PDSTAT, and PTCMD registers, see Section 5.2.2, *Power Sleep Controller (PSC)*.

6.1.2.5.1.2 Powering Up the ARMSS

In order to power up the ARMSS domain, software must do the following GPSC register settings:

1. Write '1' to the PDCTL9[0] NEXT bit
2. Write ‘1’ to the PTCMD[9] GO9 bit

6.1.2.5.1.3 Interaction of Power Down/Power Up with Watchdog Timer

As noted in Section 6.1.2.3, a watchdog timer that is closely-coupled to the Arm core is instantiated at SoC level (TIMER_5).

When the A15 core is powered down, it will not be able to maintain its periodic watchdog timer counter clearing schedule, hence causing spurious watchdog timer counter expiration and global reset. To avoid this scenario, the watchdog timer clock is turned off and it is held in reset, while the A15 core is powered down. This is automatically done by the DPSC hardware (no software intervention is needed).

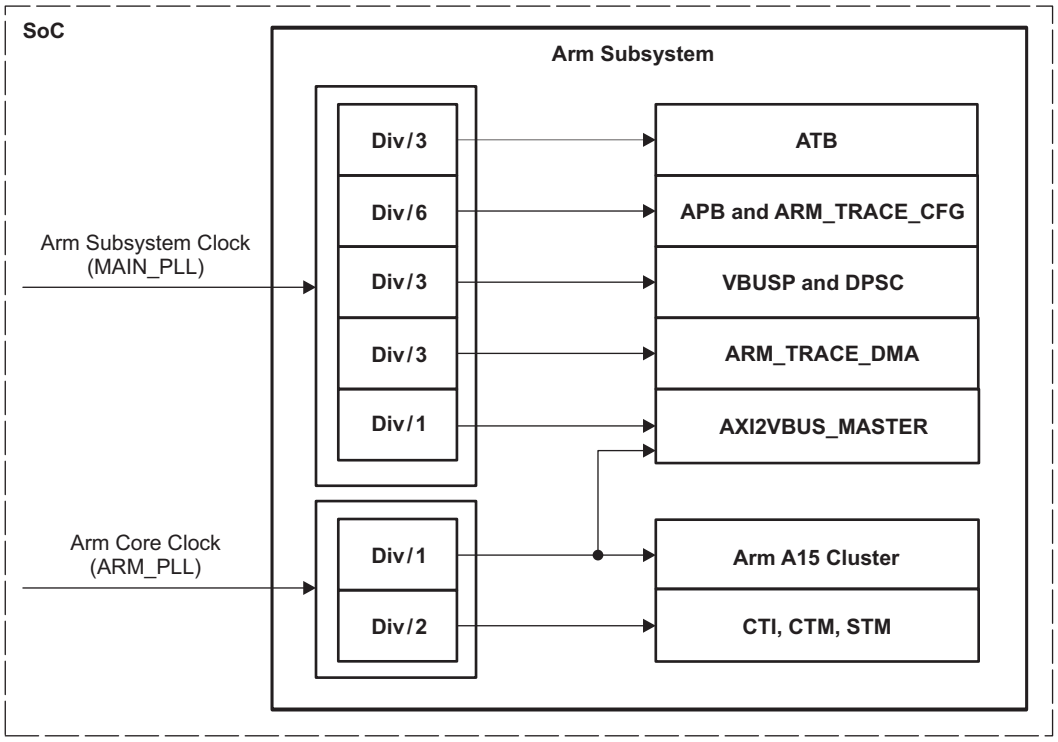
6.1.2.5.2 ARMSS Clock

The Arm subsystem has two primary functional clock inputs:

- **From SoC main PLL (MAIN_PLL).** This clock feeds the following ARMSS modules:
  - Through local DIV/1 divider: AXI2VBUS_MASTER
  - Through local DIV/3 divider: ATB, ARM_VBUSP, DPSC, and ARM_TRACE_DMA
  - Through local DIV/6 divider: APB, and ARM_TRACE_CFG
- **From dedicated ARM_PLL**. This clock feeds the following ARMSS modules:
  - Through local DIV/1 divider: A15 Cluster
  - Through local DIV/2 divider: CTI, CTM, and STM located at ARMSS level

Figure 6-3 shows the Arm Subsystem clock distribution.

Figure 6-3. Arm Subsystem Clock Distribution



For more details on MAIN_PLL and ARM_PLL, see Section 5.4, *Clock Management*.

6.1.2.5.3 ARMSS Reset

The Arm Subsystem supports a number of resets for its internal modules. The list of resets connected to the Arm Subsystem is given in Section 5.3, *Reset Management*.

The Arm Subsystem does not support the software reset feature; there is no reset hookup between the Global PSC and the Arm Subsystem. To truly support a soft reset, the Arm Subsystem must go through a full power down software sequence and hardware disconnect sequence from the interconnect. Neither of these sequences is likely to complete in the midst of a CPU hang condition (which is the reason for a soft reset in the first place). The only recovery mechanism is therefore likely to be a system-level global warm reset.

6.1.2.6 ARMSS Debug

The ARMSS debug components and features are described in Chapter 12, *On-Chip Debug*.

6.1.2.7 ARMSS Interprocessor Communication (IPC)

The Arm A15 core can communicate with other device cores (C66x DSP, PRU-ICSS PRUs) by supporting interrupt generation to and from these cores. The interprocessor communication (IPC) interrupts are assigned in the BOOT_CFG memory-mapped registers (MMRs) called IPCGRx / IPCARx. For more information, see Section 5.1, *Control Module (BOOT_CFG)*.

6.1.3 Arm Subsystem Registers

This chapter does not provide register details for the Arm Cortex-A15 and other submodules except for the ARM_VBUSP registers and the AXI2VBUS_MASTER registers.

For more detailed information about the A15 core (revision r2p2) and other Arm modules (AINTC, CoreSight components, etc), visit the Arm information website: http://infocenter.arm.com/help/index.jsp.

6.1.3.1 AXI2VBUS_MASTER Registers

Table 6-4 lists the memory-mapped registers for the AXI2VBUS_MASTER. All register offset addresses not listed in Table 6-4 should be considered as reserved locations and the register contents should not be modified.

Table 6-3. AXI2VBUS_MASTER Instances

Instance	Base Address
AXI2VBUS_MASTER	0100 0000h

Table 6-4. AXI2VBUS_MASTER Registers

Offset	Acronym	Register Name	AXI2VBUS_MASTER Physical Address	Section
00h	AXI2VBUS_PID	Peripheral Identification Register	0100 0000h	Section 6.1.3.1.1
20h	AXI2VBUS_CMD_PRI	Command Priority Setup Register	0100 0020h	Section 6.1.3.1.2
30h	AXI2VBUS_CPU0_END	Endian Mode Setup Register	0100 0030h	Section 6.1.3.1.3

Table 6-5. AXI2VBUS_MASTER Registers – R/W Modes

Register Name	Readable Mode	Writable Mode
AXI2VBUS_PID	All	None
AXI2VBUS_CMD_PRI	All	Emulation, Supervisor
AXI2VBUS_CPU0_END	All	Emulation, Supervisor

6.1.3.1.1 AXI2VBUS_PID Register (Offset = 0h) [reset = 49200000h]

The Peripheral Identification Register uniquely identifies the AXI2VBUS_MASTER and the specific revision of the AXI2VBUS_MASTER. AXI2VBUS_PID is shown in Figure 6-4 and described in Table 6-7.

Table 6-6. AXI2VBUS_PID Instances

Instance	Physical Address
AXI2VBUS_MASTER	0100 0000h

Figure 6-4. AXI2VBUS_PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PID																															
R-49200000h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-7. AXI2VBUS_PID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PID	R	49200000h	TI internal data. Identifies revision of peripheral.

Table 6-8. Register Call Summary for AXI2VBUS_PID

Arm Subsystem Registers
- AXI2VBUS_MASTER Registers: [0][1]
- AXI2VBUS_PID Register (Offset = 0h) [reset = 49200000h]: [0]

6.1.3.1.2 AXI2VBUS_CMD_PRI Register (Offset = 20h) [reset = 7h]

This register provides the command priority information about the module. The reset value of this register is 7h, the lowest VBUSM priority. Writing to this MMR only updates future transaction of VBUSM priority signal with the written value. AXI2VBUS_CMD_PRI is shown in Figure 6-5 and described in Table 6-10.

Table 6-9. AXI2VBUS_CMD_PRI Instances

Instance	Physical Address
AXI2VBUS_MASTER	0100 0020h

Figure 6-5. AXI2VBUS_CMD_PRI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													PRI		
R-0h																													R/W-7h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-10. AXI2VBUS_CMD_PRI Register Field Definitions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Read returns reset value
2-0	PRI	R/W	7h	Set priority of the command. Default is 7h which means the lowest VBUSM priority

Table 6-11. Register Call Summary for AXI2VBUS_CMD_PRI

Arm Subsystem Registers
- AXI2VBUS_MASTER Registers: [0][1]
- AXI2VBUS_CMD_PRI Register (Offset = 20h) [reset = 7h]: [0]

6.1.3.1.3 AXI2VBUS_CPU0_END Register (Offset = 30h) [reset = FFFFFFFFh]

This register provides AXI2VBUS_MASTER module endian mode setup and status for the CPU. AXI2VBUS_CPU0_END is shown in Figure 6-6 and described in Table 6-13.

Table 6-12. AXI2VBUS_CPU0_END Instances

Instance	Physical Address
AXI2VBUS_MASTER	0100 0030h

Figure 6-6. AXI2VBUS_CPU0_END Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENDIAN																															
R/W-FFFFFFFFh																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-13. AXI2VBUS_CPU0_END Register Field Definitions

Bit	Field	Type	Reset	Description
31-0	ENDIAN	R/W	FFFFFFFFh	CPU Endian mode configuration and status register • Little endian: 0xFFFFFFFF (all ones)

Table 6-14. Register Call Summary for AXI2VBUS_CPU0_END

Arm Subsystem Registers
- AXI2VBUS_MASTER Registers: [0][1]
- AXI2VBUS_CPU0_END Register (Offset = 30h) [reset = 0h]: [0]

6.1.3.2 ARM_VBUSP Registers

Table 6-16 lists the memory-mapped registers for the ARM_VBUSP. All register offset addresses not listed in Table 6-16 should be considered as reserved locations and the register contents should not be modified.

Table 6-15. ARM_VBUSP Instances

Instance	Base Address
ARM_VBUSP	01E8 0000h

Table 6-16. ARM_VBUSP Registers

Offset	Acronym	Register Name	ARM_VBUSP Physical Address	Section
0h	ARM_PID	Peripheral Identification Register for Arm Subsystem	01E8 0000h	Section 6.1.3.2.1
4h	ARM_INTC_PID	Peripheral Identification Register for AINTC	01E8 0004h	Section 6.1.3.2.2
14h	STM_DISABLE	Disable access to STM in Arm Subsystem	01E8 0014h	Section 6.1.3.2.3
400h	PD_CPU0_PTCMD	Power domain transition command for A15 core	01E8 0400h	Section 6.1.3.2.4
404h	PD_CPU0_PDSTAT	Power domain status for A15 core	01E8 0404h	Section 6.1.3.2.5
408h	PD_CPU0_PDCTL	Power domain control for A15 core	01E8 0408h	Section 6.1.3.2.6

CAUTION

The DPSC registers dedicated to A15 power management should not be used and can cause undefined behavior (for example, hang) if used. See Section 6.1.2.5 for details.

6.1.3.2.1 ARM_PID Register (Offset = 0h) [reset = 44900900h]

ARM_PID is shown in Figure 6-7 and described in Table 6-18.

Peripheral Identification Register for Arm Subsystem.

Table 6-17. ARM_PID Instances

Instance	Physical Address
ARM_VBUSP	01E8 0000h

Figure 6-7. ARM_PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PID																															
R-44900900h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-18. ARM_PID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PID	R	44900900h	Peripheral Identification Register for Arm Subsystem

Table 6-19. Register Call Summary for ARM_PID

Arm Subsystem Registers
- ARM_PID Register (Offset = 0h) [reset = 44900900h]: [0]
- ARM_VBUSP Registers: [0]

6.1.3.2.2 ARM_INTC_PID Register (Offset = 4h) [reset = 44910900h]

ARM_INTC_PID is shown in Figure 6-8 and described in Table 6-21.

Peripheral Identification Register for AINTC.

Table 6-20. ARM_INTC_PID Instances

Instance	Physical Address
ARM_VBUSP	01E8 0004h

Figure 6-8. ARM_INTC_PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PID																															
R-44910900h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-21. ARM_INTC_PID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PID	R	44910900h	Peripheral Identification Register for AINTC

Table 6-22. Register Call Summary for ARM_INTC_PID

Arm Subsystem Registers

- ARM_INTC_PID Register (Offset = 4h) [reset = 44910900h]: [0]
- ARM_VBUSP Registers: [0]

6.1.3.2.3 STM_DISABLE Register (Offset = 14h) [reset = 1h]

STM_DISABLE is shown in Figure 6-9 and described in Table 6-24.

Disable access to STM in Arm Subsystem.

Table 6-23. STM_DISABLE Instances

Instance	Physical Address
ARM_VBUSP	01E8 0014h

Figure 6-9. STM_DISABLE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STM_DISABLE																															
R/W-1h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-24. STM_DISABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STM_DISABLE	R/W	1h	Disable access to STM in Arm Subsystem • 0: STM feature enabled • 1: STM feature disabled

Table 6-25. Register Call Summary for STM_DISABLE

Arm Subsystem Registers
- STM_DISABLE Register (Offset = 14h) [reset = 1h]: [0][1][2]
- ARM_VBUSP Registers: [0]

6.1.3.2.4 PD_CPU0_PTCMD Register (Offset = 400h) [reset = X]

PD_CPU0_PTCMD is shown in Figure 6-10 and described in Table 6-27.

Power Domain Transition Command for A15 core.

Table 6-26. PD_CPU0_PTCMD Instances

Instance	Physical Address
ARM_VBUSP	01E8 0400h

Figure 6-10. PD_CPU0_PTCMD Register

31	30	29	28	27	26	25	24
RESERVED							
W-X							
23	22	21	20	19	18	17	16
RESERVED							
W-X							
15	14	13	12	11	10	9	8
RESERVED							
W-X							
7	6	5	4	3	2	1	0
RESERVED							GO_CPU
W-X							W-0h

LEGEND: W = Write Only; *-n* = value after reset

Table 6-27. PD_CPU0_PTCMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	W	X	
0	GO_CPU	W	0h	CPU power domain GO transition. • 0: Writes of 0 have no effect • 1: Writes of 1 causes the Arm CorePac DPSC hardware to transit domain to PDCTL.NEXT_CPU state programmed one

Table 6-28. Register Call Summary for PD_CPU0_PTCMD

Arm Subsystem Registers • PD_CPU0_PTCMD Register (Offset = 400h) [reset = X]: [0] • ARM_VBUSP Registers: [0]

6.1.3.2.5 PD_CPU0_PDSTAT Register (Offset = 404h) [reset = X]

PD_CPU0_PDSTAT is shown in Figure 6-11 and described in Table 6-30.

Power Domain Status for A15 core.

Table 6-29. PD_CPU0_PDSTAT Instances

Instance	Physical Address
ARM_VBUSP	01E8 0404h

Figure 6-11. PD_CPU0_PDSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-X							
23	22	21	20	19	18	17	16
RESERVED						DOMAIN_STATE	
R-X						R-X	
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						NEXT_CPU	
R-0h						R-0h	

LEGEND: R = Read Only; *-n* = value after reset

Table 6-30. PD_CPU0_PDSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	X	Reserved
18-16	DOMAIN_STATE	R	X	Shows CPU power domain ACTUAL state. • 000: Powered ON • 001: Powered ON and held-in-reset • 010: Powered OFF. Dynamic wakeup on interrupt. • 011: Powered OFF. • 100: Domain in transition NEXT state should always match current domain state unless it is in transition. The only exception is when debug prevents powering down.
15-2	RESERVED	R	0h	Reserved
1-0	NEXT_CPU	R	0h	Shows programmed CPU power domain NEXT state. • 00: Powered ON • 01: Powered ON and held-in-reset • 10: Powered OFF. Dynamic wakeup on interrupt. • 11: Powered OFF. These bits essentially reflect the NEXT state stored in the PDCTL shadow register inside the Arm CorePac. This field may not match the fields programmed into PDCTL until a GO transition command is issued

Table 6-31. Register Call Summary for PD_CPU0_PDSTAT

Arm Subsystem Registers
• PD_CPU0_PDSTAT Register (Offset = 404h) [reset = X]: [0]
• ARM_VBUSP Registers: [0]

6.1.3.2.6 PD_CPU0_PDCTL Register (Offset = 408h) [reset = X]

PD_CPU0_PDCTL is shown in Figure 6-12 and described in Table 6-33.

Power Domain Control for A15 core.

Table 6-32. PD_CPU0_PDCTL Instances

Instance	Physical Address
ARM_VBUSP	01E8 0408h

Figure 6-12. PD_CPU0_PDCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-X							
23	22	21	20	19	18	17	16
RESERVED							
R/W-X							
15	14	13	12	11	10	9	8
RESERVED							
R/W-X							
7	6	5	4	3	2	1	0
RESERVED						NEXT_CPU	
R/W-X						R/W-0h	

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-33. PD_CPU0_PDCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	X	Reserved
1-0	NEXT_CPU	R/W	0h	CPU power domain NEXT state. • 00: Powered ON • 01: Powered ON and held-in-reset • 10: Powered OFF. Dynamic wakeup on interrupt • 11: Powered OFF

Table 6-34. Register Call Summary for PD_CPU0_PDCTL

Arm Subsystem Registers
- PD_CPU0_PDCTL Register (Offset = 408h) [reset = X]: [0]
- ARM_VBUSP Registers: [0]

6.2 C66x DSP Subsystem

The purpose of this section is to provide an overview of the DSP subsystem (C66x CorePac) along with some integration details.

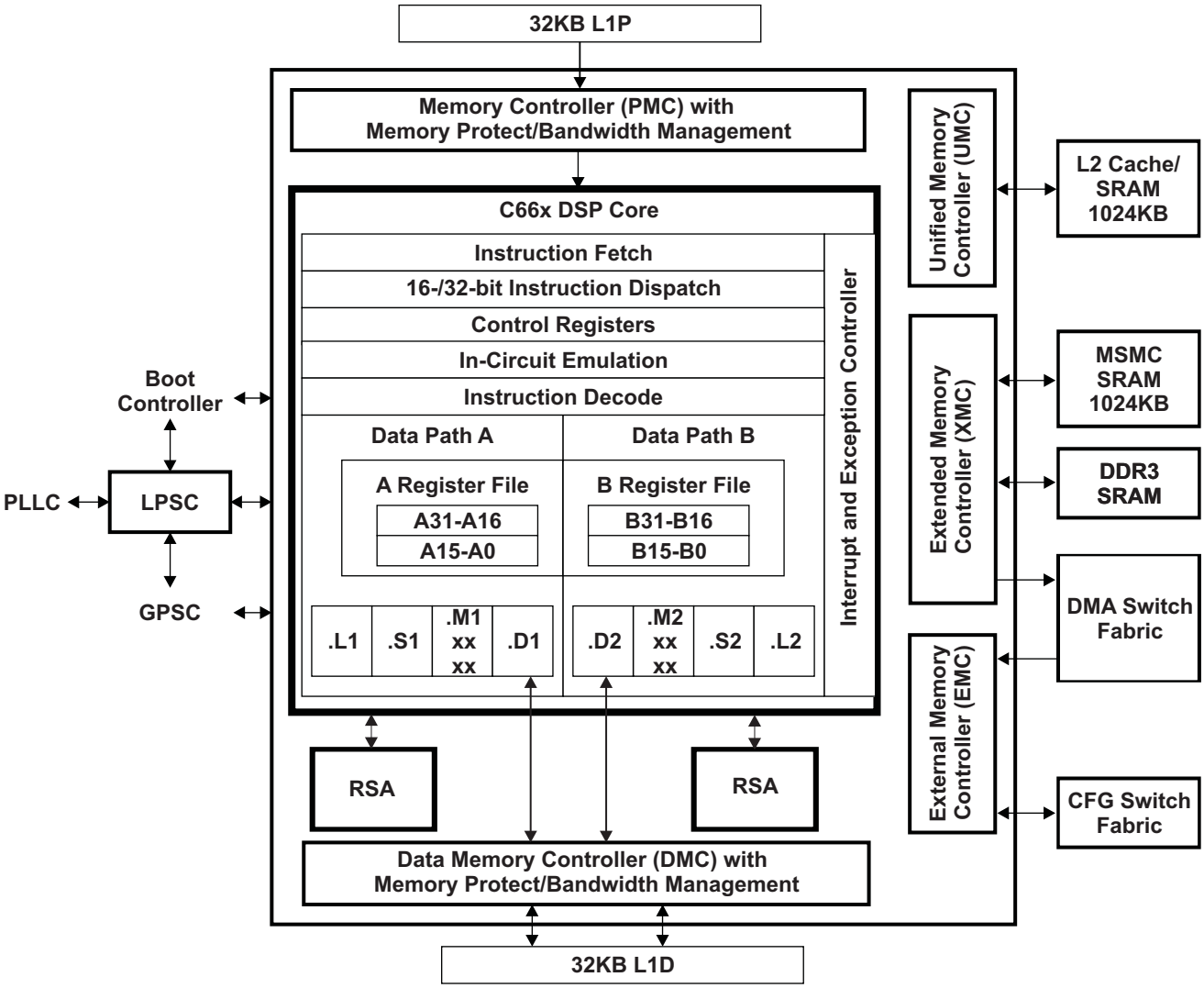
6.2.1 DSP Subsystem Overview

The DSP sybsystem (C66x CorePac) supports the following key features:

- Fixed/Floating-point C66x CPU based on a superset of the C64x+ and C67x+ ISA
- Program Memory Controller (PMC):
  - 32KB Level 1 Program (L1P) Cache/SRAM
- Data Memory Controller (DMC):
  - 32KB L1 Data (L1D) Cache/SRAM
- Unified Memory Controller (UMC):
  - 1024KB L2 Cache/SRAM
- External Memory Controller (EMC):
  - Internal DMA (IDMA) engine
  - One 128-bit VBUSM slave port from TeraNet_DMA
  - One 32-bit VBUSP master port to TeraNet_CFG
- XMC (Extended Memory Controller):
  - One 256-bit port to MSMC controller
- Multistream prefetch buffer
- Address extension/translation (32-bit to 36-bit)
- Memory protection for multiple segments
- Memory protection for all internal L1/L2 RAM
- Error Detection for L1P
- Error Detection and Correction for L1D
- Error Detection and Correction for all L2
- Integrated C66x CorePac interrupt controller (INTC) that works in conjunction with Chip-level Interrupt Controller (CIC) for distribution of system interrupts to the C66x core. Interrupts can be routed directly to the C66x core or through the CIC module in a flexible manner
- Integrated leakage and dynamic power management
- Debug/emulation capabilities:
  - Support for halt mode, real time and monitor mode debug capabilities
  - Support for processor instruction trace and system trace (**printf**-style debug)
- Dedicated timer module (TIMER_0) for the C66x CorePac, integrated at SoC level. TIMER_0 can be used as either general-purpose timer or watchdog timer

Figure 6-13 shows an overview of the C66x CorePac.

Figure 6-13. C66x CorePac Overview



Fore more information about:

- C66x CorePac, see the *TMS320C66x DSP CorePac User Guide* (SPRUGW0).
- C66x CPU core, see the *TMS320C66x DSP CPU and Instruction Set Reference Guide* (SPRUGH7).
- C66x cache memory system, see the *TMS320C66x DSP Cache User Guide* (SPRUGY8).
- C66x debug/trace support, see Chapter 12, *On-Chip Debug*.

6.2.2 DSP Subsystem Integration

This section describes the integration of the module in the device, including information about clocks, resets, and hardware requests.

Table 6-35 through Table 6-36 summarize the integration of the DSP subsystem.

Table 6-35. DSP Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
DSP	PD8	LPSC18	TeraNet

Table 6-36. DSP Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
DSP	DSP_CLK	CHIP_CLK1	PLL Controller	DSP Subsystem clock. It drives the various DSP internal resources via dividers (1/1; 1/2; 1/3; 1/4)
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
DSP	DSP_GRST	MOD_G_RST	LPSC18	Global Reset
	DSP_LRST	LRST	LPSC18	Local Reset
	DSP_POR_Z	POR	PSM8	Power on Reset
	GEM_TRSTZ			Reset from DEBUGSS

Refer to Chapter 5 for details on DSP power/clock/reset management.

6.2.2.1 DSP Interrupt Sources

Table 6-37 shows the C66x DSP interrupt sources.

Table 6-37. C66x DSP Interrupt Sources

Event No.	Event Source Name	Description
0	EVT0	C66x CorePac internal event
1	EVT1	C66x CorePac internal event
2	EVT2	C66x CorePac internal event
3	EVT3	C66x CorePac internal event
4	PRU-ICSS_0_HOST_INT0	PRU-ICSS_0 host interrupt 0
5	PRU-ICSS_1_HOST_INT0	PRU-ICSS_1 host interrupt 0
6	EDMACC_0_REGION_0_INT	EDMACC_0 region 0 DMA completion interrupt
7	EDMACC_1_REGION_0_INT	EDMACC_1 region 0 DMA completion interrupt
8	IPC_GR0	BOOT_CFG interprocessor communication register 0 interrupt
9	EMU_DTDMA	C66x CorePac internal event
10	MSMC_MPF_ERROR0	MSMC memory protection fault indicator for system master PrivID = 0 (C66x CorePac)
11	EMU_RTDXRX	C66x CorePac internal event
12	EMU_RTDXTX	C66x CorePac internal event
13	IDMA0	C66x CorePac internal event
14	IDMA1	C66x CorePac internal event
15	SEM_ERR0	Semaphore Master 0 error interrupt
16	SEM_INT0	Semaphore Master 0 grant interrupt
17	PCIE_INT4	PCIE interrupt 4

Table 6-37. C66x DSP Interrupt Sources (continued)

Event No.	Event Source Name	Description
18	RESERVED	Reserved
19	RESERVED	Reserved
20	MSGMGR_QUE_PEND_4	Message Manager queue 0 pending interrupt
21	MSGMGR_QUE_PEND_36	Message Manager queue 36 pending interrupt
22	MSGMGR_QUE_PEND_48	Message Manager queue 48 pending interrupt
23	MSGMGR_QUE_PEND_56	Message Manager queue 56 pending interrupt
24	NSS_QPEND36	NSS transmit queue 36 pending interrupt
25	NSS_QPEND37	NSS transmit queue 37 pending interrupt
26	NSS_QPEND38	NSS transmit queue 38 pending interrupt
27	NSS_QPEND39	NSS transmit queue 39 pending interrupt
28	NSS_QPEND0	NSS transmit queue 0 pending interrupt
29	NSS_QPEND1	NSS transmit queue 1 pending interrupt
30	NSS_QPEND2	NSS transmit queue 2 pending interrupt
31	NSS_QPEND3	NSS transmit queue 3 pending interrupt
32	TIMER_0_INTL	TIMER_0 low interrupt
33	TIMER_0_INTH	TIMER_0 high interrupt
34	TIMER_1_INTL	TIMER_1 low interrupt
35	TIMER_1_INTH	TIMER_1 high interrupt
36	TIMER_2_INTL	TIMER_2 low interrupt
37	TIMER_2_INTH	TIMER_2 high interrupt
38	TIMER_3_INTL	TIMER_3 low interrupt
39	TIMER_3_INTH	TIMER_3 high interrupt
40	TIMER_4_INTL	TIMER_4 low interrupt
41	TIMER_4_INTH	TIMER_4 high interrupt
42	TIMER_5_INTL	TIMER_5 low interrupt
43	TIMER_5_INTH	TIMER_5 high interrupt
44	RESERVED	Reserved
45	RESERVED	Reserved
46	RESERVED	Reserved
47	RESERVED	Reserved
48	CIC_0_OUT32	CIC output 32 host interrupt
49	CIC_0_OUT33	CIC output 33 host interrupt
50	CIC_0_OUT34	CIC output 34 host interrupt
51	CIC_0_OUT35	CIC output 35 host interrupt
52	CIC_0_OUT36	CIC output 36 host interrupt
53	CIC_0_OUT37	CIC output 37 host interrupt
54	CIC_0_OUT38	CIC output 38 host interrupt
55	CIC_0_OUT39	CIC output 39 host interrupt
56	CIC_0_OUT40	CIC output 40 host interrupt
57	CIC_0_OUT41	CIC output 41 host interrupt
58	CIC_0_OUT42	CIC output 42 host interrupt
59	CIC_0_OUT43	CIC output 43 host interrupt
60	CIC_0_OUT44	CIC output 44 host interrupt
61	CIC_0_OUT45	CIC output 45 host interrupt
62	CIC_0_OUT46	CIC output 46 host interrupt
63	CIC_0_OUT47	CIC output 47 host interrupt
64	CIC_0_OUT48	CIC output 48 host interrupt

Table 6-37. C66x DSP Interrupt Sources (continued)

Event No.	Event Source Name	Description
65	CIC_0_OUT49	CIC output 49 host interrupt
66	CIC_0_OUT50	CIC output 50 host interrupt
67	CIC_0_OUT51	CIC output 51 host interrupt
68	CIC_0_OUT52	CIC output 52 host interrupt
69	CIC_0_OUT53	CIC output 53 host interrupt
70	CIC_0_OUT54	CIC output 54 host interrupt
71	CIC_0_OUT55	CIC output 55 host interrupt
72	CIC_0_OUT56	CIC output 56 host interrupt
73	CIC_0_OUT57	CIC output 57 host interrupt
74	CIC_0_OUT58	CIC output 58 host interrupt
75	CIC_0_OUT59	CIC output 59 host interrupt
76	CIC_0_OUT60	CIC output 60 host interrupt
77	CIC_0_OUT61	CIC output 61 host interrupt
78	CIC_0_OUT62	CIC output 62 host interrupt
79	CIC_0_OUT63	CIC output 63 host interrupt
80	CIC_0_OUT64	CIC output 64 host interrupt
81	CIC_0_OUT65	CIC output 65 host interrupt
82	CIC_0_OUT66	CIC output 66 host interrupt
83	CIC_0_OUT67	CIC output 67 host interrupt
84	CIC_0_OUT68	CIC output 68 host interrupt
85	CIC_0_OUT69	CIC output 69 host interrupt
86	CIC_0_OUT70	CIC output 70 host interrupt
87	CIC_0_OUT71	CIC output 71 host interrupt
88	CIC_0_OUT72	CIC output 72 host interrupt
89	CIC_0_OUT73	CIC output 73 host interrupt
90	CIC_0_OUT74	CIC output 74 host interrupt
91	CIC_0_OUT75	CIC output 75 host interrupt
92	CIC_0_OUT76	CIC output 76 host interrupt
93	CIC_0_OUT77	CIC output 77 host interrupt
94	CIC_0_OUT78	CIC output 78 host interrupt
95	CIC_0_OUT79	CIC output 79 host interrupt
96	INTERR	C66x CorePac internal event
97	EMC_IDMAERR	C66x CorePac internal event
98	PBISTINT	C66x CorePac internal event
99	RESERVED	Reserved
100	EFIINT0	C66x CorePac internal event
101	EFIINT1	C66x CorePac internal event
102	GPIOMUX_INT16	GPIOMUX16 output
103	GPIOMUX_INT17	GPIOMUX17 output
104	GPIOMUX_INT18	GPIOMUX18 output
105	GPIOMUX_INT19	GPIOMUX19 output
106	GPIOMUX_INT20	GPIOMUX20 output
107	GPIOMUX_INT21	GPIOMUX21 output
108	GPIOMUX_INT22	GPIOMUX22 output
109	GPIOMUX_INT23	GPIOMUX23 output
110	MDMAERREVT	C66x CorePac internal event
111	EDC_NCERR	C66x CorePac internal event

Table 6-37. C66x DSP Interrupt Sources (continued)

Event No.	Event Source Name	Description
112	EDC_CORRERR	C66x CorePac internal event
113	PMC_ED	C66x CorePac internal event
114	EDMACC_0_TC_AET_INT	EDMACC_0 advanced event trigger (AET) interrupt
115	EDMACC_1_TC_AET_INT	EDMACC_1 advanced event trigger (AET) interrupt
116	UMC_ED1	C66x CorePac internal event
117	UMC_ED2	C66x CorePac internal event
118	PDC_INT	C66x CorePac internal event
119	SYS_CMPA	C66x CorePac internal event
120	PMC_CMPA	C66x CorePac internal event
121	PMC_DMPA	C66x CorePac internal event
122	DMC_CMPA	C66x CorePac internal event
123	DMC_DMPA	C66x CorePac internal event
124	UMC_CMPA	C66x CorePac internal event
125	UMC_DMPA	C66x CorePac internal event
126	EMC_CMPA	C66x CorePac internal event
127	EMC_BUSERR	C66x CorePac internal event

6.2.3 DSP Subsystem Functional Description

For detailed DSP subsystem functional description, see the documents referenced in Section 6.2.1, *DSP Subsystem Overview*.

6.2.4 DSP Subsystem Registers

Table 6-38 lists the memory-mapped registers for the C66x CorePac. All register offset addresses not listed in Table 6-38 should be considered as reserved locations and the register contents should not be modified. For base addresses of the DSP subsystem internal modules, see Chapter 2, *Memory Map*.

The C66x CorePac registers are described in detail in the *TMS320C66x DSP CorePac User Guide* (SPRUGW0).

Table 6-38. C66x CorePac Registers

Offset	Acronym	Register Name
800000h to 80000Ch	EVTFLAG_0 to EVTFLAG_3	Event Flag Registers
800020h to 80002Ch	EVTSET_0 to EVTSET_3	Event Set Registers
800040h to 80004Ch	EVTCLR_0 to EVTCLR_3	Event Clear Registers
800080h to 80008Ch	EVTMASK_0 to EVTMASK_3	Event Mask Registers
8000A0h to 8000ACh	MEVTFLAG_0 to MEVTFLAG_3	Masked Event Flag Registers
8000C0h to 8000CCh	EXPMASK_0 to EXPMASK_3	Exception Mask Registers
8000E0h to 8000ECh	MEXPFLAG_0 to MEXPFLAG_3	Masked Exception Flag Registers
800104h	INTMUX1	Interrupt Mux Register
800108h	INTMUX2	Interrupt Mux Register
80010Ch	INTMUX3	Interrupt Mux Register
800140h	AEGMUX0	Advanced Event Generator Mux Register
800144h	AEGMUX1	Advanced Event Generator Mux Register
800180h	INTXSTAT	Interrupt Exception Status Register
800184h	INTXCLR	Interrupt Exception Clear Register
800188h	INTDMASK	Dropped Interrupt Mask Register
8001C0h	EVTASRT	Event Assert Register
810000h	PDCCMD	Power-Down Controller Command Register
811100h	EDCINTMASK	Error Detect and Correct Interrupt Mask Register
812000h	MM_REVID	C66x CorePac Revision ID Register
820000h	IDMA0_STAT	IDMA Channel 0 Status Register
820004h	IDMA0_MASK	IDMA Channel 0 Mask Register
820008h	IDMA0_SOURCE	IDMA Channel 0 Source Address Register
82000Ch	IDMA0_DEST	IDMA Channel 0 Destination Address Register
820010h	IDMA0_COUNT	IDMA Channel 0 Count Register
820100h	IDMA1_STAT	IDMA Channel 1 Status Register
820108h	IDMA1_SOURCE	IDMA Channel 1 Source Address Register
82010Ch	IDMA1_DEST	IDMA Channel 1 Destination Address Register
820110h	IDMA1_COUNT	IDMA Channel 1 Count Register
820200h	CPUARBE	EMC DSP Arbitration Control Register
820204h	IDMAARBE	EMC IDMA Arbitration Control Register
820208h	SDMAARBE	EMC Slave DMA Arbitration Control Register
820210h	ECFGARBE	EMC CFG Arbitration Control Register
820300h	ICFGMPFAR	CFG Memory Protection Fault Address Register
820304h	ICFGMPFSR	CFG Memory Protection Fault Status Register
820308h	ICFGMPFCR	CFG Memory Protection Fault Command Register
820408h	ECFGERR	CFG Bus Error Register
82040Ch	ECFGERRCLR	CFG Bus Error Clear Register
820500h	PAMAP0	PAMAP Register
820504h	PAMAP1	PAMAP Register
820508h	PAMAP2	PAMAP Register
82050Ch	PAMAP3	PAMAP Register

Table 6-38. C66x CorePac Registers (continued)

Offset	Acronym	Register Name
820510h	PAMAP4	PAMAP Register
820514h	PAMAP5	PAMAP Register
820518h	PAMAP6	PAMAP Register
82051Ch	PAMAP7	PAMAP Register
820520h	PAMAP8	PAMAP Register
820524h	PAMAP9	PAMAP Register
820528h	PAMAP10	PAMAP Register
82052Ch	PAMAP11	PAMAP Register
820530h	PAMAP12	PAMAP Register
820534h	PAMAP13	PAMAP Register
820538h	PAMAP14	PAMAP Register
82053Ch	PAMAP15	PAMAP Register
821104h	EDCINTFLG	Error Detect and Correct Interrupt Flag Register
821108h	L1DEDCMD	L1D Error Detect Command Register
82110Ch	L1DDCSTAT	L1D Error Detect DATA Correctable Status Register
821110h	L1DDNCSTAT	L1D Error Detect DATA Non-Correctable Status Register
821114h	L1DTCSTAT	L1D Error Detect TAG Correctable Status Register
821118h	L1DTNCSTAT	L1D Error Detect TAG Non-Correctable Status Register
82111Ch	L1DDEDADDR	L1D Error Detect Correctable and Non-Correctable DATA Address Register
821120h	L1DTEDADDR	L1D Error Detect Correctable and Non-Correctable TAG Address Register
821124h	L1DEDCNT	L1D EDC Count Register
821128h	L2TEDCMD	L2 Error Detect Command Regsiter
82112Ch	L2TCSTAT	L2 Error Detect TAG Correctable Status Register
821130h	L2TNCSTAT	L2 Error Detect TAG Non-Correctable Status Register
821134h	L2TEDADDR	L2 Error Detect TAG Correctable and Non-Correctable Address Register
821138h	L2MCSTAT	L2 Error Detect MPPA Correctable Status Register
82113Ch	L2MNCSTAT	L2 Error Detect MPPA Non-Correctable Status Register
821140h	L2MEDADDR	L2 Error Detect MPPA Correctable and Non-Correctable Address Register
821144h	L2SCSTAT	L2 Error Detect SNOP Correctable Stauts Register
821148h	L2SNCSTAT	L2 Error Detect SNOP Non-Correctable Stauts Register
82114Ch	L2SEDADDR	L2 Error Detect SNOP Correctable and Non-Correctable Address Register
821150h	L2LCSTAT	L2 Error Detect LRU Correctable Status Register
821154h	L2LNCSTAT	L2 Error Detect LRU Non-Correctable Status Register
821158h	L2LEDADDR	L2 Error Detect LRU Correctable and Non-Correctable Address Register
82115Ch	L2TEDCNT	L2 Error Detect Parity Error Count Register
821160h	L1PTEDCMD	L1P Error Detect TAG Command Register
821164h	L1PTEDSTAT	L1P Error Detect TAG Status Register
821168h	L1PTEDADDR	L1P Error Detect TAG Lower Address Register
82116Ch	L1DTEDCNT	L1P Error Detect TAG Parity Error Count Register
840000h	L2CFG	L2 Configuration Register
840020h	L1PCFG	L1P Configuration Register
840024h	L1PCC	L1P Cache Control Register
840040h	L1DCFG	L1D Cache Configuration Register
840044h	L1DCC	L1D Cache Control Register
841000h	CPUARBU	L2 DSP Arbitration Control Register
841004h	IDMAARBU	L2 IDMA Arbitration Control Register
841008h	SDMAARBU	L2 Slave DMA Arbitration Control Register

Table 6-38. C66x CorePac Registers (continued)

Offset	Acronym	Register Name
84100Ch	UCARBU	L2 User Coherence Arbitration Control Register
841010h	MDMAARBU	L2 Master DMA Arbitration Control Register
841040h	CPUARBD	L1 DSP Arbitration Control Register
841044h	IDMAARBD	L1 IDMA Arbitration Control Register
841048h	SDMAARBD	L1 Slave DMA Arbitration Control Register
84104Ch	UCARBD	L1 User Coherence Arbitration Control Register
844000h	L2WBAR	L2 Writeback Base Address Register
844004h	L2WWC	L2 Writeback Word Count Register
844010h	L2WIBAR	L2 Writeback-Invalidate Base Address Register
844014h	L2WIWC	L2 Writeback-Invalidate Word Count Register
844018h	L2IBAR	L2 Invalidate Base Address Register
84401Ch	L2IWC	L2 Invalidate Word Count Register
844020h	L1PIBAR	L1 Program Invalidate Base Address Register
844024h	L1PIWC	L1 Program Invalidate Word Count Register
844030h	L1DWIBAR	L1D Writeback-Invalidate Base Address Register
844034h	L1DWIWC	L1D Writeback-Invalidate Word Count Register
844040h	L1DWBAR	L1D Writeback Base Address Register
844044h	L1DWWC	L1D Writeback Word Count Register
844048h	L1DIBAR	L1D Invalidate Base Address Register
84404Ch	L1DIWC	L1D Invalidate Word Count Register
845000h	L2WB	L2 Writeback Register
845004h	L2WBINV	L2 Writeback-Invalidate Register
845008h	L2INV	L2 Invalidate Register
845028h	L1PINV	L1 Program Invalidate Register
845040h	L1DWB	L1D Writeback Register
845044h	L1DWBINV	L1D Writeback-Invalidate Register
845048h	L1DINV	L1D Invalidate Register
846004h	L2EDSTAT	L2 Error Detection Status Register
846008h	L2EDCMD	L2 Error Detection Command Register
84600Ch	L2EDADDR	L2 Error Detection Address Register
846018h	L2EDCPEC	L2 Error Detection Correctable Parity Error Counter Register
84601Ch	L2EDCNEC	L2 Error Detection Non-correctable Parity Error Counter Register
846020h	MDMAERR	MDMA Bus Error Register
846024h	MDMAERRCLR	MDMA Bus Error Clear Register
846030h	L2EDCEN	L2 Error Detection and Correction Enable Register
846404h	L1PEDSTAT	L1P Error Detection Status Register
846408h	L1PEDCMD	L1P Error Detection Command Register
84640Ch	L1PEDADDR	L1P Error Detection Address Register
848000h to 8483FCh	MAR_0 to MAR_255	Memory Attribute Registers
84A000h	L2MPFAR	L2 Memory Protection Fault Address Register
84A004h	L2MPFSR	L2 Memory Protection Fault Set Register
84A008h	L2MPFCR	L2 Memory Protection Fault Clear Register
84A200h to 84A27Ch	L2MPPA_0 to L2MPPA_31	L2 Memory Protection Page Attribute Registers
84A400h	L1PMPFAR	L1P Memory Protection Fault Address Register
84A404h	L1PMPFSR	L1P Memory Protection Fault Set Register
84A408h	L1PMPFCR	L1P Memory Protection Fault Clear Register
84A640h to 84A67Ch	L1PMPPA_0 to L1PMPPA_15	L1P Memory Protection Page Attribute Registers

Table 6-38. C66x CorePac Registers (continued)

Offset	Acronym	Register Name
84AC00h	L1DMPFAR	L1D Memory Protection Fault Address Register
84AC04h	L1DMPFSR	L1D Memory Protection Fault Set Register
84AC08h	L1DMPFCR	L1D Memory Protection Fault Clear Register
84AD00h to 84AD0Ch	MPLK_0 to MPLK_3	Memory Protection Lock Registers
84AD10h	MPLKCMD	Memory Protection Lock Command Register
84AD14h	MPLKSTAT	Memory Protection Lock Status Register
84AE40h to 84AE7Ch	L1DMPPA_0 to L1DMPPA_15	L1D Memory Page Protection Attribute Registers

6.3 C66x Cache Subsystem

The purpose of this section is to provide an overview of the C66x cache memory architecture and to specify its configuration in this device. Details on the C66x cache functionality can be found in the *TMS320C66x DSP Cache User Guide* (SPRUGY8).

The device contains a 1024KB level-2 memory (L2), a 32KB level-1 program memory (L1P), and a 32KB level-1 data memory (L1D). Each memory has a unique location in the memory map (see Chapter 2, *Memory Map*).

After device reset, L1P and L1D cache are configured as all cache, by default. The L1P and L1D cache can be reconfigured via software through the L1PMODE field of the L1P Configuration Register (L1PMODE) and the L1DMODE field of the L1D Configuration Register (L1DCFG) of the C66x CorePac. L1D is a two-way set-associative cache, while L1P is a direct-mapped cache.

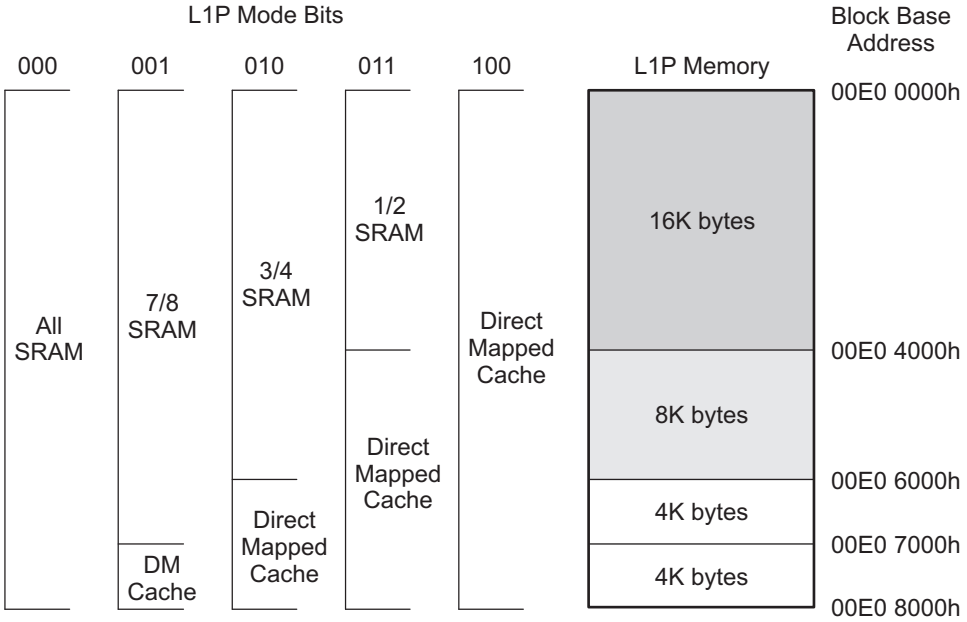
6.3.1 L1P Memory

The L1P memory configuration for this device is as follows:

- Region 0 size is 0K bytes (disabled)
- Region 1 size is 32K bytes with no wait states

Figure 6-14 shows the available SRAM/cache configurations for L1P.

Figure 6-14. L1P Memory Configurations



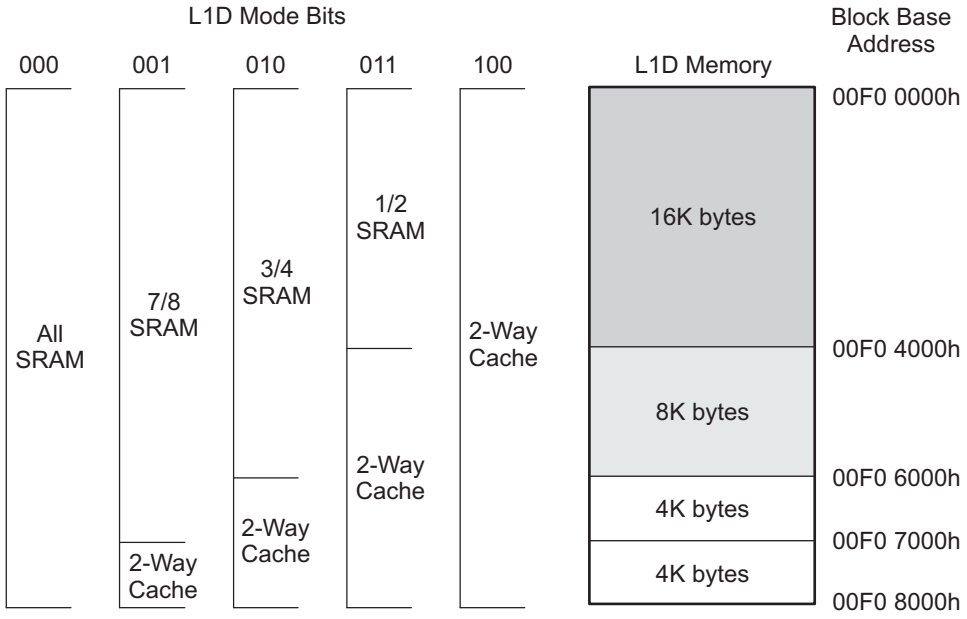
6.3.2 L1D Memory

The L1D memory configuration for this device is as follows:

- Region 0 size is 0K bytes (disabled)
- Region 1 size is 32K bytes with no wait states.

Figure 6-15 shows the available SRAM/cache configurations for L1D.

Figure 6-15. L1D Memory Configurations



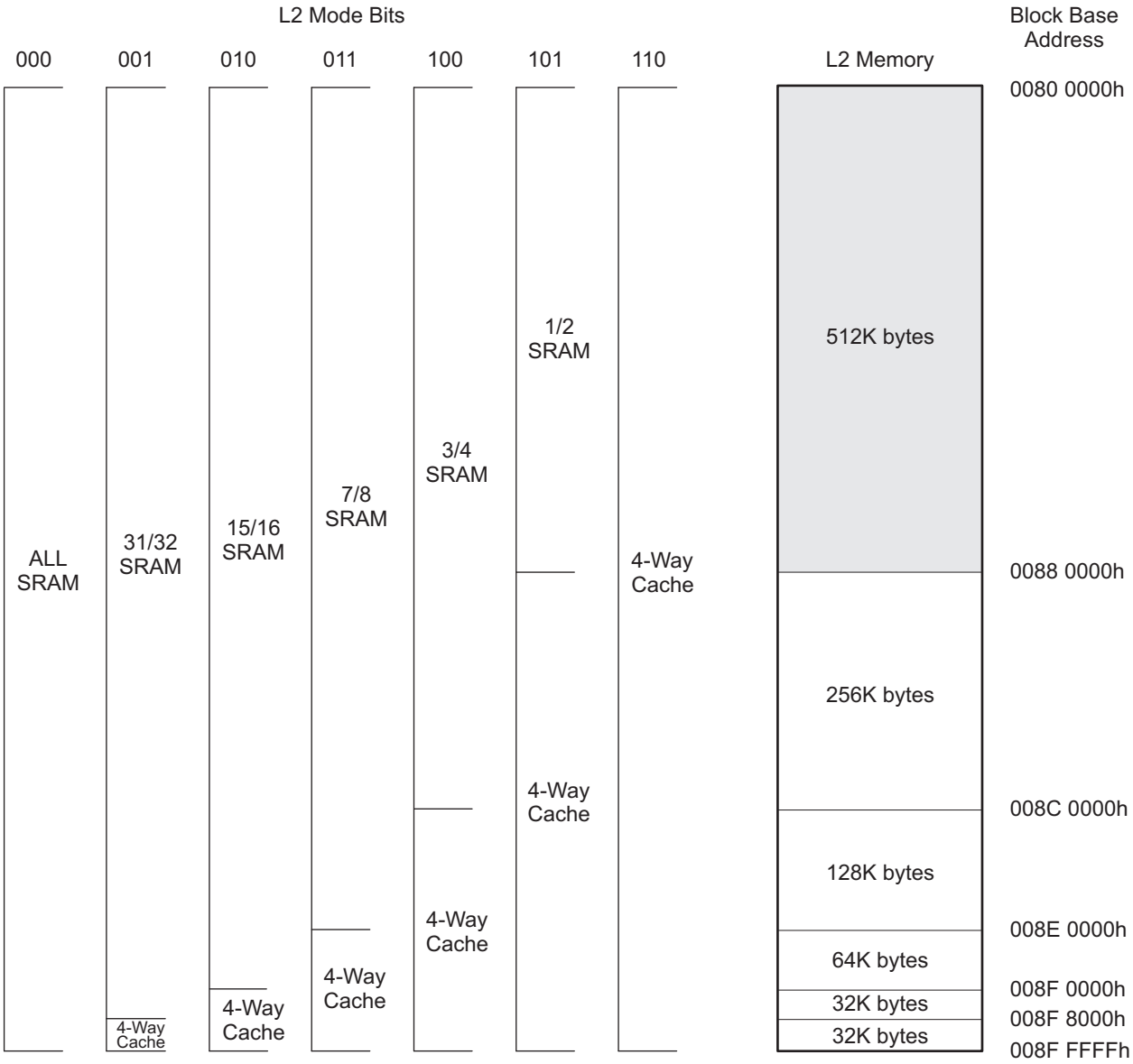
6.3.3 L2 Memory

The L2 memory configuration for this device is as follows:

- 1024KB of memory
- Local starting address is 0080 0000h.

L2 memory can be configured as all SRAM, all 4-way set-associative cache, or a mix of the two. The amount of L2 memory that is configured as cache is controlled through the L2MODE field of the L2 Configuration Register (L2CFG) of the C66x CorePac. Figure 6-16 shows the available SRAM/cache configurations for L2. By default, L2 is configured as all SRAM after device reset.

Figure 6-16. L2 Memory Configurations



6.3.4 DSP ECC configuration

The device supports ECC/parity on all DSP internal SRAMs. All the ECC/parity features are enabled by default (that is, after device reset). Software can disable the features per SRAM type if not required.

Table 6-39 summarizes the ECC/parity support in DSP.

Table 6-39. DSP ECC/Parity Support

DSP Memory	ECC/Parity Support
L1P Data RAM	Parity
L1P Tag	Parity
L1D Data RAM	ECC-SECDED (Single Error Correction, Double Error Detection)
L1D Tag	ECC-SECDED
L2 Data RAM	ECC-SECDED
L2 Tag	ECC-SECDED

6.4 Programmable Real-Time Unit Subsystem and Industrial Communication Subsystem (PRU-ICSS)

This section describes the Programmable Real-Time Unit and Industrial Communication Subsystems (PRU-ICSS) in this device.

NOTE: The supported set of features and peripherals is device part number dependent. For more information, see the device Data Manual.

6.4.1 PRU-ICSS Overview

The Programmable Real-Time Unit Subsystem and Industrial Communication Subsystem (PRU-ICSS) consists of:

- Two 32-bit load/store RISC CPU cores — Programmable Real-Time Units (PRU0 and PRU1)
- Data RAMs per PRU core
- Instruction RAMs per PRU core
- Shared RAM
- Peripheral modules
- Interrupt controller (ICSS_INTC).

The programmable nature of the PRU cores, along with their access to pins, events and all device resources, provides flexibility in implementing fast real-time responses, specialized data handling operations, custom peripheral interfaces, and in offloading tasks from the other processor cores of the device.

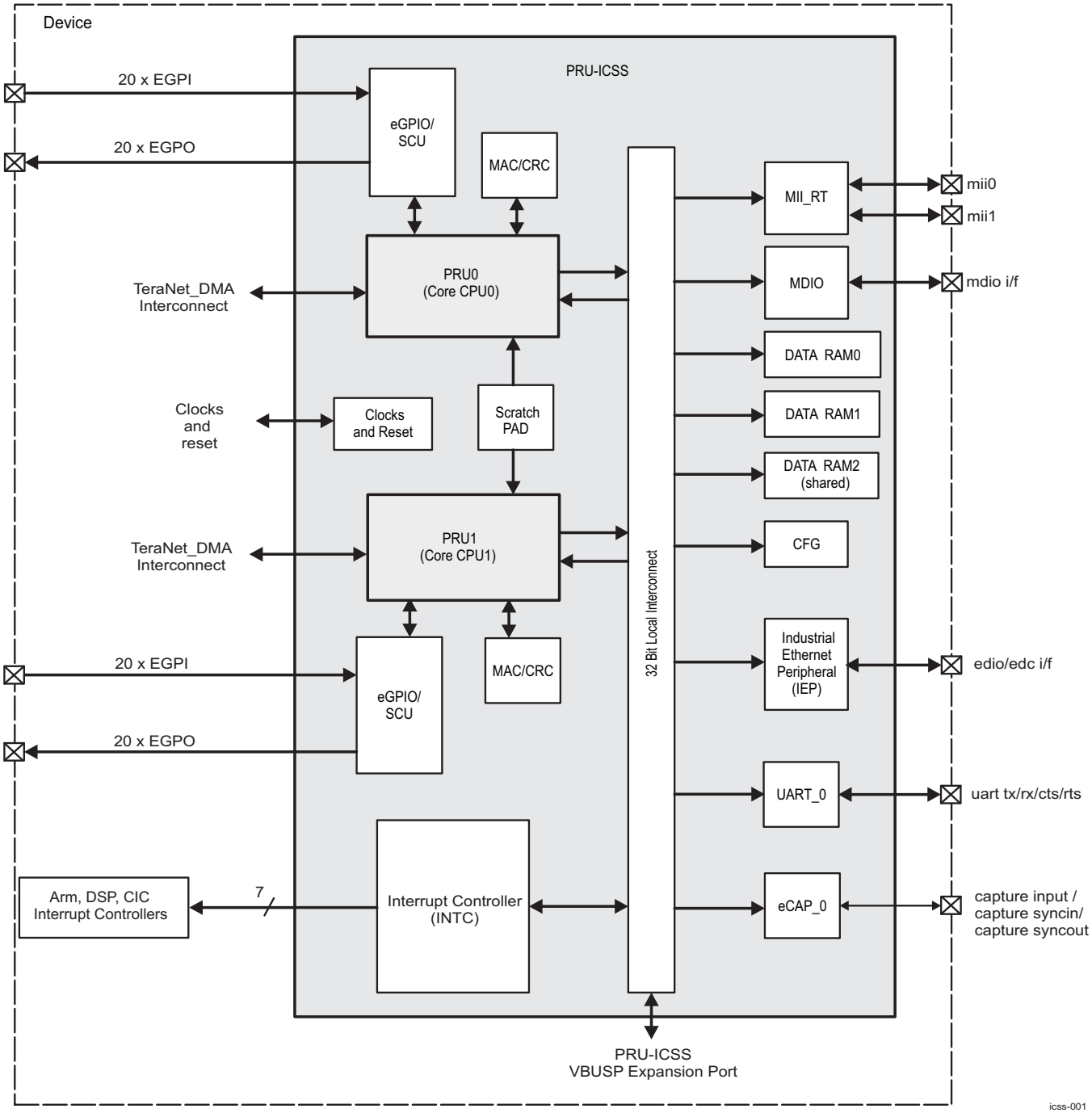
The device has integrated two identical PRU subsystems (PRU-ICSS_0 and PRU-ICSS_1).

The PRU cores within each PRU-ICSS have access to all resources on the SoC through the Interface Master port, and the external host processors can access the PRU-ICSS resources through the Interface Slave port. The 32-bit interconnect bus connects the various internal and external masters to the resources inside the PRU-ICSS. The PRU cores within the subsystems also have access to all resources on the SoC through the TeraNet DMA Interconnect. A subsystem local Interrupt Controller — ICSS_INTC handles system input events and posts events back to the device-level host CPUs.

The PRU cores are programmed with a small, deterministic instruction set. Each PRU can operate independently or in coordination with each other and can also work in coordination with the device-level host CPU. This interaction between processors is determined by the nature of the firmware loaded into the PRU's instruction memory.

Figure 6-17 shows an overview of the PRU subsystem.

Figure 6-17. PRU-ICSS Overview



6.4.1.1 PRU-ICSS Key Features

The PRU subsystem includes the following main features:

- Two PRU CPUs
  - 20 Enhanced General-Purpose Inputs (EGPI) and 20 Enhanced General-Purpose Outputs (EGPO)
  - Asynchronous capture [Serial Capture Unit (SCU)] with EnDat 2.2 protocol and Sigma-Delta demodulation support
    NOTE: There is no Sigma-Delta modulator inside the PRU. However, Sigma-Delta support is enabled through digital filtering hardware in the PRU to perform Sinc filtering.

  - Multiplier with accumulation (MAC)
  - CRC16/CRC32 HW accelerator
  - 16-KB program RAM per PRU CPU (signified IRAM0 for PRU0 and IRAM1 for PRU1) with ECC
  - 8-KB data RAM per PRU CPU (signified RAM0 for PRU0 and RAM1 for PRU1) with ECC
  - Two high-performance master (initiator) ports on the TeraNet_DMA interconnect — one per PRU
- 64-KB general purpose memory RAM (signified RAM2) with ECC, shared between PRU0 and PRU1
- One Scratch-Pad (SPAD) memory
  - 3 Banks of 30 × 32-bit registers
- Broadside direct connect between PRU cores within subsystem. Optional address translation for PRU transaction to External Host
- 16 software events generated by two PRUs
- One Ethernet MII_RT module (PRUSS_MII_RT_CFG) with two MII ports and configurable connections to PRUs
- One MDIO Port (PRUSS_MII_MDIO) to control external Ethernet PHY
- One Industrial Ethernet Peripheral (IEP) to manage/generate Industrial Ethernet functions
  - One Industrial Ethernet 64-bit timer with 9 capture and 16 compare events with slow and fast compensation
- 16550-compatible UART with a dedicated 192-MHz clock to support 12-Mbps PROFIBUS
- Enhanced Capture Module (eCAP_0)
- Interrupt Controller (ICSS_INTC)
  - Up to 64 input events supported
  - Supports up to to 10 interrupt channels
  - Generation of 10 Host interrupts: 2 Host interrupts to PRU0 and PRU1, 1 Host interrupt to PRU-ICSS_0 and PRU-ICSS_1, 7 Host interrupts exported from the ICSS for signaling the Arm interrupt controllers (pulse and level provided)
  - Each system event can be enabled and disabled
  - Each host event can be enabled and disabled
  - Hardware prioritization of events
- One 32-bit VBUSP slave (target) port for memory mapped register and internal memories access
- Two (master and slave) 32-bit VBUSP ports for low-latency interface between PRU-ICSS subsystems
- Flexible power management support
- Integrated 32-bit interconnect
- All memories support ECC

PRU-ICSS unsupported features:

- Only 4 bits are supported of the 32-bit Industrial Ethernet Digital Data Input
- Only 4 bits are supported of the 32-bit Industrial Ethernet Digital Data Output
- UART Modem interface is not supported
- 12 Enhanced General-Purpose Inputs (pr<0/1>_pru0_gpi[31:20]) and 12 Enhanced General-Purpose Outputs (pr<0/1>_pru0_gpo[31:20])

6.4.2 PRU-ICSS Environment

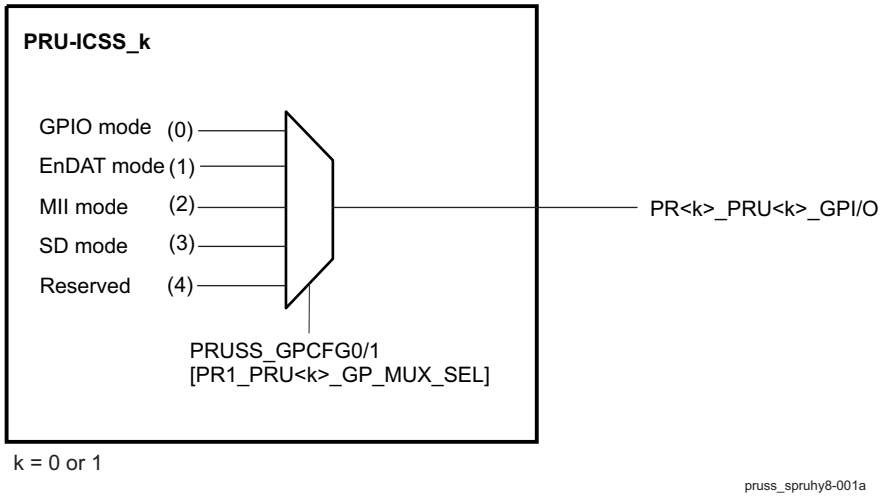
This section specifies the PRU-ICSS subsystem (top) interface signals to the device environment components.

6.4.2.1 PRU-ICSS I/O Interface

The PRU-ICSS_0 and PRU-ICSS_1 external interface signals are described in Table 6-40 and Table 6-41, respectively. The PRU-ICSS has a large number of available I/O signals. Most of these are multiplexed with other functional signals at the device level.

The PRU-ICSS_0 and PRU-ICSS_1 also support an internal wrapper multiplexing that expands the device top-level multiplexing. This wrapper multiplexing is controlled by the PRUSS_GPCFGx register (where x = 0 or 1) in the PRU-ICSS CFG register space and allows MII_RT, EnDAT, and Sigma Delta functionality to be muxed with the PRU GPI/O device signals, as shown in Figure 6-18. The PRU-ICSS wrapper multiplexing is described with the device-level signals in Table 6-40 and Table 6-41. Note that the device top-level muxing has higher priority over the internal wrapper muxing.

Figure 6-18. PRU-ICSS Internal Wrapper Multiplexing



NOTE: Additionally to PRU-ICSS wrapper multiplexing the device I/O logic maps the PRU-ICSS signals to the different device pads by programming in the BOOT_CFG Module. For more information, refer to theSection 5.1.3.1.1, *Pad Configuration Registers* in the Section 5.1, *BOOT_CFG*.

Table 6-40. PRU-ICSS_0 I/O Signals

Device Level Signal Name	Alternate Function via Internal Wrapper Multiplexing				I/O	Description	Reset
	PRUSS_GPCFG0[29-26] PR1_PRU0_GP_MUX_SEL=						0h - GPIO
PRU0 GP Signals	0h - GPIO mode (default)	1h - EnDAT mode	2h - MII mode	3h - SD mode			
PR0_PRU0_GPO0	pr0_pru0_pru_r30_out[0]	pr0_pru0_endat0_clk			O	PRU0 R30 Outputs/ Mux options	0
PR0_PRU0_GPO1	pr0_pru0_pru_r30_out[1]	pr0_pru0_endat0_out		pr0_pru0_pru_r30_out[1]	O	PRU0 R30 Outputs/ Mux options	0
PR0_PRU0_GPO2	pr0_pru0_pru_r30_out[2]	pr0_pru0_endat0_out_en			O	PRU0 R30 Outputs/ Mux options	0
PR0_PRU0_GPO3	pr0_pru0_pru_r30_out[3]	pr0_pru0_endat1_clk			O	PRU0 R30 Outputs/ Mux options	0
PR0_PRU0_GPO4	pr0_pru0_pru_r30_out[4]	pr0_pru0_endat1_out			O	PRU0 R30 Outputs/ Mux options	0
PR0_PRU0_GPO5	pr0_pru0_pru_r30_out[5]	pr0_pru0_endat1_out_en			O	PRU0 R30 Outputs/ Mux options	0
PR0_PRU0_GPO6	pr0_pru0_pru_r30_out[6]	pr0_pru0_endat2_clk			O	PRU0 R30 Outputs/ Mux options	0
PR0_PRU0_GPO7	pr0_pru0_pru_r30_out[7]	pr0_pru0_endat2_out			O	PRU0 R30 Outputs/ Mux options	0
PR0_PRU0_GPO8	pr0_pru0_pru_r30_out[8]	pr0_pru0_endat2_out_en			O	PRU0 R30 Outputs/ Mux options	0
PR0_PRU0_GPO9	pr0_pru0_pru_r30_out[9]				O	PRU0 R30 Outputs	0
PR0_PRU0_GPO10	pr0_pru0_pru_r30_out[10]				O	PRU0 R30 Outputs	0
PR0_PRU0_GPO11	pr0_pru0_pru_r30_out[11]		pr0_mii1_txd[0]		O	PRU0 R30 Outputs/ Mux options	0
PR0_PRU0_GPO12	pr0_pru0_pru_r30_out[12]		pr0_mii1_txd[1]		O	PRU0 R30 Outputs/ Mux options	0
PR0_PRU0_GPO13	pr0_pru0_pru_r30_out[13]		pr0_mii1_txd[2]		O	PRU0 R30 Outputs/ Mux options	0
PR0_PRU0_GPO14	pr0_pru0_pru_r30_out[14]		pr0_mii1_txd[3]		O	PRU0 R30 Outputs/ Mux options	0
PR0_PRU0_GPO15	pr0_pru0_pru_r30_out[15]		pr0_mii1_txen		O	PRU0 R30 Outputs/ Mux options	0
PR0_PRU0_GPO16	pr0_pru0_pru_r30_out[16]				O	PRU0 R30 Outputs	0
PR0_PRU0_GPO17	pr0_pru0_pru_r30_out[17]				O	PRU0 R30 Outputs	0
PR0_PRU0_GPO18	pr0_pru0_pru_r30_out[18]				O	PRU0 R30 Outputs	0
PR0_PRU0_GPO19	pr0_pru0_pru_r30_out[19]				O	PRU0 R30 Outputs	0
PR0_PRU0_GPI0	pr0_pru0_pru_r31_in[0]		pr0_mii0_rxd[0]	pr0_pru0_sd0_clk	I	PRU0 R31 Inputs/ Mux options	HiZ

Table 6-40. PRU-ICSS_0 I/O Signals (continued)

Device Level Signal Name	Alternate Function via Internal Wrapper Multiplexing				I/O	Description	Reset
PR0_PRU0_GPI1	pr0_pru0_pru_r31_in[1]		pr0_mii0_rxd[1]	pr0_pru0_sd0_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI2	pr0_pru0_pru_r31_in[2]		pr0_mii0_rxd[2]	pr0_pru0_sd1_clk	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI3	pr0_pru0_pru_r31_in[3]		pr0_mii0_rxd[3]	pr0_pru0_sd1_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI4	pr0_pru0_pru_r31_in[4]		pr0_mii0_rxdv	pr0_pru0_sd2_clk	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI5	pr0_pru0_pru_r31_in[5]		pr0_mii0_rxer	pr0_pru0_sd2_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI6	pr0_pru0_pru_r31_in[6]		pr0_mii_mr0_clk	pr0_pru0_sd3_clk	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI7	pr0_pru0_pru_r31_in[7]			pr0_pru0_sd3_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI8	pr0_pru0_pru_r31_in[8]		pr0_mii0_rxlink	pr0_pru0_sd4_clk	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI9	pr0_pru0_pru_r31_in[9]	pr0_pru0_endat0_in	pr0_mii0_col	pr0_pru0_sd4_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI10	pr0_pru0_pru_r31_in[10]	pr0_pru0_endat1_in	pr0_mii0_crs	pr0_pru0_sd5_clk	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI11	pr0_pru0_pru_r31_in[11]	pr0_pru0_endat2_in		pr0_pru0_sd5_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI12	pr0_pru0_pru_r31_in[12]			pr0_pru0_sd6_clk	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI13	pr0_pru0_pru_r31_in[13]			pr0_pru0_sd6_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI14	pr0_pru0_pru_r31_in[14]			pr0_pru0_sd7_clk	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI15	pr0_pru0_pru_r31_in[15]			pr0_pru0_sd7_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI16	pr0_pru0_pru_r31_in[16]	pr0_pru0_pru_r31_in[16]	pr0_mii_mt1_clk, pr0_pru0_pru_r31_in[16]	pr0_pru0_sd8_clk, pr0_pru0_pru_r31_in[16]	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI17	pr0_pru0_pru_r31_in[17]			pr0_pru0_sd8_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR0_PRU0_GPI18	pr0_pru0_pru_r31_in[18]				I	PRU0 R31 Inputs	HiZ
PR0_PRU0_GPI19	pr0_pru0_pru_r31_in[19]				I	PRU0 R31 Inputs	HiZ
PRU1 GP Signals	PRUSS_GPCFG1[29-26] PR1_PRU1_GP_MUX_SEL=						0h - GPIO
	0h - GPIO mode (default)	1h - EnDAT mode	2h - MII mode	3h - SD mode			

Table 6-40. PRU-ICSS_0 I/O Signals (continued)

Device Level Signal Name	Alternate Function via Internal Wrapper Multiplexing				I/O	Description	Reset
PR0_PRU1_GPO0	pr0_pru1_pru_r30_out[0]	pr0_pru1_endat0_clk			O	PRU1 R30 Outputs/ Mux options	0
PR0_PRU1_GPO1	pr0_pru1_pru_r30_out[1]	pr0_pru1_endat0_out		pr0_pru1_pru_r30_out[1]	O	PRU1 R30 Outputs/ Mux options	0
PR0_PRU1_GPO2	pr0_pru1_pru_r30_out[2]	pr0_pru1_endat0_out_en			O	PRU1 R30 Outputs/ Mux options	0
PR0_PRU1_GPO3	pr0_pru1_pru_r30_out[3]	pr0_pru1_endat1_clk			O	PRU1 R30 Outputs/ Mux options	0
PR0_PRU1_GPO4	pr0_pru1_pru_r30_out[4]	pr0_pru1_endat1_out			O	PRU1 R30 Outputs/ Mux options	0
PR0_PRU1_GPO5	pr0_pru1_pru_r30_out[5]	pr0_pru1_endat1_out_en			O	PRU1 R30 Outputs/ Mux options	0
PR0_PRU1_GPO6	pr0_pru1_pru_r30_out[6]	pr0_pru1_endat2_clk			O	PRU1 R30 Outputs/ Mux options	0
PR0_PRU1_GPO7	pr0_pru1_pru_r30_out[7]	pr0_pru1_endat2_out			O	PRU1 R30 Outputs/ Mux options	0
PR0_PRU1_GPO8	pr0_pru1_pru_r30_out[8]	pr0_pru1_endat2_out_en			O	PRU1 R30 Outputs/ Mux options	0
PR0_PRU1_GPO9	pr0_pru1_pru_r30_out[9]				O	PRU1 R30 Outputs/ Mux options	0
PR0_PRU1_GPO10	pr0_pru1_pru_r30_out[10]				O	PRU1 R30 Outputs/ Mux options	0
PR0_PRU1_GPO11	pr0_pru1_pru_r30_out[11]		pr0_mii0_txd[0]		O	PRU1 R30 Outputs/ Mux options	0
PR0_PRU1_GPO12	pr0_pru1_pru_r30_out[12]		pr0_mii0_txd[1]		O	PRU1 R30 Outputs/ Mux options	0
PR0_PRU1_GPO13	pr0_pru1_pru_r30_out[13]		pr0_mii0_txd[2]		O	PRU1 R30 Outputs/ Mux options	0
PR0_PRU1_GPO14	pr0_pru1_pru_r30_out[14]		pr0_mii0_txd[3]		O	PRU1 R30 Outputs/ Mux options	0
PR0_PRU1_GPO15	pr0_pru1_pru_r30_out[15]		pr0_mii0_txen		O	PRU1 R30 Outputs/ Mux options	0
PR0_PRU1_GPO16	pr0_pru1_pru_r30_out[16]				O	PRU1 R30 Outputs	0
PR0_PRU1_GPO17	pr0_pru1_pru_r30_out[17]				O	PRU1 R30 Outputs	0
PR0_PRU1_GPO18	pr0_pru1_pru_r30_out[18]				O	PRU1 R30 Outputs	0
PR0_PRU1_GPO19	pr0_pru1_pru_r30_out[19]				O	PRU1 R30 Outputs	0
PR0_PRU1_GPI0	pr0_pru1_pru_r31_in[0]		pr0_mii1_rxd[0]	pr0_pru1_sd0_clk	I	PRU1 R31 Inputs/ Mux options	HiZ

Table 6-40. PRU-ICSS_0 I/O Signals (continued)

Device Level Signal Name	Alternate Function via Internal Wrapper Multiplexing				I/O	Description	Reset
PR0_PRU1_GPI1	pr0_pru1_pru_r31_in[1]		pr0_mii1_rxd[1]	pr0_pru1_sd0_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI2	pr0_pru1_pru_r31_in[2]		pr0_mii1_rxd[2]	pr0_pru1_sd1_clk	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI3	pr0_pru1_pru_r31_in[3]		pr0_mii1_rxd[3]	pr0_pru1_sd1_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI4	pr0_pru1_pru_r31_in[4]		pr0_mii1_rxdv	pr0_pru1_sd2_clk	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI5	pr0_pru1_pru_r31_in[5]		pr0_mii1_rxer	pr0_pru1_sd2_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI6	pr0_pru1_pru_r31_in[6]		pr0_mii_mr1_clk	pr0_pru1_sd3_clk	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI7	pr0_pru1_pru_r31_in[7]			pr0_pru1_sd3_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI8	pr0_pru1_pru_r31_in[8]		pr0_mii1_rxlink	pr0_pru1_sd4_clk	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI9	pr0_pru1_pru_r31_in[9]	pr0_pru1_endat0_in	pr0_mii1_col	pr0_pru1_sd4_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI10	pr0_pru1_pru_r31_in[10]	pr0_pru1_endat1_in	pr0_mii1_crs	pr0_pru1_sd5_clk	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI11	pr0_pru1_pru_r31_in[11]	pr0_pru1_endat2_in		pr0_pru1_sd5_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI12	pr0_pru1_pru_r31_in[12]			pr0_pru1_sd6_clk	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI13	pr0_pru1_pru_r31_in[13]			pr0_pru1_sd6_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI14	pr0_pru1_pru_r31_in[14]			pr0_pru1_sd7_clk	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI15	pr0_pru1_pru_r31_in[15]			pr0_pru1_sd7_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI16	pr0_pru1_pru_r31_in[16]	pr0_pru1_pru_r31_in[16]	pr0_mii_mt0_clk, pr0_pru1_pru_r31_in[16]	pr0_pru1_sd8_clk, pr0_pru1_pru_r31_in[16]	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI17	pr0_pru1_pru_r31_in[17]			pr0_pru1_sd8_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR0_PRU1_GPI18	pr0_pru1_pru_r31_in[18]				I	PRU1 R31 Inputs	HiZ
PR0_PRU1_GPI19	pr0_pru1_pru_r31_in[19]				I	PRU1 R31 Inputs	HiZ
MDIO	MDIO						
PR0_MDIO_MDCLK	pr0_mdio_mdclk				O	MDIO Clock	0
PR0_MDIO_DATA	pr0_mdio_data				I/O	MDIO Data	HiZ

Table 6-40. PRU-ICSS_0 I/O Signals (continued)

Device Level Signal Name	Alternate Function via Internal Wrapper Multiplexing	I/O	Description	Reset
Industrial Ethernet	Industrial Ethernet			
PR0_EDIO_OUTVALID	pr0_edio_outvalid	O	IEP Digital I/O Output Valid	0
PR0_EDIO_DATA[0:3]	pr0_edio_data[0:3]	I/O	IEP Digital I/Os Data In	HiZ
PR0_EDC_SYNC0_OUT	pr0_edc_sync0_out	O	IEP Distributed Clock Sync Out	0
PR0_EDC_SYNC1_OUT	pr0_edc_sync1_out	O	IEP Distributed Clock Sync Out	0
PR0_EDC_LATCH0_IN	pr0_edc_latch0_in	I	IEP Distributed Clock Latch In	HiZ
PR0_EDC_LATCH1_IN	pr0_edc_latch1_in	I	IEP Distributed Clock Latch In	HiZ
UART	UART			
PR0_UART0_CTSN	pr0_uart0_ctsn	I	UART Clear to Send	HiZ
PR0_UART0_RTSN	pr0_uart0_rtsn	O	UART Request to Send	0
PR0_UART0_RXD	pr0_uart0_rxd	I	UART Receive Data	HiZ
PR0_UART0_TXD	pr0_uart0_txd	O	UART Transmit Data	0
ECAP	ECAP			
PR0_eCAP0_eCAP_CAPIN_APWM_O	pr0_ecap0_ecap_capin_apwm_o	I/O	Enhanced capture (ECAP) input or Auxiliary PWM out	HiZ
PR0_eCAP0_eCAP_SYNCIN	pr0_ecap0_ecap_syncin	I	Enhanced capture (ECAP) Sync In	0
PR0_eCAP0_eCAP_SYNCOUT	pr0_ecap0_ecap_syncout	O	Enhanced capture (ECAP) Sync Out	0

Table 6-41. PRU-ICSS_1 I/O Signals

Device Level Signal Name	Alternate Function via Internal Wrapper Multiplexing				I/O	Description	Reset
PRU0 GP Signals	PRUSS_GPCFG0[29-26] PR1_PRU0_GP_MUX_SEL=						0h - GPIO
	0h - GPIO mode (default)	1h - EnDAT mode	2h - MII mode	3h - SD mode			
PR1_PRU0_GPO0	pr1_pru0_pru_r30_out[0]	pr1_pru0_endat0_clk			O	PRU0 R30 Outputs/ Mux options	0
PR1_PRU0_GPO1	pr1_pru0_pru_r30_out[1]	pr1_pru0_endat0_out		pr1_pru0_pru_r30_out[1]	O	PRU0 R30 Outputs/ Mux options	0
PR1_PRU0_GPO2	pr1_pru0_pru_r30_out[2]	pr1_pru0_endat0_out_en			O	PRU0 R30 Outputs/ Mux options	0

Table 6-41. PRU-ICSS_1 I/O Signals (continued)

Device Level Signal Name	Alternate Function via Internal Wrapper Multiplexing			I/O	Description	Reset
PR1_PRU0_GPO3	pr1_pru0_pru_r30_out[3]	pr1_pru0_endat1_clk		O	PRU0 R30 Outputs/ Mux options	0
PR1_PRU0_GPO4	pr1_pru0_pru_r30_out[4]	pr1_pru0_endat1_out		O	PRU0 R30 Outputs/ Mux options	0
PR1_PRU0_GPO5	pr1_pru0_pru_r30_out[5]	pr1_pru0_endat1_out_en		O	PRU0 R30 Outputs/ Mux options	0
PR1_PRU0_GPO6	pr1_pru0_pru_r30_out[6]	pr1_pru0_endat2_clk		O	PRU0 R30 Outputs/ Mux options	0
PR1_PRU0_GPO7	pr1_pru0_pru_r30_out[7]	pr1_pru0_endat2_out		O	PRU0 R30 Outputs/ Mux options	0
PR1_PRU0_GPO8	pr1_pru0_pru_r30_out[8]	pr1_pru0_endat2_out_en		O	PRU0 R30 Outputs/ Mux options	0
PR1_PRU0_GPO9	pr1_pru0_pru_r30_out[9]			O	PRU0 R30 Outputs	0
PR1_PRU0_GPO10	pr1_pru0_pru_r30_out[10]			O	PRU0 R30 Outputs	0
PR1_PRU0_GPO11	pr1_pru0_pru_r30_out[11]	pr1_mii1_txd[0]		O	PRU0 R30 Outputs/ Mux options	0
PR1_PRU0_GPO12	pr1_pru0_pru_r30_out[12]	pr1_mii1_txd[1]		O	PRU0 R30 Outputs/ Mux options	0
PR1_PRU0_GPO13	pr1_pru0_pru_r30_out[13]	pr1_mii1_txd[2]		O	PRU0 R30 Outputs/ Mux options	0
PR1_PRU0_GPO14	pr1_pru0_pru_r30_out[14]	pr1_mii1_txd[3]		O	PRU0 R30 Outputs/ Mux options	0
PR1_PRU0_GPO15	pr1_pru0_pru_r30_out[15]	pr1_mii1_txen		O	PRU0 R30 Outputs/ Mux options	0
PR1_PRU0_GPO16	pr1_pru0_pru_r30_out[16]			O	PRU0 R30 Outputs	0
PR1_PRU0_GPO17	pr1_pru0_pru_r30_out[17]			O	PRU0 R30 Outputs	0
PR1_PRU0_GPO18	pr1_pru0_pru_r30_out[18]			O	PRU0 R30 Outputs	0
PR1_PRU0_GPO19	pr1_pru0_pru_r30_out[19]			O	PRU0 R30 Outputs	0
PR1_PRU0_GPI0	pr1_pru0_pru_r31_in[0]	pr1_mii0_rxd[0]	pr1_pru0_sd0_clk	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI1	pr1_pru0_pru_r31_in[1]	pr1_mii0_rxd[1]	pr1_pru0_sd0_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI2	pr1_pru0_pru_r31_in[2]	pr1_mii0_rxd[2]	pr1_pru0_sd1_clk	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI3	pr1_pru0_pru_r31_in[3]	pr1_mii0_rxd[3]	pr1_pru0_sd1_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI4	pr1_pru0_pru_r31_in[4]	pr1_mii0_rxdv	pr1_pru0_sd2_clk	I	PRU0 R31 Inputs/ Mux options	HiZ

Table 6-41. PRU-ICSS_1 I/O Signals (continued)

Device Level Signal Name	Alternate Function via Internal Wrapper Multiplexing				I/O	Description	Reset
PR1_PRU0_GPI5	pr1_pru0_pru_r31_in[5]		pr1_mii0_rxer	pr1_pru0_sd2_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI6	pr1_pru0_pru_r31_in[6]		pr1_mii_mr0_clk	pr1_pru0_sd3_clk	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI7	pr1_pru0_pru_r31_in[7]			pr1_pru0_sd3_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI8	pr1_pru0_pru_r31_in[8]		pr1_mii0_rxlink	pr1_pru0_sd4_clk	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI9	pr1_pru0_pru_r31_in[9]	pr1_pru0_endat0_in	pr1_mii0_col	pr1_pru0_sd4_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI10	pr1_pru0_pru_r31_in[10]	pr1_pru0_endat1_in	pr1_mii0_crs	pr1_pru0_sd5_clk	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI11	pr1_pru0_pru_r31_in[11]	pr1_pru0_endat2_in		pr1_pru0_sd5_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI12	pr1_pru0_pru_r31_in[12]			pr1_pru0_sd6_clk	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI13	pr1_pru0_pru_r31_in[13]			pr1_pru0_sd6_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI14	pr1_pru0_pru_r31_in[14]			pr1_pru0_sd7_clk	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI15	pr1_pru0_pru_r31_in[15]			pr1_pru0_sd7_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI16	pr1_pru0_pru_r31_in[16]	pr1_pru0_pru_r31_in[16]	pr1_mii_mt1_clk, pr1_pru0_pru_r31_in[16]	pr1_pru0_sd8_clk, pr1_pru0_pru_r31_in[16]	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI17	pr1_pru0_pru_r31_in[17]			pr1_pru0_sd8_d	I	PRU0 R31 Inputs/ Mux options	HiZ
PR1_PRU0_GPI18	pr1_pru0_pru_r31_in[18]				I	PRU0 R31 Inputs	HiZ
PR1_PRU0_GPI19	pr1_pru0_pru_r31_in[19]				I	PRU0 R31 Inputs	HiZ
PRU1 GP Signals	PRUSS_GPCFG1[29-26] PR1_PRU1_GP_MUX_SEL=						0h - GPIO
	0h - GPIO mode (default)	1h - EnDAT mode	2h - MII mode	3h - SD mode			
PR1_PRU1_GPO0	pr1_pru1_pru_r30_out[0]	pr1_pru1_endat0_clk			O	PRU1 R30 Outputs/ Mux options	0
PR1_PRU1_GPO1	pr1_pru1_pru_r30_out[1]	pr1_pru1_endat0_out		pr1_pru1_pru_r30_out[1]	O	PRU1 R30 Outputs/ Mux options	0
PR1_PRU1_GPO2	pr1_pru1_pru_r30_out[2]	pr1_pru1_endat0_out_en			O	PRU1 R30 Outputs/ Mux options	0
PR1_PRU1_GPO3	pr1_pru1_pru_r30_out[3]	pr1_pru1_endat1_clk			O	PRU1 R30 Outputs/ Mux options	0

Table 6-41. PRU-ICSS_1 I/O Signals (continued)

Device Level Signal Name	Alternate Function via Internal Wrapper Multiplexing			I/O	Description	Reset
PR1_PRU1_GPO4	pr1_pru1_pru_r30_out[4]	pr1_pru1_endat1_out		O	PRU1 R30 Outputs/ Mux options	0
PR1_PRU1_GPO5	pr1_pru1_pru_r30_out[5]	pr1_pru1_endat1_out_en		O	PRU1 R30 Outputs/ Mux options	0
PR1_PRU1_GPO6	pr1_pru1_pru_r30_out[6]	pr1_pru1_endat2_clk		O	PRU1 R30 Outputs/ Mux options	0
PR1_PRU1_GPO7	pr1_pru1_pru_r30_out[7]	pr1_pru1_endat2_out		O	PRU1 R30 Outputs/ Mux options	0
PR1_PRU1_GPO8	pr1_pru1_pru_r30_out[8]	pr1_pru1_endat2_out_en		O	PRU1 R30 Outputs/ Mux options	0
PR1_PRU1_GPO9	pr1_pru1_pru_r30_out[9]			O	PRU1 R30 Outputs	0
PR1_PRU1_GPO10	pr1_pru1_pru_r30_out[10]			O	PRU1 R30 Outputs	0
PR1_PRU1_GPO11	pr1_pru1_pru_r30_out[11]	pr1_mii0_txd[0]		O	PRU1 R30 Outputs/ Mux options	0
PR1_PRU1_GPO12	pr1_pru1_pru_r30_out[12]	pr1_mii0_txd[1]		O	PRU1 R30 Outputs/ Mux options	0
PR1_PRU1_GPO13	pr1_pru1_pru_r30_out[13]	pr1_mii0_txd[2]		O	PRU1 R30 Outputs/ Mux options	0
PR1_PRU1_GPO14	pr1_pru1_pru_r30_out[14]	pr1_mii0_txd[3]		O	PRU1 R30 Outputs/ Mux options	0
PR1_PRU1_GPO15	pr1_pru1_pru_r30_out[15]	pr1_mii0_txen		O	PRU1 R30 Outputs/ Mux options	0
PR1_PRU1_GPO16	pr1_pru1_pru_r30_out[16]			O	PRU1 R30 Outputs	0
PR1_PRU1_GPO17	pr1_pru1_pru_r30_out[17]			O	PRU1 R30 Outputs	0
PR1_PRU1_GPO18	pr1_pru1_pru_r30_out[18]			O	PRU1 R30 Outputs	0
PR1_PRU1_GPO19	pr1_pru1_pru_r30_out[19]			O	PRU1 R30 Outputs	0
PR1_PRU1_GPI0	pr1_pru1_pru_r31_in[0]	pr1_mii1_rxd[0]	pr1_pru1_sd0_clk	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI1	pr1_pru1_pru_r31_in[1]	pr1_mii1_rxd[1]	pr1_pru1_sd0_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI2	pr1_pru1_pru_r31_in[2]	pr1_mii1_rxd[2]	pr1_pru1_sd1_clk	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI3	pr1_pru1_pru_r31_in[3]	pr1_mii1_rxd[3]	pr1_pru1_sd1_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI4	pr1_pru1_pru_r31_in[4]	pr1_mii1_rxdv	pr1_pru1_sd2_clk	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI5	pr1_pru1_pru_r31_in[5]	pr1_mii1_rxer	pr1_pru1_sd2_d	I	PRU1 R31 Inputs/ Mux options	HiZ

Table 6-41. PRU-ICSS_1 I/O Signals (continued)

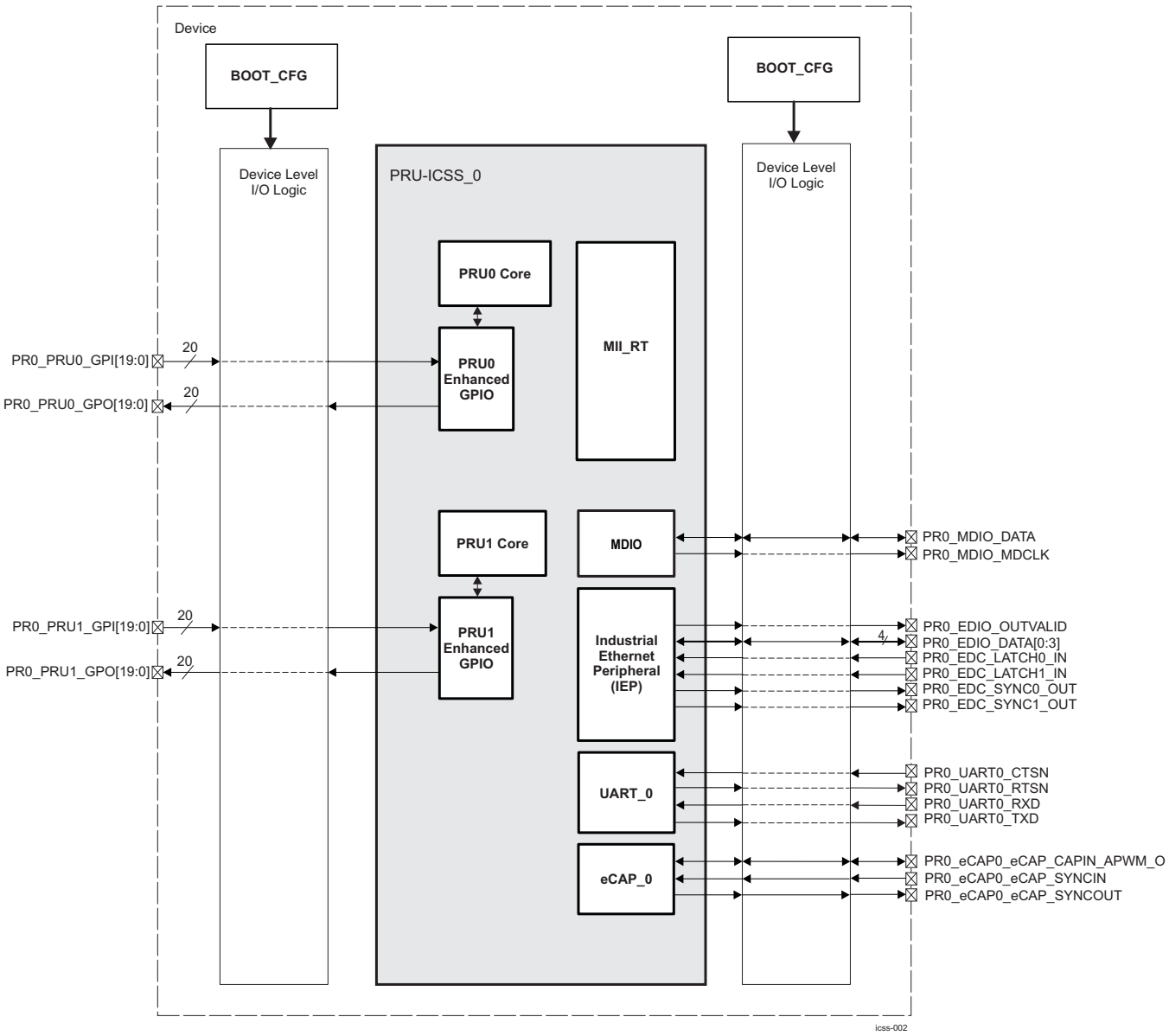
Device Level Signal Name	Alternate Function via Internal Wrapper Multiplexing				I/O	Description	Reset
PR1_PRU1_GPI6	pr1_pru1_pru_r31_in[6]		pr1_mii_mr1_clk	pr1_pru1_sd3_clk	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI7	pr1_pru1_pru_r31_in[7]			pr1_pru1_sd3_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI8	pr1_pru1_pru_r31_in[8]		pr1_mii1_rxlink	pr1_pru1_sd4_clk	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI9	pr1_pru1_pru_r31_in[9]	pr1_pru1_endat0_in	pr1_mii1_col	pr1_pru1_sd4_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI10	pr1_pru1_pru_r31_in[10]	pr1_pru1_endat1_in	pr1_mii1_crs	pr1_pru1_sd5_clk	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI11	pr1_pru1_pru_r31_in[11]	pr1_pru1_endat2_in		pr1_pru1_sd5_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI12	pr1_pru1_pru_r31_in[12]			pr1_pru1_sd6_clk	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI13	pr1_pru1_pru_r31_in[13]			pr1_pru1_sd6_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI14	pr1_pru1_pru_r31_in[14]			pr1_pru1_sd7_clk	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI15	pr1_pru1_pru_r31_in[15]			pr1_pru1_sd7_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI16	pr1_pru1_pru_r31_in[16]	pr1_pru1_pru_r31_in[16]	pr1_mii_mt0_clk, pr1_pru1_pru_r31_in[16]	pr1_pru1_sd8_clk, pr1_pru1_pru_r31_in[16]	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI17	pr1_pru1_pru_r31_in[17]			pr1_pru1_sd8_d	I	PRU1 R31 Inputs/ Mux options	HiZ
PR1_PRU1_GPI18	pr1_pru1_pru_r31_in[18]				I	PRU1 R31 Inputs	HiZ
PR1_PRU1_GPI19	pr1_pru1_pru_r31_in[19]				I	PRU1 R31 Inputs	HiZ
MDIO	MDIO						
PR1_MDIO_MDCLK	pr1_mdio_mdclk				O	MDIO Clock	0
PR1_MDIO_DATA	pr1_mdio_data				I/O	MDIO Data	HiZ
Industrial Ethernet	Industrial Ethernet						
PR1_EDIO_OUTVALID	pr1_edio_outvalid				O	IEP Digital I/O Output Valid	0
PR1_EDIO_DATA[0:3]	pr1_edio_data[0:3]				I/O	IEP Digital I/Os Data In	HiZ
PR1_EDC_SYNC0_OUT	pr1_edc_sync0_out				O	IEP Distributed Clock Sync Out	0
PR1_EDC_SYNC1_OUT	pr1_edc_sync1_out				O	IEP Distributed Clock Sync Out	0

Table 6-41. PRU-ICSS_1 I/O Signals (continued)

Device Level Signal Name	Alternate Function via Internal Wrapper Multiplexing	I/O	Description	Reset
PR1_EDC_LATCH0_IN	pr1_edc_latch0_in	I	IEP Distributed Clock Latch In	HiZ
PR1_EDC_LATCH1_IN	pr1_edc_latch1_in	I	IEP Distributed Clock Latch In	HiZ
UART	UART			
PR1_UART0_CTSN	pr1_uart0_ctsn	I	UART Clear to Send	HiZ
PR1_UART0_RTSN	pr1_uart0_rtsn	O	UART Request to Send	0
PR1_UART0_RXD	pr1_uart0_rxd	I	UART Receive Data	HiZ
PR1_UART0_TXD	pr1_uart0_txd	O	UART Transmit Data	0
ECAP	ECAP			
PR1_eCAP0_eCAP_CAPIN_APWM_O	pr1_ecap0_ecap_capin_apwm_o	I/O	Enhanced capture (ECAP) input or Auxiliary PWM out	HiZ
PR1_eCAP0_eCAP_SYNCIN	pr1_ecap0_ecap_syncin	I	Enhanced capture (ECAP) Sync In	0
PR1_eCAP0_eCAP_SYNCOUT	pr1_ecap0_ecap_syncout	O	Enhanced capture (ECAP) Sync Out	0

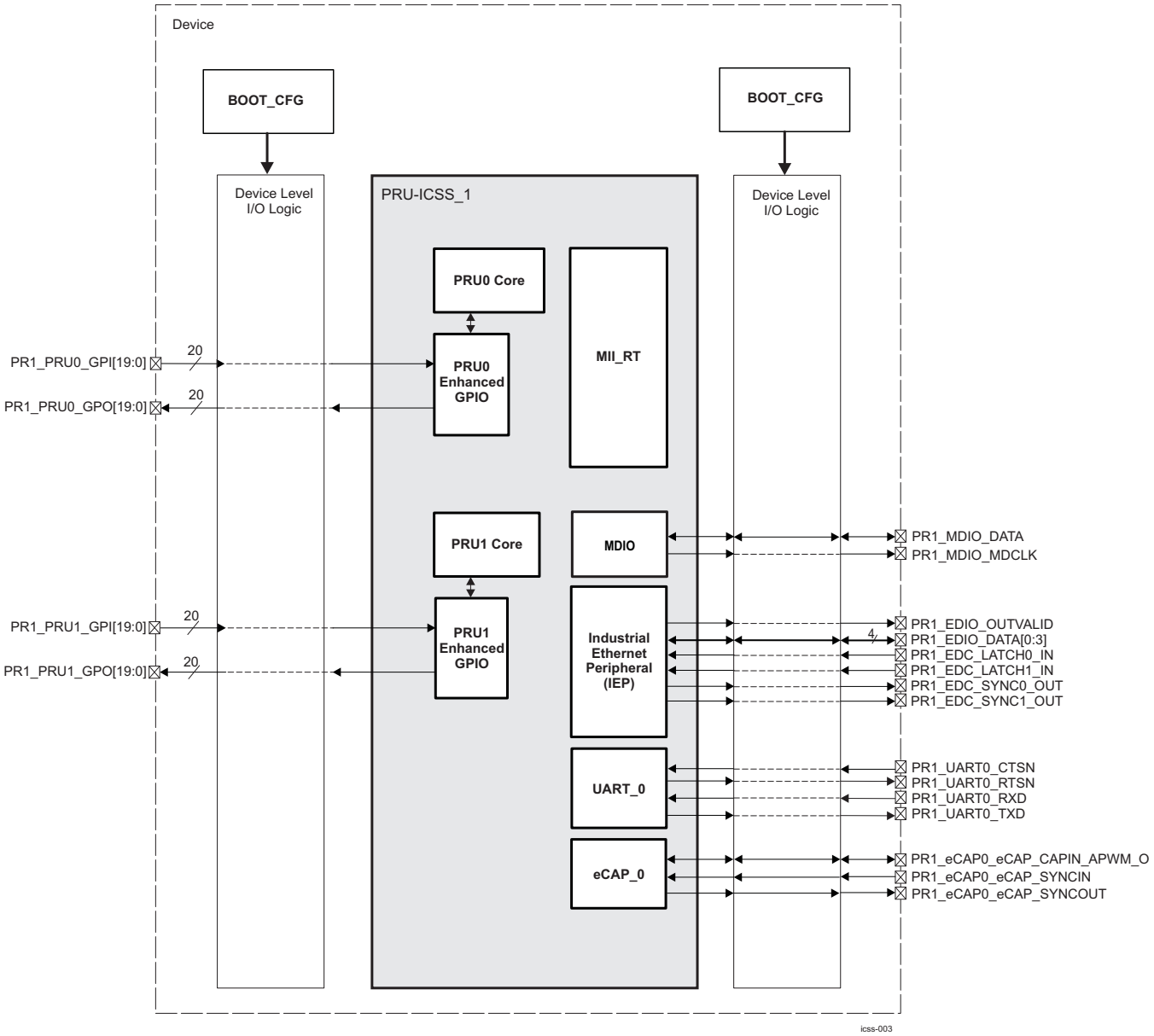
The below Figure 6-19 illustrates the PRU-ICSS_0 I/O interface signals at the device boundary.

Figure 6-19. PRU-ICSS_0 External Interface I/Os



The Figure 6-20 illustrates the PRU-ICSS_1 I/O interface signals at the device boundary.

Figure 6-20. PRU-ICSS_1 External Interface I/Os



6.4.3 PRU-ICSS Integration

The PRU-ICSS_0 and PRU-ICSS_1 subsystems integration in the device is shown in Figure 6-21 and Figure 6-22, respectively.

Figure 6-21. PRU-ICSS_0 Integration in the Device

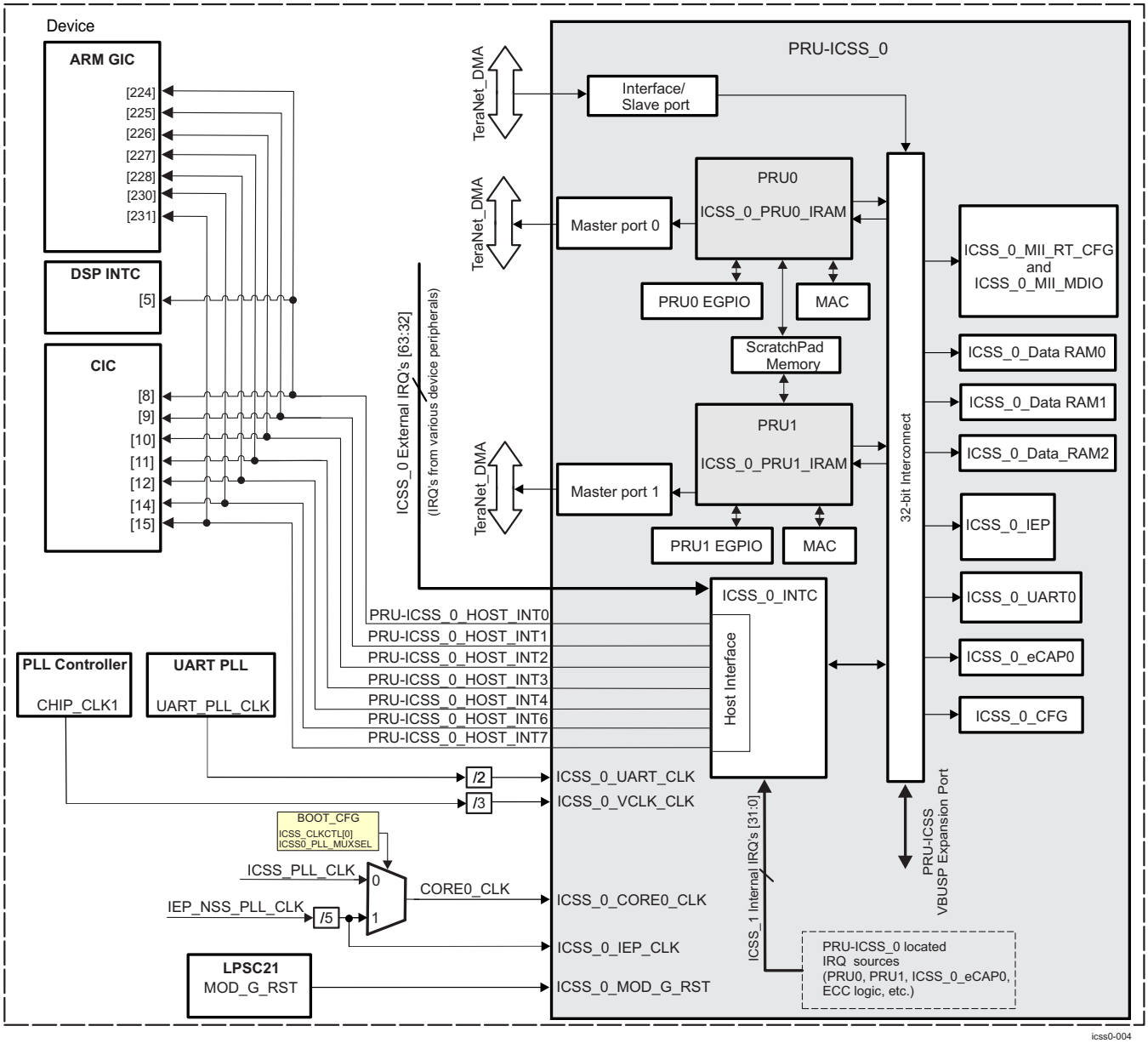
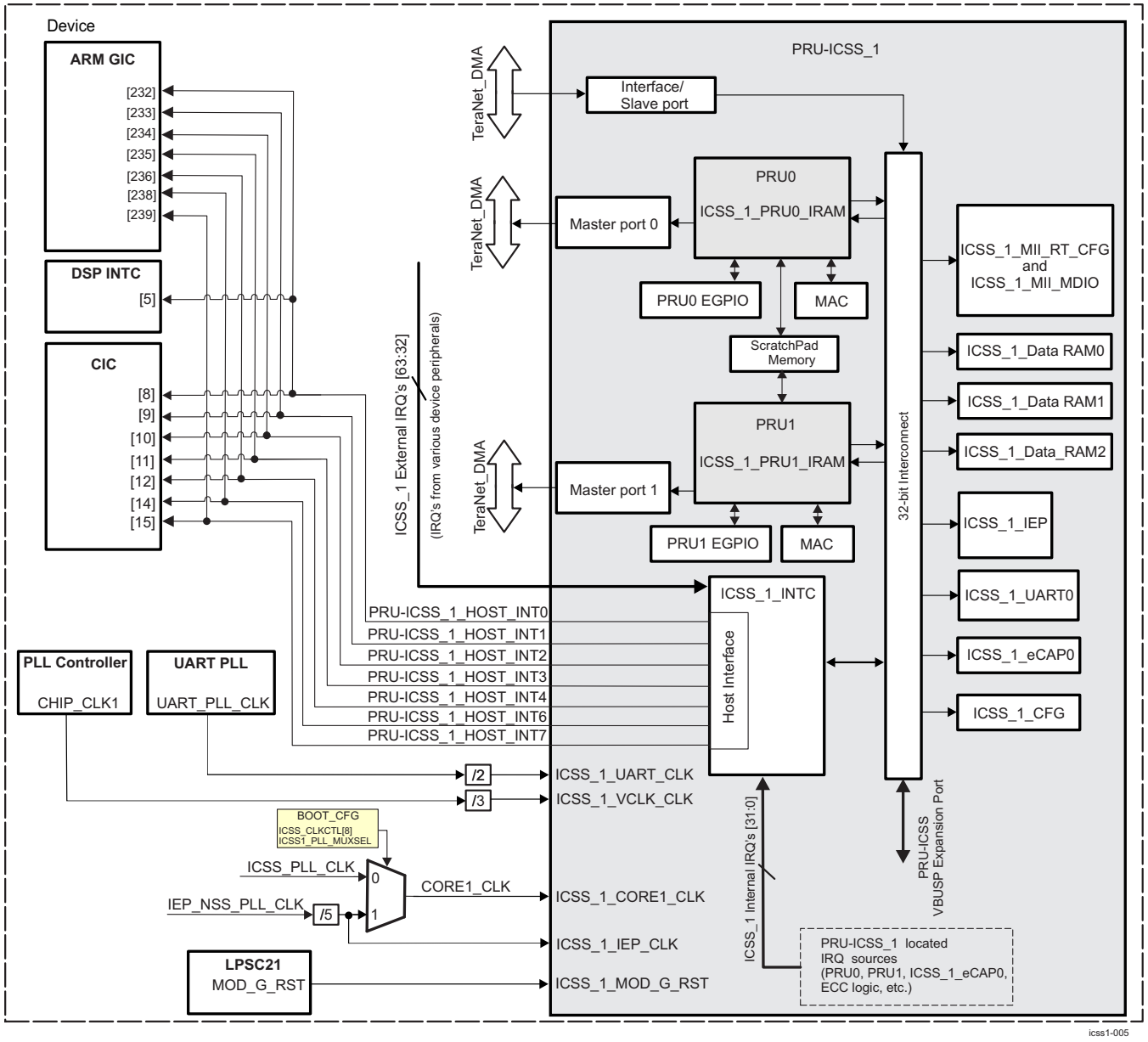


Figure 6-22. PRU-ICSS_1 Integration in the Device



The PRU-ICSS_0 and PRU-ICSS_1 integration in the device features:

- PD11 power domain instantiation
- Two master ports (PRU0 and PRU1 core initiators) on the device TeraNet_DMA interconnect
- One slave (configuration) port on the TeraNet_DMA interconnect for device hosts (ARMSS, DSP, etc.) to access various memories and registers of PRU-ICSS
- 10 output interrupt events from local interrupt controller — ICSS_0_INTC/ ICSS_1_INTC:
  - 2 events to each PRU core (events 0 and 1)
  - 7 events mapped to the device interrupt controllers (PRU-ICSS_<0/1>_HOST_INT0 event through PRU-ICSS_<0/1>_HOST_INT4 and events PRU-ICSS_<0/1>_HOST_INT6 through PRU-ICSS_<0/1>_HOST_INT7)
  - 1 event for the other PRU-ICSS (pr<0/1>_host_intr5_intr_pend_req). For more information please see Table 6-369 and Table 6-370.
- A local software gating of clocks to several modules within PRU subsystem (local clock management

protocol), as follows:
  - PRU-ICSS_IEP
  - PRU-ICSS_eCAP0
  - PRU-ICSS_UART0
  - PRU-ICSS_INTC
  - PRU-ICSS_PRU0
  - PRU-ICSS_PRU1
- 4 input clocks obtained from device PLL Controller, IEP/ NSS PLL, UART PLL and ICSS PLL:
  - a PRU-ICSS VBUS interface clock
  - a PRU-ICSS IEP functional clock
  - a PRU-ICSS UART0 clock
  - a PRU-ICSS Core clock
- Two input PRUSS_MII_RT_CFG RX and 2 input PRUSS_MII_RT_CFG TX clocks passed as external clocks on device I/Os
- No memory/register retention is supported
- One hardware non-retention (level sensitive) reset

Table 6-42 through Table 6-44 summarize the integration of the module in the device.

Table 6-42. PRU-ICSS Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
PRU-ICSS_0	PD11	LPSC21	TeraNet_DMA
PRU-ICSS_1	PD11	LPSC21	TeraNet_DMA

Table 6-43. PRU-ICSS Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
PRU-ICSS_0	ICSS_0_UART_CLK	UART_PLL_CLK / 2	UART PLL	PRU-ICSS_0 UART0 clock: ICSS_0_UART0 functional clock
	ICSS_0_VCLK_CLK	CHIP_CLK1 / 3	PLL Controller	PRU-ICSS_0 VBUS Interface clock
	ICSS_0_CORE0_CLK	CORE0_CLK	ICSS PLL or IEP/ NSS PLL output clocks	PRU-ICSS_0 Core clock: ICSS0 core clock
	ICSS_0_IEP_CLK	IEP_NSS_PLL_CLK / 5	Divided version of the IEP/ NSS PLL output clock	PRU-ICSS_0 Industrial Ethernet Peripheral functional clock
PRU-ICSS_1	ICSS_1_UART_CLK	UART_PLL_CLK / 2	UART PLL	PRU-ICSS_1 UART0 clock: ICSS_1_UART0 functional clock
	ICSS_1_VCLK_CLK	CHIP_CLK1 / 3	PLL Controller	PRU-ICSS_1 VBUS Interface clock
	ICSS_1_CORE1_CLK	CORE1_CLK	ICSS PLL or IEP/ NSS PLL output clocks	PRU-ICSS_1 Core clock: ICSS1 core clock
	ICSS_1_IEP_CLK	IEP_NSS_PLL_CLK / 5	Divided version of the IEP/ NSS PLL output clock	PRU-ICSS_1 Industrial Ethernet Peripheral functional clock
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
PRU-ICSS_0	ICSS_0_MOD_G_RST	MOD_G_RST	LPSC21	PRU-ICSS_0 module main reset
PRU-ICSS_1	ICSS_1_MOD_G_RST	MOD_G_RST	LPSC21	PRU-ICSS_1 module main reset

Table 6-44. PRU-ICSS Hardware Requests

Interrupt Requests					
Module Instance	**Event Name**	**Mapped To Input Event [Number]**			**Description**
		ARM GIC	**CIC**	**DSP INTC**	
PRU-ICSS_0	PRU-ICSS_0_HOST_INT0	[224]	[0]	[4]	PRU-ICSS_0 Host interrupt 0
	PRU-ICSS_0_HOST_INT1	[225]	[1]	-	PRU-ICSS_0 Host interrupt 1
	PRU-ICSS_0_HOST_INT2	[226]	[2]	-	PRU-ICSS_0 Host interrupt 2
	PRU-ICSS_0_HOST_INT3	[227]	[3]	-	PRU-ICSS_0 Host interrupt 3
	PRU-ICSS_0_HOST_INT4	[228]	[4]	-	PRU-ICSS_0 Host interrupt 4
	PRU-ICSS_0_HOST_INT6	[230]	[6]	-	PRU-ICSS_0 Host interrupt 6
	PRU-ICSS_0_HOST_INT7	[231]	[7]	-	PRU-ICSS_0 Host interrupt 7
PRU-ICSS_1	PRU-ICSS_1_HOST_INT0	[232]	[8]	[5]	PRU-ICSS_1 Host interrupt 0
	PRU-ICSS_1_HOST_INT1	[233]	[9]	-	PRU-ICSS_1 Host interrupt 1
	PRU-ICSS_1_HOST_INT2	[234]	[10]	-	PRU-ICSS_1 Host interrupt 2
	PRU-ICSS_1_HOST_INT3	[235]	[11]	-	PRU-ICSS_1 Host interrupt 3
	PRU-ICSS_1_HOST_INT4	[236]	[12]	-	PRU-ICSS_1 Host interrupt 4
	PRU-ICSS_1_HOST_INT6	[238]	[14]	-	PRU-ICSS_1 Host interrupt 6
	PRU-ICSS_1_HOST_INT7	[239]	[15]	-	PRU-ICSS_1 Host interrupt 7

6.4.4 PRU-ICSS Level Resources Functional Description

This section provides functional description of the device integrated PRU Subsystems modules.

6.4.4.1 PRU-ICSS Reset Management

The device supports warm reset isolation (Hard/Soft Reset, Watchdog Reset) on PRU-ICSS_0 and PRU-ICSS_1.

For more details on the PRU-ICSS Resets Isolation, see also the Section 5.3.10, *ICSS Reset Isolation*.

6.4.4.2 PRU-ICSS Power and Clock Management

The PRU-ICSS supports two levels of clock gating. First level gates all clocks inside the PRU-ICSS when requested by the PSC (Power Sleep Controller). The second level allows user software to enable/disable clocks in the clock gating register PRUSS_CGR to some internal modules, as follows:

- ICSS_0_IEP
- ICSS_0_eCAP0
- ICSS_0_UART0
- ICSS_0_INTC
- ICSS_0 PRU0 Core /ICSS_0_PRU0_IRAM
- ICSS_0 PRU1 Core /ICSS_0_PRU1_IRAM
- ICSS_1_IEP
- ICSS_1_eCAP0
- ICSS_1_UART0
- ICSS_1_INTC
- ICSS_1_PRU0 Core /ICSS_1_PRU0_IRAM
- ICSS_1_PRU1 Core/ICSS_1_PRU1_IRAM

The appropriate configuration registers block controls its local module set inside PRU-ICSS.

6.4.4.2.1 Module Clock Configurations at PRU-ICSS Top Level

IEP functional clock source selection: The clock source selection between ICSS_IEP_CLK (default) and ICSS_VCLK_CLK to the IEP module is done in register PRUSS_IEPCLK[0] OCP_EN in the PRUSS_CFG location. For more information on these PRU-ICSS level input clocks, refer to the Section 6.4.3.

Enhanced GPIO clock divider settings: In certain sample/shift clock settings of the PRU0 and PRU1 EGPIOs (when enabled in serial mode) two cascaded fractional dividers are done in the PRUSS_CFG top level configuration registers PRUSS_GPCFG0 and PRUSS_GPCFG1. In addition, EGPIO clock active edge selection control can be exerted via the bit PRU0_GPI_CLK_MODE for PRU0_EGPIO and PRU1_GPI_CLK_MODE for the PRU1_EGPIO.

- For the serial PRU0's EGPOs:
  - PRUSS_GPCFG0 [24-20]PRU0_GPO_DIV1
  - PRUSS_GPCFG0 [19-15]PRU0_GPO_DIV0
- For the serial PRU0's EGPIs:
  - PRUSS_GPCFG0 [12-8]PRU0_GPI_DIV1
  - PRUSS_GPCFG0 [7-3]PRU0_GPI_DIV0
- For the serial PRU1's EGPOs:
  - PRUSS_GPCFG1 [24-20]PRU1_GPO_DIV1
  - PRUSS_GPCFG1 [19-15]PRU1_GPO_DIV0
- For the serial PRU1's EGPIs:
  - PRUSS_GPCFG1 [12-8]PRU1_GPI_DIV1
  - PRUSS_GPCFG1 [7-3]PRU1_GPI_DIV0

6.4.4.3 Other PRU-ICSS Module Functional Registers at Subsystem Level

Enhanced GPIO. The other functional mode setting for PRUs EGPIOs at PRU-ICSS top registers level are:

- PRUSS_GPCFG0/PRUSS_GPCFG1 [14] PRU1_GPO_MODE — to select between direct or serial EGPO output mode of operation.
- PRUSS_GPCFG0/PRUSS_GPCFG1 [25] PRU1_GPO_SH_SEL — to select between the EGPO shadow registers 0 and 1 used for output shifting. For more details, refer to the Section 6.4.5.2.2.3.4, *Enhanced General-Purpose Module Outputs (R30)*.
- PRUSS_GPCFG0/PRUSS_GPCFG1 [1-0] PRU1_GPI_MODE — selects the EGPI input mode of operation (selects between "direct input", "parallel capture", "28-bit shift" or "MII_RT" modes).
- PRUSS_GPCFG0/PRUSS_GPCFG1 [13] PRU1_GPI_SB — **start bit event status for 28-bit EGPI input shift mode**. For more details, refer to the Section 6.4.5.2.2.3, *Enhanced General-Purpose Module Inputs (R31)*.

Enable address offset ("-0008 0000h") feature individually per PRU0 and PRU1 master ports in the PRUSS_PMAO register in case of accessing peripherals located in the PRU-ICSS space.

PRUSS_MII_RT_CFG interrupts mapping to PRU-ICSS_INTC is enabled in the PRUSS_MII_RT register.

PRUs scratchpad (SPAD) memory priority and configuration related bits are located in the PRUSS_SPP register.

6.4.4.4 PRU-ICSS Memory Maps

The PRU-ICSS comprises various distinct addressable regions that are mapped to both a local and global memory map. The local memory maps are maps with respect to the PRU point of view. The global memory maps are maps with respect to the Host point of view, but can also be accessed by the PRU-ICSS. Each PRU-ICSS can also access the memories within the other subsystem without going through an external port, thanks to PRU-ICSS VBUSP expansion port.

6.4.4.4.1 PRU-ICSS Local Memory Map

The PRU-ICSS memory map is documented in Table 6-45 (Instruction Space) and in Table 6-46 (Data Space). Note that these two memory maps are implemented inside the PRU-ICSS and are local to the components of the PRU-ICSS.

6.4.4.4.1.1 PRU-ICSS Local Instruction Memory Map

Each PRU has a dedicated 16 KB of Instruction Memory (16KB for PRU-ICSS_0, 16KB for PRU-ICSS_1) which needs to be initialized by an external to PRU-ICSS host processor before a PRU core executes any instructions.

CAUTION

The ICSS_0_PRU0/1_IRAM regions are ONLY accessible from masters, external to the PRU-ICSS (like Arm Cortex-A15, DSP, etc.) when the PRU0/PRU1 is NOT running. The access is via PRU-ICSS slave port on the device TeraNet_DMA interconnect.

Table 6-45. PRU-ICSS Local Instruction Memory Map

Start Address	PRU-ICSS_0		PRU-ICSS_1	
	PRU0	PRU1	PRU0	PRU1
0000 0000h	16 KB IRAM	16 KB IRAM	16 KB IRAM	16 KB IRAM

6.4.4.4.1.2 PRU-ICSS Local Data Memory Map

The local data memory map in Table 6-46 allows each PRU core to access the PRU-ICSS addressable regions (both its own subsystem and the other subsystem) and the external host's memory map.

The PRU accesses the other PRU-ICSS memory map through an VBUSP expansion port (External VBUSP port), starting at address 0004 0000h. The address seen by the other PRU-ICSS will be translated by hardware, subtracting 0004 0000h.

The PRU accesses the external Host memory map through the Interface Master port, starting at address 0008 0000h. By default, memory addresses between 0000 0000h–0007 FFFFh will correspond to the PRU-ICSS local address in Table 6-46. To access an address between 0000 0000h–0007 FFFFh of the external host map, the address offset of " –0008 0000h" feature is enabled through the PRUSS_PMAO[1] PMAO_PRU1 (for PRU1 core) and PRUSS_PMAO[0] PMAO_PRU0 (for PRU0 core) bits in the PRUSS_CFG subsystem level register space.

Table 6-46. PRU-ICSS Local Data Memory Map

Start Address	PRU-ICSS_0		PRU-ICSS_1	
	PRU0	PRU1	PRU0	PRU1
0000 0000h	PRU-ICSS_0 Data 8 KB RAM0	PRU-ICSS_0 Data 8 KB RAM1	PRU-ICSS_1 Data 8 KB RAM0	PRU-ICSS_1 Data 8 KB RAM1
0000 2000h	PRU-ICSS_0 Data 8 KB RAM1(1)	PRU-ICSS_0 Data 8 KB RAM0(1)	PRU-ICSS_1 Data 8 KB RAM1(1)	PRU-ICSS_1 Data 8 KB RAM0(1)
0000 4000h	Reserved	Reserved	Reserved	Reserved
0001 0000h	PRU-ICSS_0 Data 64 KB RAM2 (Shared RAM)	PRU-ICSS_0 Data 64 KB RAM2 (Shared RAM)	PRU-ICSS_1 Data 64 KB RAM2 (Shared RAM)	PRU-ICSS_1 Data 64 KB RAM2 (Shared RAM)
0002 0000h	PRU-ICSS_0 INTC	PRU-ICSS_0 INTC	PRU-ICSS_1 INTC	PRU-ICSS_1 INTC
0002 2000h	PRU-ICSS_0 PRU0 Control	PRU-ICSS_0 PRU0 Control	PRU-ICSS_1 PRU0 Control	PRU-ICSS_1 PRU0 Control
0002 2400h	Reserved	Reserved	Reserved	Reserved

(1) Direct access from PRU0 to Data RAM 1 and PRU1 to Data RAM 0.

Table 6-46. PRU-ICSS Local Data Memory Map (continued)

Start Address	PRU-ICSS_0		PRU-ICSS_1	
	PRU0	PRU1	PRU0	PRU1
0002 4000h	PRU-ICSS_0 PRU1 Control	PRU-ICSS_0 PRU1 Control	PRU-ICSS_1 PRU1 Control	PRU-ICSS_1 PRU1 Control
0002 4400h	Reserved	Reserved	Reserved	Reserved
0002 6000h	PRU-ICSS_0 CFG	PRU-ICSS_0 CFG	PRU-ICSS_1 CFG	PRU-ICSS_1 CFG
0002 7000h	PRU-ICSS_0 ECC_CFG	PRU-ICSS_0 ECC_CFG	PRU-ICSS_1 ECC_CFG	PRU-ICSS_1 ECC_CFG
0002 8000h	PRU-ICSS_0 UART_0	PRU-ICSS_0 UART_0	PRU-ICSS_1 UART_0	PRU-ICSS_1 UART_0
0002 A000h	Reserved	Reserved	Reserved	Reserved
0002 C000h	Reserved	Reserved	Reserved	Reserved
0002 E000h	PRU-ICSS_0 IEP	PRU-ICSS_0 IEP	PRU-ICSS_1 IEP	PRU-ICSS_1 IEP
0003 0000h	PRU-ICSS_0 eCAP_0	PRU-ICSS_0 eCAP_0	PRU-ICSS_1 eCAP_0	PRU-ICSS_1 eCAP_0
0003 2000h	PRU-ICSS_0 MII_RT_CFG	PRU-ICSS_0 MII_RT_CFG	PRU-ICSS_1 MII_RT_CFG	PRU-ICSS_1 MII_RT_CFG
0003 2400h	PRU-ICSS_0 MII_MDIO	PRU-ICSS_0 MII_MDIO	PRU-ICSS_1 MII_MDIO	PRU-ICSS_1 MII_MDIO
0003 4000h	Reserved	Reserved	Reserved	Reserved
0003 7000h	Reserved	Reserved	Reserved	Reserved
0003 8000h	Reserved	Reserved	Reserved	Reserved
0003 B000h	Reserved	Reserved	Reserved	Reserved
0004 0000h	External VBUSP port to PRU-ICSS_1	External VBUSP port to PRU-ICSS_1	External VBUSP port to PRU-ICSS_0	External VBUSP port to PRU-ICSS_0
0008 0000h	System VBUS	System VBUS	System VBUS	System VBUS

6.4.4.4.2 PRU-ICSS Global Memory Map

The global view of the PRU-ICSS internal memories and control ports is shown in Table 6-47. The offset addresses of each region are implemented inside the PRU-ICSS but the global device memory mapping places the PRU-ICSS slave port in the address range shown in the external PRU-ICSS Host top-level memory map.

The global memory map is with respect to the Host point of view (that is, device Arm Cortex-A15, DSP), but it can also be accessed by the PRU-ICSS_0/PRU-ICSS_1 itself. This is implemented via TeraNet_DMA redirecting PRU-ICSS master port traffic in the address range (0008 0000h–000B FFFFh) to the PRU-ICSS slave port when PMAO_PRU0/PMAO_PRU1 = '0b1'. Note that PRU0 and PRU1 can use either the local or global addresses to access their internal memories, but using the local addresses provides access time several cycles faster than using the global addresses. This is because when accessing via the global address the access has to be routed through the TeraNet_DMA switch fabric outside PRU-ICSS and back in through the PRU-ICSS slave port.

Example 1: PRU1 accesses its own data RAM — Data_RAM1 in the global space:

- The PRU1 CPU sets the PRUSS_PMAO[1] PMAO_PRU1 to 1 (using PRU1 local CFG address: 0002 6028h of that register) and generates destination address 0008 2000h. Thus, traffic passes through master port 1 towards PRU-ICSS_1 slave port over TeraNet_DMA to reach Data_RAM1 (location 0008 2000h - 0008 0000h = 0000 2000h in the Table 6-47).

Example 2: PRU1 accesses PRU0 data RAM — Data_RAM0 in the global space:

- The PRU1 CPU sets the PRUSS_PMAO[1] PMAO_PRU1 to 1 (using PRU1 local CFG MMR address: 0002 6028h of that register) and generates destination address 0008 0000h. Thus traffic passes through master port 1 towards PRU-ICSS1 slave port over TeraNet_DMA to reach Data_RAM0.

Example 4: PRU0 accesses a non-PRU-ICSS peripheral in the global space (address offset >= 2000 0000h):

- To access the MCASP1 config space the PRU0 keeps PRUSS_PMAO[0] PMAO_PRU0 at 0b0 and generates the MCASP1 CFG slave base address 2180 4400h. Thus traffic passes through Master port 0 and reaches MCASP1 config registers over TeraNet_DMA.

Example 5: ICSS_0_PRU1 host configures the PRU-ICSS_1 module ICSS_1_MII_RT:

- Note that in case of ICSS_0_PRU0 accessing a PRU-ICSS_1 peripheral, it must again disable the PMAO feature (writing at 0002 6028h PRUSS_PMAO[1] PMAO_PRU1 =0b0) and generate the physical address through its master port (1) adding global memory space offset of the IEP (0002 E000h from the Table 6-47) to the PRU-ICSS_1 TeraNet_DMA base address (20AC 0000h). The physical address generated from PRU-ICSS1 PRU1 therefore equals 20AE E000h).

Each of the PRU cores can access the rest of the device memory (including memory mapped peripheral and configuration registers) using the global memory space addresses.

Table 6-47. PRU-ICSS Global Memory Map

Offset Address	PRU-ICSS_0
	Target
0000 0000h	PRU-ICSS_0 Data 8 KB RAM0
0000 2000h	PRU-ICSS_0 Data 8 KB RAM1
0001 0000h	PRU-ICSS_0 Data 64 KB RAM2 (shared)
0002 0000h	PRU-ICSS_0 INTC
0002 2000h	PRU-ICSS_0 PRU0 Control
0002 2400h	PRU-ICSS_0 PRU0 Debug
0002 4000h	PRU-ICSS_0 PRU1 Control
0002 4400h	PRU-ICSS_0 PRU1 Debug
0002 6000h	PRU-ICSS_0 CFG
0002 7000h	PRU-ICSS_0 ECC_CFG
0002 8000h	PRU-ICSS_0 UART_0
0002 A000h	Reserved
0002 C000h	Reserved
0002 E000h	PRU-ICSS_0 IEP
0003 0000h	PRU-ICSS_0 eCAP_0
0003 2000h	PRU-ICSS_0 MII_RT_CFG
0003 2400h	PRU-ICSS_0 MII_MDIO
0003 4000h	PRU-ICSS_0 PRU0 16 KB IRAM
0003 8000h	PRU-ICSS_0 PRU1 16 KB IRAM
0004 0000h	External VBUSP port to PRU-ICSS_1

NOTE: The 0008_0000h-offset-subtraction feature must be enabled only in case of PRU global accesses (0008 0000h–000B FFFFh) to resources within the PRU subsystem. The PMAO feature must be disabled when accessing PRU-ICSS external locations.

6.4.4.5 PRU_ICSS_CFG Registers

Table 6-49 lists the memory-mapped registers for the PRU-ICSS subsystem. All register offset addresses not listed in Table 6-49 should be considered as reserved locations and the register contents should not be modified.

Table 6-48. PRU_ICSS_CFG Instances

Instance	Base Address
PRU_ICSS_0_CFG	20AA 6000h
PRU_ICSS_1_CFG	20AE 6000h

Table 6-49. PRU_ICSS_CFG Registers

Offset	Acronym	Register Name	PRU_ICSS_0_CFG Physical Address	PRU_ICSS_1_CFG Physical Address	Section
0h	PRUSS_REVID	Revision Register	20AA 6000h	20AE 6000h	Section 6.4.4.5.1
4h	RESERVED		20AA 6004h	20AE 6004h	
8h	PRUSS_GPCFG0	General Purpose Configuration 0 Register	20AA 6008h	20AE 6008h	Section 6.4.4.5.2
Ch	PRUSS_GPCFG1	General Purpose Configuration 1 Register	20AA 600Ch	20AE 600Ch	Section 6.4.4.5.3
10h	PRUSS_CGR	Clock Gating Register	20AA 6010h	20AE 6010h	Section 6.4.4.5.4
14h	RESERVED		20AA 6014h	20AE 6014h	
28h	PRUSS_PMAO	PRU Master Address Offset Register	20AA 6028h	20AE 6028h	Section 6.4.4.5.5
2Ch	PRUSS_MII_RT	MII_RT Event Enable Register	20AA 602Ch	20AE 602Ch	Section 6.4.4.5.6
30h	PRUSS_IEPCLK	IEP Clock Source Register defines the source of the IEP clock.	20AA 6030h	20AE 6030h	Section 6.4.4.5.7
34h	PRUSS_SPP	Scratch Pad Priority and Configuration Register	20AA 6034h	20AE 6034h	Section 6.4.4.5.8
40h	PRUSS_PIN_MX	Pin Mux Select Register	20AA 6040h	20AE 6040h	Section 6.4.4.5.9

6.4.4.5.1 PRUSS_REVID Register (Offset = 0h) [reset = 47000A02h]

PRUSS_REVID is shown in Figure 6-23 and described in Table 6-51.

The Revision Register contains the ID and revision information.

Table 6-50. PRUSS_REVID Instances

Instance	Physical Address
PRU_ICSS_0_CFG	20AA 6000h
PRU_ICSS_1_CFG	20AE 6000h

Figure 6-23. PRUSS_REVID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REVISION																															
R-47000A02h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-51. PRUSS_REVID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REVISION	R	47000A02h	TI internal data. Identifies revision of peripheral.

Table 6-52. Register Call Summary for PRUSS_REVID

PRU-ICSS Level Resources Functional Description
- PRUSS_REVID Register (Offset = 0h) [reset = 47000A02h]: [0]
- PRU_ICSS_CFG Registers: [0]

6.4.4.5.2 PRUSS_GPCFG0 Register (Offset = 8h) [reset = 0h]

PRUSS_GPCFG0 is shown in Figure 6-24 and described in Table 6-54.

The General Purpose Configuration 0 Register defines the GPI/O configuration for PRU0.

Table 6-53. PRUSS_GPCFG0 Instances

Instance	Physical Address
PRU_ICSS_0_CFG	20AA 6008h
PRU_ICSS_1_CFG	20AE 6008h

Figure 6-24. PRUSS_GPCFG0 Register

31	30	29	28	27	26	25	24
RESERVED		PR1_PRU0_GP_MUX_SEL				PRU0_GPO_SH_SEL	PRU0_GPO_DIV1
R-0h		R/W-0h				R-0h	R/W-0h
23	22	21	20	19	18	17	16
PRU0_GPO_DIV1				PRU0_GPO_DIV0			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
PRU0_GPO_DIV0	PRU0_GPO_MODE	PRU0_GPI_SB	PRU0_GPI_DIV1				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				
7	6	5	4	3	2	1	0
PRU0_GPI_DIV0					PRU0_GPI_CLK_MODE	PRU0_GPI_MODE	
R/W-0h					R/W-0h	R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-54. PRUSS_GPCFG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29-26	PR1_PRU0_GP_MUX_SEL	R/W	0h	Controls the PRU-ICSS wrap mux selection. 0h: GP selected 1h: ENDAT (Peripheral Interface) 2h: MII mode 3h: SD mode 4h: Reserved
25	PRU0_GPO_SH_SEL	R	0h	Defines which shadow register is currently getting used for GPO shifting. 0h: gpo_sh0 is selected 1h: gpo_sh1 is selected
24-20	PRU0_GPO_DIV1	R/W	0h	Divisor value (divide by PRU0_GPO_DIV1 + 1). 0h: Div 1.0 1h: Div 1.5 2h: Div 2.0 .. 1Eh: Div 16.0 1Fh: Reserved

Table 6-54. PRUSS_GPCFG0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-15	PRU0_GPO_DIV0	R/W	0h	Divisor value (divide by PRU0_GPO_DIV0 + 1). 0h: Div 1.0 1h: Div 1.5 2h: Div 2.0 .. 1Eh: Div 16.0 1Fh: Reserved
14	PRU0_GPO_MODE	R/W	0h	PRU GPO (R30) modes: 0h: Direct output mode 1h: Serial output mode
13	PRU0_GPI_SB	R/W	0h	Start Bit event for 28-bit shift mode. PRU0_GPI_SB (pru0_r31_status[29]) is set when first capture of a 1 on pru0_r31_status[0]. Read 0h: Start Bit event has not occurred. Read 1h: Start Bit event occurred. Write 0h: No Effect. Write 1h: Will clear PRU0_GPI_SB and clear the whole shift register.
12-8	PRU0_GPI_DIV1	R/W	0h	Divisor value (divide by PRU0_GPI_DIV1 + 1). 0h: Div 1.0 1h: Div 1.5 2h: Div 2.0 .. 1Eh: Div 16.0 1Fh: Reserved
7-3	PRU0_GPI_DIV0	R/W	0h	Divisor value (divide by PRU0_GPI_DIV0 + 1). 0h: Div 1.0 1h: Div 1.5 2h: Div 2.0 .. 1Eh: Div 16.0 1Fh: Reserved
2	PRU0_GPI_CLK_MODE	R/W	0h	Parallel 16-bit capture mode clock edge. 0h: Use the positive edge of pru0_r31_status[16] 1h: Use the negative edge of pru0_r31_status[16]
1-0	PRU0_GPI_MODE	R/W	0h	PRU GPI (R31) modes: 0h: Direct input mode 1h: 16-bit parallel capture mode 2h: 28-bit shift mode 3h: MII_RT mode

Table 6-55. Register Call Summary for PRUSS_GPCFG0

PRU-ICSS PRU Cores

- Block Diagram and Signal Configuration: [0][1][2]
- Basic Programming Model: [0]

Table 6-55. Register Call Summary for PRUSS_GPCFG0 (continued)

PRU-ICSS Level Resources Functional Description • PRUSS_GPCFG0 Register (Offset = 8h) [reset = 0h]: [0] • Other PRU-ICSS Module Functional Registers at Subsystem Level: [0][1][2][3] • Module Clock Configurations at PRU-ICSS Top Level: [0][1][2][3][4] • PRU_ICSS_CFG Registers: [0]
Sigma Delta (SD) Decimation Filtering H/W • Basic Programming Example: [0] • Block Diagram and Signals: [0][1][2][3][4]
PRU-ICSS Environment • PRU-ICSS I/O Interface: [0][1]

6.4.4.5.3 PRUSS_GPCFG1 Register (Offset = Ch) [reset = 0h]

PRUSS_GPCFG1 is shown in Figure 6-25 and described in Table 6-57.

The General Purpose Configuration 1 Register defines the GPI/O configuration for PRU1.

Table 6-56. PRUSS_GPCFG1 Instances

Instance	Physical Address
PRU_ICSS_0_CFG	20AA 600Ch
PRU_ICSS_1_CFG	20AE 600Ch

Figure 6-25. PRUSS_GPCFG1 Register

31	30	29	28	27	26	25	24
RESERVED		PR1_PRU1_GP_MUX_SEL				PRU1_GPO_SH_SEL	PRU1_GPO_DIV1
R-0h		R/W-0h				R-0h	R/W-0h
23	22	21	20	19	18	17	16
PRU1_GPO_DIV1				PRU1_GPO_DIV0			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
PRU1_GPO_DIV0	PRU1_GPO_MODE	PRU1_GPI_SB	PRU1_GPI_DIV1				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				
7	6	5	4	3	2	1	0
PRU1_GPI_DIV0					PRU1_GPI_CLK_MODE	PRU1_GPI_MODE	
R/W-0h					R/W-0h	R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-57. PRUSS_GPCFG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29-26	PR1_PRU1_GP_MUX_SEL	R/W	0h	Controls the PRU-ICSS wrap mux selection. 0h: GP selected 1h: ENDAT (Peripheral Interface) 2h: MII mode 3h: SD mode 4h: Reserved
25	PRU1_GPO_SH_SEL	R	0h	Defines which shadow register is currently getting used for GPO shifting. 0h = gpo_sh0 is selected 1h = gpo_sh1 is selected
24-20	PRU1_GPO_DIV1	R/W	0h	Divisor value (divide by PRU1_GPO_DIV1 + 1). 0h: Div 1.0 1h: Div 1.5 2h: Div 2.0 .. 1Eh: Div 16.0 1Fh: Reserved

Table 6-57. PRUSS_GPCFG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-15	PRU1_GPO_DIV0	R/W	0h	Divisor value (divide by PRU1_GPO_DIV0 + 1). 0h: Div 1.0 1h: Div 1.5 2h: Div 2.0 .. 1Eh: Div 16.0 1Fh: Reserved
14	PRU1_GPO_MODE	R/W	0h	PRU GPO (R30) modes: 0h: Direct output mode 1h: Serial output mode
13	PRU1_GPI_SB	R/W	0h	28-bit shift mode Start Bit event. PRU1_GPI_SB (pru1_r31_status[29]) is set when first capture of a 1 on pru1_r31_status[0]. Read 0: Start Bit event has not occurred. Read 1: Start Bit event occurred. Write 0: No Effect. Write 1: Will clear PRU1_GPI_SB and clear the whole shift register.
12-8	PRU1_GPI_DIV1	R/W	0h	Divisor value (divide by PRU1_GPI_DIV1 + 1). 0h: Div 1.0 1h: Div 1.5 2h: Div 2.0 .. 1Eh: Div 16.0 1Fh: Reserved
7-3	PRU1_GPI_DIV0	R/W	0h	Divisor value (divide by PRU1_GPI_DIV0 + 1). 0h: Div 1.0 1h: Div 1.5 2h: Div 2.0 .. 1Eh: Div 16.0 1Fh: Reserved
2	PRU1_GPI_CLK_MODE	R/W	0h	Parallel 16-bit capture mode clock edge. 0h: Use the positive edge of pru1_r31_status[16] 1h: Use the negative edge of pru1_r31_status[16]
1-0	PRU1_GPI_MODE	R/W	0h	PRU GPI (R31) modes: 0h: Direct input mode 1h: 16-bit parallel capture mode 2h: 28-bit shift mode 3h: MII_RT mode

Table 6-58. Register Call Summary for PRUSS_GPCFG1

PRU-ICSS Level Resources Functional Description • PRUSS_GPCFG1 Register (Offset = Ch) [reset = 0h]: [0] • Other PRU-ICSS Module Functional Registers at Subsystem Level: [0][1][2][3] • Module Clock Configurations at PRU-ICSS Top Level: [0][1][2][3][4] • PRU_ICSS_CFG Registers: [0]
PRU-ICSS Environment • PRU-ICSS I/O Interface: [0][1]

6.4.4.5.4 PRUSS_CGR Register (Offset = 10h) [reset = 00024924h]

PRUSS_CGR is shown in Figure 6-26 and described in Table 6-60.

The Clock Gating Register controls the state of Clock Management of the different modules. Software should not clear {module}_CLK_EN until {module}_CLK_STOP_ACK is 1h.

Table 6-59. PRUSS_CGR Instances

Instance	Physical Address
PRU_ICSS_0_CFG	20AA 6010h
PRU_ICSS_1_CFG	20AE 6010h

Figure 6-26. PRUSS_CGR Register

31	30	29	28	27	26	25	24
ICSS_STOP_ACK	ICSS_STOP_REQ	RESERVED					
R/W-0h	R-0h	R-0h					
23	22	21	20	19	18	17	16
RESERVED						IEP_CLK_EN	IEP_CLK_STOP_ACK
R-0h						R/W-1h	R-0h
15	14	13	12	11	10	9	8
IEP_CLK_STOP_REQ	ECAP_CLK_EN	ECAP_CLK_STOP_ACK	ECAP_CLK_STOP_REQ	UART_CLK_EN	UART_CLK_STOP_ACK	UART_CLK_STOP_REQ	PRUSS_INTC_CLK_EN
R/W-0h	R/W-1h	R-0h	R/W-0h	R/W-1h	R-0h	R/W-0h	R/W-1h
7	6	5	4	3	2	1	0
PRUSS_INTC_CLK_STOP_ACK	PRUSS_INTC_CLK_STOP_REQ	PRU1_CLK_EN	PRU1_CLK_STOP_ACK	PRU1_CLK_STOP_REQ	PRU0_CLK_EN	PRU0_CLK_STOP_ACK	PRU0_CLK_STOP_REQ
R-0h	R/W-0h	R/W-1h	R-0h	R/W-0h	R/W-1h	R-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-60. PRUSS_CGR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ICSS_STOP_ACK	R/W	0h	Acknowledgement that ICSS clock can be stopped. 0h: Not Ready to Gate Clock 1h: Ready to Gate Clock
30	ICSS_STOP_REQ	R	0h	ICSS request to stop clock. 0h: No Gate Clock Request 1h: Gate Clock Request
29-18	RESERVED	R	0h	Reserved
17	IEP_CLK_EN	R/W	1h	IEP clock enable. 0h: Disable Clock 1h: Enable Clock
16	IEP_CLK_STOP_ACK	R	0h	Acknowledgement that IEP clock can be stopped. 0h: Not Ready to Gate Clock 1h: Ready to Gate Clock
15	IEP_CLK_STOP_REQ	R/W	0h	IEP request to stop clock. 0h: Do not request to stop Clock 1h: Request to stop Clock

Table 6-60. PRUSS_CGR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	ECAP_CLK_EN	R/W	1h	ECAP clock enable. 0h: Disable Clock 1h: Enable Clock
13	ECAP_CLK_STOP_ACK	R	0h	Acknowledgement that ECAP clock can be stopped. 0h: Not Ready to Gate Clock 1h: Ready to Gate Clock
12	ECAP_CLK_STOP_REQ	R/W	0h	ECAP request to stop clock. 0h: Do not request to stop Clock 1h: Request to stop Clock
11	UART_CLK_EN	R/W	1h	UART clock enable. 0h: Disable Clock 1h: Enable Clock
10	UART_CLK_STOP_ACK	R	0h	Acknowledgement that UART clock can be stopped. 0h: Not Ready to Gate Clock 1h: Ready to Gate Clock
9	UART_CLK_STOP_REQ	R/W	0h	UART request to stop clock. 0h: Do not request to stop Clock 1h: Request to stop Clock
8	PRUSS_INTC_CLK_EN	R/W	1h	PRUSS_INTC clock enable. 0h: Disable Clock 1h: Enable Clock
7	PRUSS_INTC_CLK_STOP_ACK	R	0h	Acknowledgement that PRUSS_INTC clock can be stopped. 0h: Not Ready to Gate Clock 1h: Ready to Gate Clock
6	PRUSS_INTC_CLK_STOP_REQ	R/W	0h	PRUSS_INTC request to stop clock. 0h: Do not request to stop Clock 1h: Request to stop Clock
5	PRU1_CLK_EN	R/W	1h	PRU1 clock enable. 0h: Disable Clock 1h: Enable Clock
4	PRU1_CLK_STOP_ACK	R	0h	Acknowledgement that PRU1 clock can be stopped. 0h: Not Ready to Gate Clock 1h: Ready to Gate Clock
3	PRU1_CLK_STOP_REQ	R/W	0h	PRU1 request to stop clock. 0h: Do not request to stop Clock 1h: Request to stop Clock
2	PRU0_CLK_EN	R/W	1h	PRU0 clock enable. 0h: Disable Clock 1h: Enable Clock
1	PRU0_CLK_STOP_ACK	R	0h	Acknowledgement that PRU0 clock can be stopped. 0h: Not Ready to Gate Clock 1h: Ready to Gate Clock
0	PRU0_CLK_STOP_REQ	R/W	0h	PRU0 request to stop clock. 0h: Do not request to stop Clock 1h: Request to stop Clock

Table 6-61. Register Call Summary for PRUSS_CGR

PRU-ICSS Level Resources Functional Description
• PRUSS_CGR Register (Offset = 10h) [reset = 00024924h]: [0] • PRU_ICSS_CFG Registers: [0] • PRU-ICSS Power and Clock Management: [0]

6.4.4.5.5 PRUSS_PMAO Register (Offset = 28h) [reset = 0h]

PRUSS_PMAO is shown in Figure 6-27 and described in Table 6-63.

The PRU Master Address Offset Register enables for the PRU Master Port Address to have an offset of minus 0008_0000h. This enables the PRU to access External Host address space starting at 0000_0000h.

Table 6-62. PRUSS_PMAO Instances

Instance	Physical Address
PRU_ICSS_0_CFG	20AA 6028h
PRU_ICSS_1_CFG	20AE 6028h

Figure 6-27. PRUSS_PMAO Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						PMAO_PRU1	PMAO_PRU0
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-63. PRUSS_PMAO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	PMAO_PRU1	R/W	0h	PRU1 Master Port Address Offset Enable. 0h: Disable address offset. 1h: Enable address offset of -0008_0000h.
0	PMAO_PRU0	R/W	0h	PRU0 Master Port Address Offset Enable. 0h: Disable address offset. 1h: Enable address offset of -0008_0000h.

Table 6-64. Register Call Summary for PRUSS_PMAO

PRU-ICSS Level Resources Functional Description
- PRUSS_PMAO Register (Offset = 28h) [reset = 0h]: [0]
- PRU_ICSS_CFG Registers: [0]
- Other PRU-ICSS Module Functional Registers at Subsystem Level: [0]
- PRU-ICSS Global Memory Map: [0][1][2][3]
- PRU-ICSS Local Data Memory Map: [0][1]

6.4.4.5.6 PRUSS_MII_RT Register (Offset = 2Ch) [reset = 1h]

PRUSS_MII_RT is shown in Figure 6-28 and described in Table 6-66.

The MII_RT Event Enable Register enables MII_RT mode events to the PRU-ICSS_INTC.

Table 6-65. PRUSS_MII_RT Instances

Instance	Physical Address
PRU_ICSS_0_CFG	20AA 602Ch
PRU_ICSS_1_CFG	20AE 602Ch

Figure 6-28. PRUSS_MII_RT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							MII_RT_EVENT_EN
R-0h							R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-66. PRUSS_MII_RT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	MII_RT_EVENT_EN	R/W	1h	Enables the MII_RT Events to the PRU-ICSS_INTC. 0h: Disabled (use external events). 1h: Enabled (use MII_RT events).

Table 6-67. Register Call Summary for PRUSS_MII_RT

PRU-ICSS Local Interrupt Controller • PRU-ICSS Interrupt Requests Mapping: [0][1][2][3][4][5][6] • PRU-ICSS Interrupt Controller System Events: [0]
PRU-ICSS Level Resources Functional Description • PRUSS_MII_RT Register (Offset = 2Ch) [reset = 1h]: [0] • Other PRU-ICSS Module Functional Registers at Subsystem Level: [0] • PRU_ICSS_CFG Registers: [0]

6.4.4.5.7 PRUSS_IEPCLK Register (Offset = 30h) [reset = 0h]

PRUSS_IEPCLK is shown in Figure 6-29 and described in Table 6-69.

The IEP Clock Source Register defines the source of the IEP clock.

Table 6-68. PRUSS_IEPCLK Instances

Instance	Physical Address
PRU_ICSS_0_CFG	20AA 6030h
PRU_ICSS_1_CFG	20AE 6030h

Figure 6-29. PRUSS_IEPCLK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							OCP_EN
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-69. PRUSS_IEPCLK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	OCP_EN	R/W	0h	IEP clock source. 0h: ICSS_IEP_CLK is the source 1h: ICSS_VCLK_CLK is the source. While this is selected no transactions should be active. It can only be cleared by a hardware reset.

Table 6-70. Register Call Summary for PRUSS_IEPCLK

PRU-ICSS Level Resources Functional Description • PRUSS_IEPCLK Register (Offset = 30h) [reset = 0h]: [0] • Module Clock Configurations at PRU-ICSS Top Level: [0] • PRU_ICSS_CFG Registers: [0]
PRU-ICSS Industrial Ethernet Peripheral (IEP) • PRU-ICSS IEP Clock Generation: [0][1][2]

6.4.4.5.8 PRUSS_SPP Register (Offset = 34h) [reset = 0h]

PRUSS_SPP is shown in Figure 6-30 and described in Table 6-72.

The Scratch Pad Priority and Configuration Register defines the access priority assigned to the PRU cores and configures the scratch pad XFR shift functionality.

Table 6-71. PRUSS_SPP Instances

Instance	Physical Address
PRU_ICSS_0_CFG	20AA 6034h
PRU_ICSS_1_CFG	20AE 6034h

Figure 6-30. PRUSS_SPP Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						XFR_SHIFT_EN	PRU1_PAD_HP_EN
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-72. PRUSS_SPP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	XFR_SHIFT_EN	R/W	0h	Enables XIN XOUT shift functionality. When enabled, R0[4-0] (internal to PRU) defines the 32-bit offset for XIN and XOUT operations with the scratch pad. 0h: Disabled. 1h: Enabled.
0	PRU1_PAD_HP_EN	R/W	0h	Defines which PRU wins write cycle arbitration to a common scratch pad bank. The PRU which has higher priority will always perform the write cycle with no wait states. The lower PRU will get stalled wait states until higher PRU is not performing write cycles. If the lower priority PRU writes to the same byte has the higher priority PRU, then the lower priority PRU will over write the bytes. 0h: PRU0 has highest priority. 1h: PRU1 has highest priority.

Table 6-73. Register Call Summary for PRUSS_SPP

PRU-ICSS PRU Cores • Optional XIN/XOUT Shift: [0] • Example Scratch Pad Operations: [0]
PRU-ICSS Level Resources Functional Description • PRUSS_SPP Register (Offset = 34h) [reset = 0h]: [0] • Other PRU-ICSS Module Functional Registers at Subsystem Level: [0] • PRU_ICSS_CFG Registers: [0]

6.4.4.5.9 PRUSS_PIN_MX Register (Offset = 40h) [reset = 0h]

PRUSS_PIN_MX is shown in Figure 6-31 and described in Table 6-75.

The Pin Mux Select Register defines the state of the PRU-ICSS internal pinmuxing.

Table 6-74. PRUSS_PIN_MX Instances

Instance	Physical Address
PRU_ICSS_0_CFG	20AA 6040h
PRU_ICSS_1_CFG	20AE 6040h

Figure 6-31. PRUSS_PIN_MX Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				PWM3_REMAP_EN		PWM0_REMAP_EN	
R-0h				R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-75. PRUSS_PIN_MX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved. Always write 0.
11-10	PWM3_REMAP_EN	R/W	0h	Remaps the eHRPWM3_SYNCI to a PRU-ICSS Host Interrupt. This bit field is only controlled by PRU-ICSS_0. If enabled, PRU-ICSS Host Interrupt 7 controls eHRPWM3_SYNCI. See Section 11.5.3.2 for more details about eHRPWM3_SYNCI signal. 0h: Disabled 1h: PRU-ICSS_0 Host Interrupt 7 controls eHRPWM3_SYNCI 2h: PRU-ICSS_1 Host Interrupt 7 controls eHRPWM3_SYNCI 3h: Reserved
9-8	PWM0_REMAP_EN	R/W	0h	Remaps the eHRPWM0_SYNCI to a PRU-ICSS Host Interrupt. This bit field is only controlled by PRU-ICSS_0. If enabled, PRU-ICSS Host Interrupt 6 controls eHRPWM0_SYNCI. See Section 11.5.3.2 for more details about eHRPWM0_SYNCI signal. 0h: Disabled 1h: PRU-ICSS_0 Host Interrupt 6 controls eHRPWM0_SYNCI 2h: PRU-ICSS_1 Host Interrupt 6 controls eHRPWM0_SYNCI 3h: Reserved
7-0	RESERVED	R	0h	Reserved.

Table 6-76. Register Call Summary for PRUSS_PIN_MX

PRU-ICSS Level Resources Functional Description • PRUSS_PIN_MX Register (Offset = 40h) [reset = 0h]: [0] • PRU_ICSS_CFG Registers: [0]
PRU-ICSS Environment

6.4.5 PRU-ICSS PRU Cores

This section describes the functionality of the two Programmable Real-time Unit (PRU) processors (PRU0 and PRU1) integrated in each of the device PRU-ICSS.

6.4.5.1 PRU Cores Overview

The PRU is a processor optimized for performing embedded tasks that require manipulation of packed memory mapped data structures, handling of system events that have tight real-time constraints and interfacing with systems external to the SoC. The PRU is both very small and very efficient at handling such tasks.

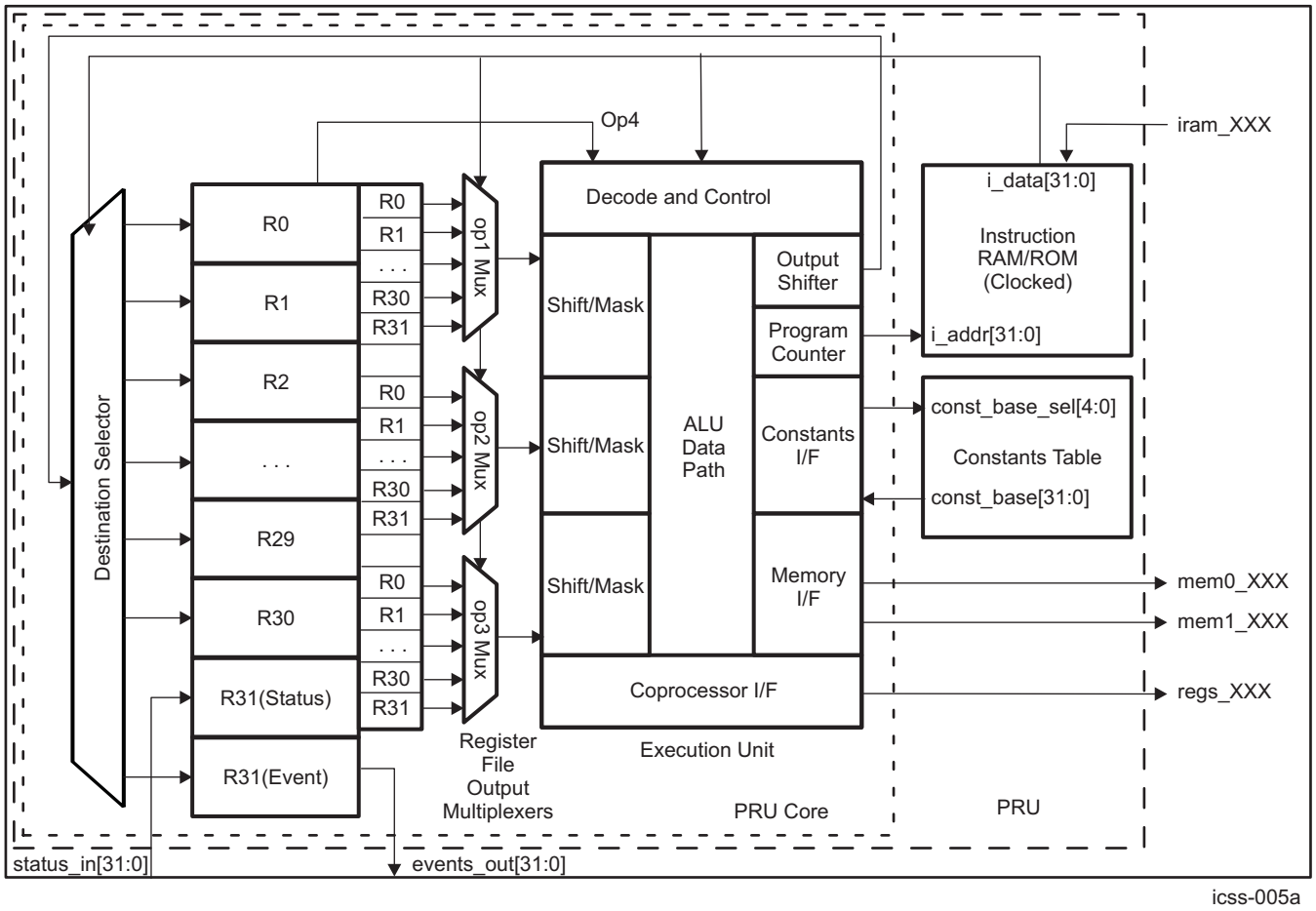
The major attributes of the PRU are in Table 6-77.

Table 6-77. PRU Features

Attribute	Value
IO Architecture	Load/Store
Data Flow Architecture	Register to Register
Core Level Bus Architecture	
Type	4-Bus Harvard (1 Instruction, 3 Data)
Instruction I/F	32-Bit
Memory I/F 0	32-Bit
Memory I/F 1	32-Bit
Execution Model	
Issue Type	Scalar
Pipelining	None (Purposefully)
Ordering	In Order
ALU Type	Unsigned Integer
Registers	
General Purpose (GP)	30 (R1 – R30)
External Status	1 (R31)
GP/Indexing	1 (R0)
Addressability in Instruction	Bit, Byte (8-bit), Half-word (16-bit), Word (32-bit), Pointer
Addressing Modes	
Load Immediate	16-bit Immediate
Load/Store – Memory	Register Base + Register Offset
	Register Base + 8-bit Immediate Offset
	Register Base with auto increment/decrement
	Constant Table Base + Register Offset
	Constant Table Base + 8-bit Immediate Offset
	Constant Table Base with auto increment/decrement
Data Path Width	32-bit
Instruction Width	32-bit
Accessibility to Internal PRU Structures	Provides 32-bit slave with three regions: • Instruction RAM • Control/Status registers • Debug access to internal registers (R0-R31) and constant table

The processor is based on a four-bus architecture which allows instructions to be fetched and executed concurrently with data transfers. In addition, an input is provided in order to allow external status information to be reflected in the internal processor status register. Figure 6-32 shows a block diagram of the processing element and the associated instruction RAM/ROM that contains the code that is to be executed.

Figure 6-32. PRU Block Diagram



6.4.5.2 PRU Cores Functional Description

This section describes the PRU cores supported functionality by describing the constant table, module interface and enhanced GPIOs.

6.4.5.2.1 PRUs Constant Table

The PRU Constants Table is a structure of hard-coded memory addresses for commonly used peripherals and memories. The constants table exists to more efficiently load/store data to these commonly accessed addresses by:

- Eliminating the PRU instruction that pre-loads a hard-coded address into the internal register file.
- Maximizing the usage of the PRU register file for embedded processing applications by moving many of the commonly used constant or deterministically calculated base addresses from the internal register file to an external table.

Table 6-78. PRU0/1 Constant Table

Entry No.	Region Pointed To	Value [31:0]
0	PRU-ICSS INTC (local)	0002_0000h
1	TIMER_1	0221_0000h
2	I2C_0	0253_0000h
3	PRU-ICSS eCAP (local)	0003_0000h
4	PRU-ICSS CFG (local)	0002_6000h
5	MMC0	2300_0000h
6	SPI0	2180_5400h
7	PRU-ICSS UART0 (local)	0002_8000h
8	MCASP_0 DMA	2180_4000h
9	NSSUL	0400_0000h
10	SEC_MGR	0250_0000h
11	UART_1	0253_1000h
12	UART_2	0253_1400h
13	CIC	0260_0000h
14	DCAN_0 CFG	0260_B200h
15	DCAN_1 CFG	0260_B400h
16	SPI1	2180_5800h
17	I2C_1	0253_0400h
18	EPWM_0	021D_0000h
19	ECAP_0	021D_1800h
20	PRU-ICSS_0, PRU-ICSS_1 IEP (Local) + 256MB	000A_E000h
21	PRU-ICSS MDIO (local)	0003_2400h
22	Semaphore	0264_0000h
23	Message Manager Queue Proxy Region5	02A5_0000h
24	PRU-ICSS PRU0/1 Data RAM (local)	0000_0n00h, n = c24_blk_index[3:0]
25	PRU-ICSS PRU1/0 Data RAM (local)	0000_2n00h, n = c25_blk_index[3:0]
26	PRU-ICSS IEP (local)	0002_En00h, n = c26_blk_index[3:0]
27	PRU-ICSS MII_RT (local)	0003_2n00h, n = c27_blk_index[3:0]
28	PRU-ICSS Shared RAM (local)	00nn_nn00h, nnnn = c28_pointer[15:0]
29	EDMA_0_CC_CFG	02nn_nn00h, nnnn = c29_pointer[15:0]
30	MSMC SRAM	0Cnn_nn00h, nnnn = c30_pointer[15:0]
31	DDR3A DATA	80nn_nn00h, nnnn = c31_pointer[15:0]

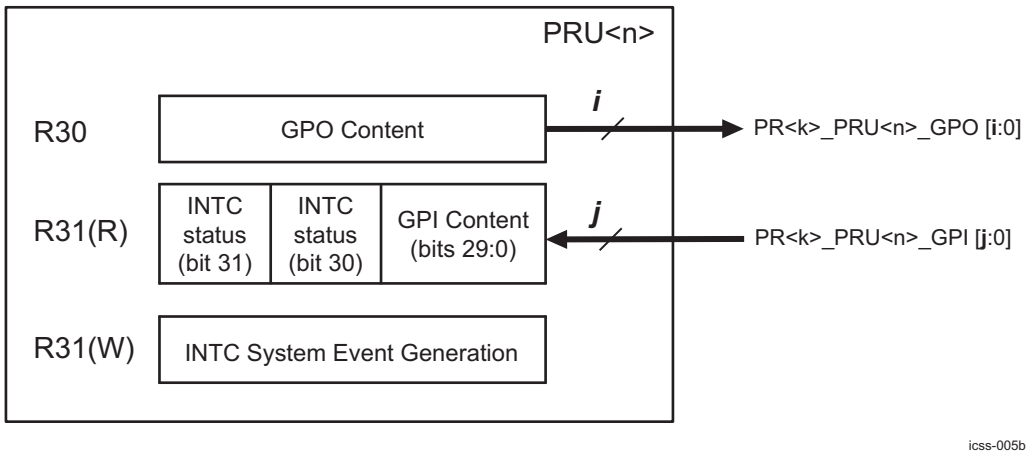
NOTE: The addresses in constants entries 24–31 are partially programmable. Their programmable bit field (for example, c24_blk_index[3:0]) is programmable through the PRU CTRL register space. As a general rule, the PRU should configure this field before using the partially programmable constant entries.

6.4.5.2.2 PRU Module Interface

The PRU module interface consists of the PRU internal registers 30 and 31 (R30 and R31). Figure 6-33 shows the PRU module interface and the functionality of R30 and R31. The register R31 serves as an interface with the dedicated PRU general purpose input (GPI) pins and ICSS_INTC. Reading R31 returns status information from the GPI pins and ICSS_INTC via the PRU Real Time Status Interface. Writing to R31 generates PRU system events via the PRU Event Interface. The register R30 serves as an interface with the dedicated PRU general purpose output (GPO) pins.

NOTE: The below sections cover different functional modes of the PRUn cores, (where n=0,1), enhanced GPIO (EGPIO) interface. The register bits which control EGPIO functionalities are part of the (PRUSS1_CFG and PRUSS2_CFG) space. For descriptions of these EGPIO register bitfield controls, refer to the Section 6.4.4.3.

Figure 6-33. PRU Module Interface



6.4.5.2.2.1 Real-Time Status Interface Mapping (R31): Interrupt Events Input

The PRU Real Time Status Interface directly feeds information into register 31 (R31) of the PRU's internal register file. The firmware on the PRU uses the status information to make decisions during execution. The status interface is comprised of signals from different modules inside of the PRU-ICSS which require some level of interaction with the PRU. More details on the Host interrupts imported into bit 30 and 31 of register R31 of both the PRUs is provided in the Section 6.4.6, *PRU-ICSS Local Interrupt Controller*.

Table 6-79. Real-Time Status Interface Mapping (R31) Field Descriptions

Bit	Field	Description
31	pru_intr_in[1]	PRU Host Interrupt 1 from local ICSS_INTC
30	pru_intr_in[0]	PRU Host Interrupt 0 from local ICSS_INTC
29:0	prun_r31_status[29:0]	Status inputs from primary input via Enhanced GPI port

6.4.5.2.2.2 Event Interface Mapping (R31): PRU System Events

This PRU Event Interface directly feeds pulsed event information out of the PRU's internal ALU. These events are exported out of the PRU-ICSS and need to be connected to the system interrupt controller at the SoC level. The event interface can be used by the firmware to create software interrupts from the PRU to the Host processor.

Figure 6-34. Event Interface Mapping (R31)

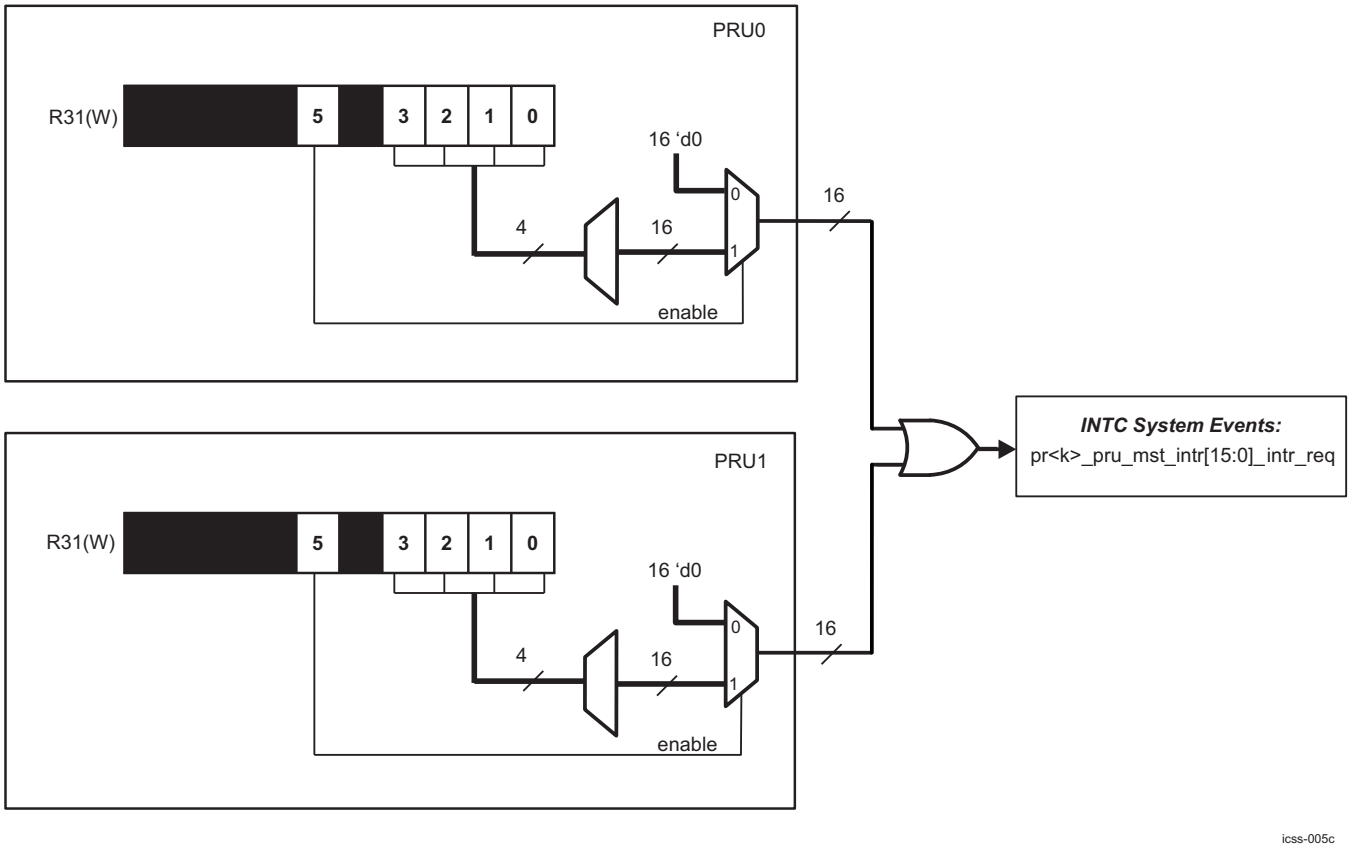


Table 6-80. Event Interface Mapping (R31) Field Descriptions

Bit	Field	Description
31:6	Reserved	
5	prun_r31_vec_valid	Valid strobe for vector output
4	Reserved	
3:0	prun_r31_vec[3:0]	Vector output

Simultaneously writing a '1' to prun_r31_vec_valid (R31 bit 5) and a channel number from 0 to 15 to prun_r31_vec[3:0] (R31 bits 3:0) creates a pulse on the output of the corresponding prk_pru_mst_intr[x]_intr_req INTC system event. For example, writing '100000' will generate a pulse on prk_pru_mst_intr[0]_intr_req, writing '100001' will generate a pulse on prk_pru_mst_intr[1]_intr_req, and so on to where writing '101111' will generate a pulse on prk_pru_mst_intr[15]_intr_req and writing '0xxxxx' will not generate any system event pulses. The output values from both PRU cores in a subsystem are ORed together.

The output channels 0-15 are connected to the ICSS_INTC system events 16-31, respectively. This allows the PRU to assert one of the system events 16-31 by writing to its own R31 register. The system event is used to either post a completion event to one of the host CPUs (Arm) or to signal the other PRU. The host to be signaled is determined by the system interrupt to interrupt channel mapping (programmable). The 16 events are named as prk_pru_mst_intr<15:0>_intr_req. See the Section 6.4.6.4, *PRU-ICSS Interrupt Requests Mapping* , in the section, *PRU-ICSS Local Interrupt Controller*, for more details.

6.4.5.2.2.3 General-Purpose Inputs (R31): Enhanced PRU GP Module

The PRU-ICSS implements an enhanced General Purpose Input/Output (GPIO) module with SCU that supports the following general-purpose input modes: direct input, 16-bit parallel capture, 28-bit serial shift in, and MII_RT. Register R31 serves as an interface with the general-purpose inputs. Table 6-81 describes the input modes in detail.

NOTE: Each PRU core can only be configured for one GPI mode at a time. Each mode uses the same R31 signals and internal register bits for different purposes. A summary is found in Table 6-82.

NOTE: The PRUSS_GPCFGn register, bitfield [29:26] PR1_PRUn_GP_MUX_SEL (where n = 0 or 1) in the PRU-ICSS CFG register space needs to be set to 0h for GP mode. For a given PRU core, the following IO modes are mutually exclusive: GP mode, Sigma Delta mode, and 3 channel Peripheral I/F mode.

Table 6-81. PRU R31 (GPI) Modes

Mode	Function	Configuration
Direct input	GPI[19:0] feeds directly into the PRU R31	Default state
16-bit parallel capture	DATAIN[0:15] is captured by the posedge or negedge of CLOCKIN	• Enabled by CFG_GPCFGn register • CLOCKIN edge selected by CFG_GPCFGn register
28-bit shift in	DATAIN is sampled and shifted into a 28-bit shift register. Shift Counter (Cnt_16) feature uses … • Shift Counter (Cnt_16) feature is mapped to pru<n>_r31_status[28]. • SB (Start Bit detection) feature is mapped to pru<n>_r31_status[29]	• Enabled by CFG_GPCFGn register • Cnt_16 is self clearing and is connected to the PRU INTC • Start Bit (SB) is cleared by CFG_GPCFGn register
MII_RT	mii_rt_r31_status [29:0] internally driven by the MII_RT module	Enabled by CFG_GPCFG0/1

Table 6-82. PRU GPI Signals and Configurations

Pad Names at Device Level[1]	GPI Modes		
	Direct input	Parallel Capture	28-Bit Shift in
PR<k>_PRU<n>_GPI0	GPI0	DATAIN0	DATAIN
PR<k>_PRU<n>_GPI1	GPI1	DATAIN1	
PR<k>_PRU<n>_GPI2	GPI2	DATAIN2	
PR<k>_PRU<n>_GPI3	GPI3	DATAIN3	
PR<k>_PRU<n>_GPI4	GPI4	DATAIN4	
PR<k>_PRU<n>_GPI5	GPI5	DATAIN5	
PR<k>_PRU<n>_GPI6	GPI6	DATAIN6	
PR<k>_PRU<n>_GPI7	GPI7	DATAIN7	
PR<k>_PRU<n>_GPI8	GPI8	DATAIN8	
PR<k>_PRU<n>_GPI9	GPI9	DATAIN9	
PR<k>_PRU<n>_GPI10	GPI10	DATAIN10	
PR<k>_PRU<n>_GPI11	GPI11	DATAIN11	
PR<k>_PRU<n>_GPI12	GPI12	DATAIN12	
PR<k>_PRU<n>_GPI13	GPI13	DATAIN13	
PR<k>_PRU<n>_GPI14	GPI14	DATAIN14	
PR<k>_PRU<n>_GPI15	GPI15	DATAIN15	

(1) These pins also being used for Sigma Delta or Peripheral I/F mode.

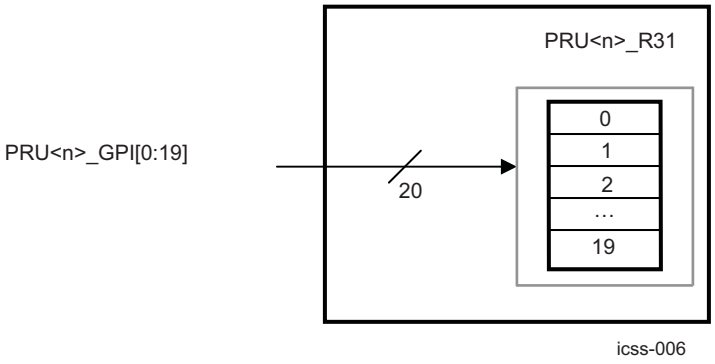
Table 6-82. PRU GPI Signals and Configurations (continued)

Pad Names at Device Level[1]	GPI Modes		
	Direct input	Parallel Capture	28-Bit Shift in
PR<k>_PRU<n>_GPI16	GPI16	CLOCKIN	
PR<k>_PRU<n>_GPI17	GPI17		
PR<k>_PRU<n>_GPI18	GPI18		
PR<k>_PRU<n>_GPI19	GPI19		

6.4.5.2.2.3.1 PRU EGPIs Direct Input

The prun_r31_status [0:19] bits of the internal PRU register file are mapped to device-level, general purpose input pins (PRUn_GPI [0:19]). In GPI Direct Input mode, PRUn_GPI [0:19] feeds directly to prun_r31_status [0:19]. Each PRU of the PRU-ICSS has a separate mapping to device input signals - PR0_PRU0_GPI[19:0] / PR1_PRU0_GPI[19:0] for the PRU-ICSS_0/ PRU-ICSS_1 PRU0 core and PR0_PRU1_GPI[19:0] / PR1_PRU1_GPI[19:0] for the PRU-ICSS_0/ PRU-ICSS_1 PRU1 core so that there are 40 total general purpose inputs to the PRU-ICSS_0/ PRU-ICSS_1 . For more details, refer also to the Section 6.4.2. See the device's system reference guide or datasheet for device specific pin mapping.

Figure 6-35. PRU R31 (EGPI) Direct Input Mode Block Diagram

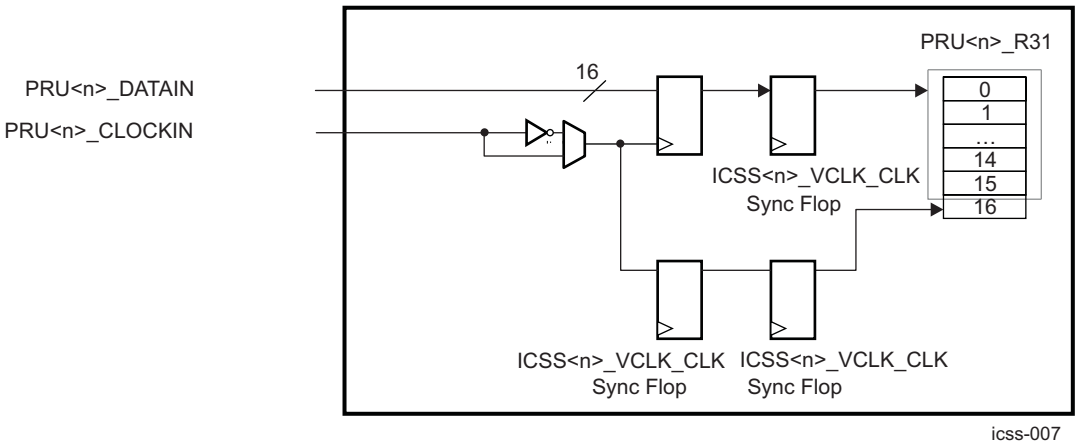


6.4.5.2.2.3.2 PRU EGPIs 16-Bit Parallel Capture

The prun_r31_status [0:15] and prun_r31_status [16] bits of the internal PRU register file mapped to device-level, general purpose input pins (PRUn_DATAIN [0:15] and PRUn_CLOCKIN, respectively). PRUn_CLOCKIN is designated for an external strobe clock, and is used to capture PRUn_DATAIN [0:15].

The PRUn_DATAIN can be captured either by the positive or the negative edge of PRUn_CLOCK, programmable through the PRU-ICSS CFG register space. If the clocking is configured through the PRU-ICSS CFG register to be positive, then it will equal PRU<n>_CLOCK; however, if the clocking is configured to be negative, then it will equal PRU<n>_CLOCK inverted.

Figure 6-36. PRU R31 (EGPI) 16-Bit Parallel Capture Mode Block Diagram



6.4.5.2.2.3.3 PRU EGPIs 28-Bit Shift In

In 28-bit shift in mode, the device-level, general-purpose input pin PRUn_DATAIN is sampled and shifted into a 28-bit shift register on an internal clock pulse. The register fills in LSB order (from bit 0 to 27) and then overflows into a bit bucket. The 28-bit register is mapped to prun_r31_status [0:27] and can be cleared in software through the PRU-ICSS CFG register space.

Note, the PRU will continually capture and shift the DATAIN input when the GPI mode has been set to 28-bit shift in.

The shift rate is controlled by the effective divisor of two cascaded dividers applied to the ICSS<n>_VCLK_CLK clock (150MHz/ 200MHz/ 225MHz). These cascaded dividers can each be configured through the PRU-ICSS CFG register space to a value of {1, 1.5, …, 16}. Table 6-83 shows sample effective clock values and the divisor values that can be used to generate these clocks.

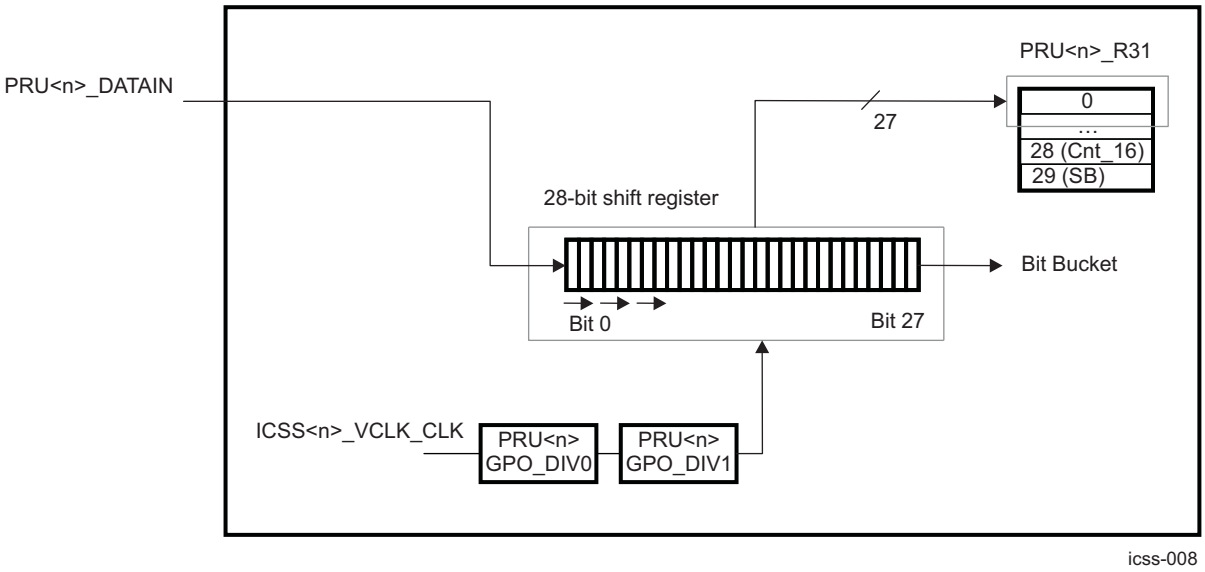
Table 6-83. PRU EGPIs Effective Clock Values

Generated clock	PRUn_GPI_DIV0	PRUn_GPI_DIV1
8-MHz	12.5 (17h)	2 (02h)
10-MHz	10 (12h)	2 (02h)
16-MHz	16 (1Eh)	1 (00h)
20-MHz	10 (12h)	1 (00h)

The 28-bit shift mode also supports the following features:

- SB (Start Bit detection) is mapped to prun_r31_status[29] and is set when the first 1 is captured on PRUn_DATAIN. The SB flag in pru<n>_r31_status[29] is cleared in software through the PRU-ICSS CFG register space.
- Cnt_16 (Shift Counter) is mapped to pru<n>_r31_status[28] and is set on every 16 shift clock samples after the Start Bit has been received. CNT_16 is self clearing and is connected to the ICSS_INTC. See the PRU-ICSS Interrupt Controller (ICSS_INTC) section for more details.

Figure 6-37. PRU R31 (EGPI) 28-Bit Shift Mode



6.4.5.2.2.3.4 General-Purpose Outputs (R30): Enhanced PRU GP Module

The PRU-ICSS implements an enhanced General Purpose Input/Output (GPIO) module that supports two general-purpose output modes: direct output and shift out.

Table 6-84 describes these modes in detail.

NOTE: Each PRU core can only be configured for one GPO mode at a time. Each mode uses the same R30 signals and internal register bits for different purposes. A summary is found in Table 6-84.

NOTE: The PRUSS_GPCFGn register, bitfield [29:26] PR1_PRUn_GP_MUX_SEL (where n = 0 or 1) in the PRU-ICSS CFG register space needs to be set to 0h for GP mode. For a given PRU core, the following IO modes are mutually exclusive: GP mode, Sigma Delta mode, and 3 channel Peripheral I/F mode.

Table 6-84. PRU R30 (EGPO) Output Mode

Mode	Function	Configuration
Direct output	pru<n>_r30[19:0] feeds directly to GPO[19:0]	Default state
Shift out	• pru<n>_r30[0] is shifted out on DATAOUT on every rising edge of pru<n>_r30[1] (CLOCKOUT). • LOAD_GPO_SH0 (Load Shadow Register 0) is mapped to pru<n>_r30[29]. • LOAD_GPO_SH1 (Load Shadow Register 1) is mapped to pru<n>_r30[30]. • ENABLE_SHIFT is mapped to pru<n>_r30[31].	Enabled by CFG_GPCFGn register

Table 6-85. GPO Mode Descriptions

Pad Names at Device Level (1)	GPO Modes	
	Direct output	Shift out
PR<k>_PRU<n>_GPO0	GPO0	DATAOUT

(1) These pins are also used for Sigma Delta or Peripheral I/F mode.

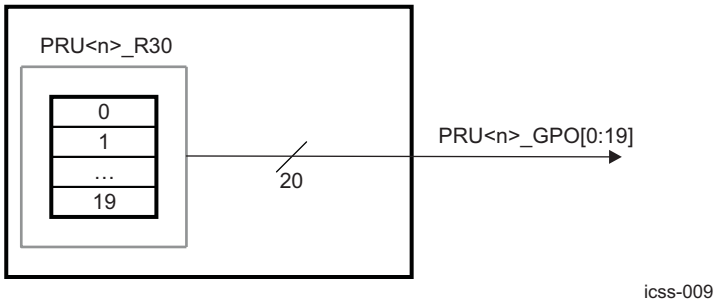
Table 6-85. GPO Mode Descriptions (continued)

Pad Names at Device Level [1]	GPO Modes	
	Direct output	Shift out
PR<k>_PRU<n>_GPO1	GPO1	CLOCKOUT
PR<k>_PRU<n>_GPO2	GPO2	
PR<k>_PRU<n>_GPO3	GPO3	
PR<k>_PRU<n>_GPO4	GPO4	
PR<k>_PRU<n>_GPO5	GPO5	
PR<k>_PRU<n>_GPO6	GPO6	
PR<k>_PRU<n>_GPO7	GPO7	
PR<k>_PRU<n>_GPO8	GPO8	
PR<k>_PRU<n>_GPO9	GPO9	
PR<k>_PRU<n>_GPO10	GPO10	
PR<k>_PRU<n>_GPO11	GPO11	
PR<k>_PRU<n>_GPO12	GPO12	
PR<k>_PRU<n>_GPO13	GPO13	
PR<k>_PRU<n>_GPO14	GPO14	
PR<k>_PRU<n>_GPO15	GPO15	
PR<k>_PRU<n>_GPO16	GPO16	
PR<k>_PRU<n>_GPO17	GPO17	
PR<k>_PRU<n>_GPO18	GPO18	
PR<k>_PRU<n>_GPO19	GPO19	

6.4.5.2.2.3.4.1 PRU EGPOs Direct Output

The prun_r30 [19:0] bits of the internal PRU register files are mapped to device-level, general-purpose output pins (PRUn_GPO[0:19]). In GPO Direct Output mode, prun_r30[0:19] feed directly to PRUn_GPO[0:19]. Each PRU of the PRU-ICSS has a separate mapping to pins, so that there are 40 total general-purpose outputs from the PRU-ICSS. See the device's system reference guide or datasheet for device-specific pin mapping.

Figure 6-38. PRU R30 (EGPO) Direct Output Mode Block Diagram



NOTE: R30 is not initialized after reset. To avoid unintended output signals, R30 should be initialized before pinmux configuration of PRU signals.

6.4.5.2.2.3.4.2 PRU EGPO Shift Out

In shift out mode, data is shifted out of prun_r30[0] (PRUn_DATAOUT) on every rising edge of prun_r30[1] (PRUn_CLOCK). The shift rate is controlled by the effective divisor of two cascaded dividers applied to the ICSS<n>_VCLK_CLK clock (150MHz/ 200MHz/ 225MHz). These cascaded dividers can each be configured through the PRU-ICSS CFG register space to a value of {1, 1.5, …, 16}. Table 6-86 shows sample effective clock values and the divisor values that can be used to generate these clocks. Note that PRUn_CLOCKOUT is a free-running clock that starts when the PRU GPO mode is set to shift out mode.

Table 6-86. Effective Clock Values

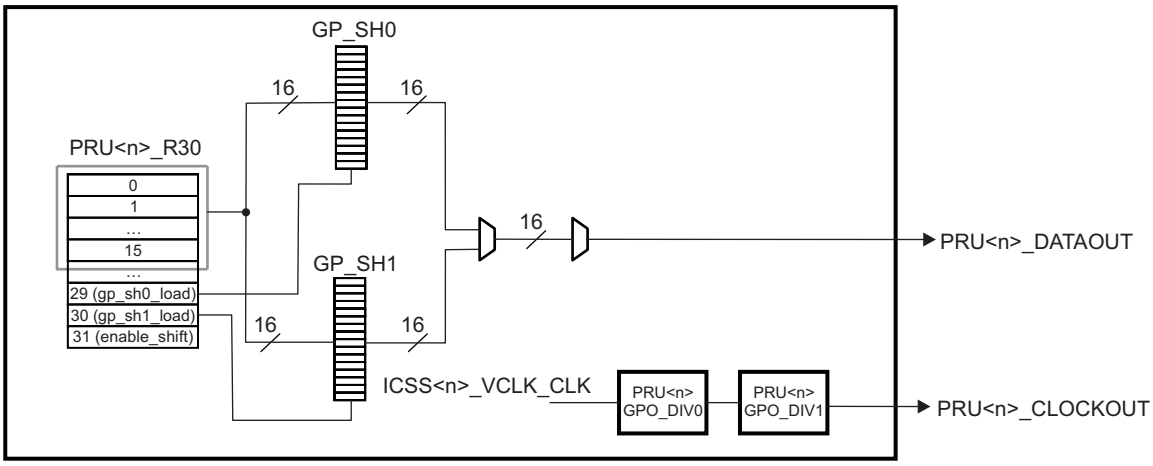
Generated Clock	PRUn_GPO_DIV0	PRUn_GPO_DIV1
8 MHz	12.5 (17h)	2 (02h)
10 MHz	10 (12h)	2 (02h)
16 MHz	16 (1Eh)	1 (00h)
20 MHz	10 (12h)	1 (00h)

Shift out mode uses two 16-bit shadow registers (gpo_sh0 and gpo_sh1) to support ping-pong buffers. Each shadow register has independent load controls programmable through prun_r30[29:30] (PRUn_LOAD_GPO_SH [0:1]). While PRUn_LOAD_GPO_SH [0/1] is set, the contents of prun_r30[0:15] are loaded into gpo_sh0/1.

NOTE: If any device-level pins mapped to prun_r30[2:15] are configured for the prun_r30 [2:15] pinmux mode, then these pins will reflect the shadow register value written to prun_r30. Any pin configured for a different pinmux setting will not reflect the shadow register value written to prun_r30.

The data shift will start from the LSB of gpo_sh0 when prun_r30[31] (PRUn_ENABLE_SHIFT) is set. Note that if no new data is loaded into gpo_shnn after shift operation, the shift operation will continue looping and shifting out the pre-loaded data. When PRUn_ENABLE_SHIFT is cleared, the shift operation will finish shifting out the current shadow register, stop, and then reset.

Figure 6-39. PRU R30 (GPO) Shift Out Mode Block Diagram



icss-010

Follow these steps to use the GPO shift out mode:

Step One: Initialization

1. Load 16-bits of data into gpo_sh0:
   (a) Set R30[29] = 1 (PRUn_LOAD_GPO_SH0)
   (b) Load data in R30[15:0]
   (c) Clear R30[29] to turn off load controller

2. Load 16-bits of data into gpo_sh1:
   (a) Set R30[30] = 1 (PRUn_LOAD_GPO_SH1)
   (b) Load data in R30[15:0]
   (c) Clear R30[30] to turn off load controller
3. Start shift operation:
   (a) Set R30[31] = 1 (PRUn_ENABLE_SHIFT)

Step 2: Shift Loop

1. Monitor when a shadow register has finished shifting out data and can be loaded with new data:
   (a) Poll PRUn_GPI_SH_SEL bit of the GPCFGn register
   (b) Load new 16-bits of data into gpo_sh0 if PRUn_GPI_SH_SEL = 1
   (c) Load new 16-bits of data into gpo_sh1 if PRUn_GPI_SH_SEL = 0
2. If more data to be shifted out, loop to Shift Loop
3. If no more data, exit loop

Step 3: Exit

1. End shift operation:
   (a) Clear R30[31] to turn off shift operation

NOTE: Until the shift operation is disabled, the shift loop will continue looping and shifting out the pre-loaded data if no new data has been loaded into gpo_shn.

6.4.5.2.2.3.5 Sigma Delta (SD) Decimation Filtering H/W

Sigma-delta Sinc filtering is achieved by the combination of PRU hardware and firmware. PRU hardware provides hardware integrators that do the accumulation part of Sinc filtering, while the differentiation part is done in firmware. The integrator serves to count the number of 1's per clock event. Each channel has three cascaded counters, which are the accumulators for the Sinc3 filter. Each counter is 24 bits, giving a maximum count of 16,777,215. Each channel has a free running rollover clock counter. This sample counter updates the count value on the effective clock event for that channel. Each channel also contains a programmable counter compare block, and the compare register has a size of 8 bits. However, the minimum value is 4 and maximum value is 202 due to the 24-bit accumulator. Once sample counter compare value is reached, the shadow register copy is updated and the shadow register copy flag is set.

Features of the integrators in PRUs SD Demodulator:

- Up to nine channels concurrent counting
- Flexible clock source configuration for each channel; option of independent clock source for each channel or one clock source for three channels
- Programmable, 8-bit sample counter compare register; used to set the OSR of Sinc filter
- Three 24-bit cascaded counters per channel for accumulation, only Sinc3 and Sinc2 modes supported
- Common channel enable (all channels are active or none are active)

6.4.5.2.2.3.5.1 Block Diagram and Signals

The Sigma Delta's I/Os are multiplexed with the PRU GPI/GPO signals, as shown in Table 6-87. Note the PR<k>_PRU<n>_GP_MUX_SEL bitfield in the PRUSS_GPCFG0/1 register must be set to 3h for configure the GPI/GPO signals for SD mode.

Table 6-87. PRU GPI Signals and Configurations for Sigma Delta

Signal Names at Device Level [1]	Sigma Delta (SD) Mode (PRUSS_GPCFGn[29:26] PR1_PRUn_GP_MUX_SEL = 3h)	
PR<k>_PRU<n>_GPI0	SD0_CLK	SD demodulator clock channel 0

(1) Note these signals are shared with the GP and Peripheral I/Fs. To configure for Sigma Delta, PRUSS_GPCFG0/1[29-26] PR1_PRUn_GP_MUX_SEL (where n = 0 or 1) needs to be set to 3h for SD mode.

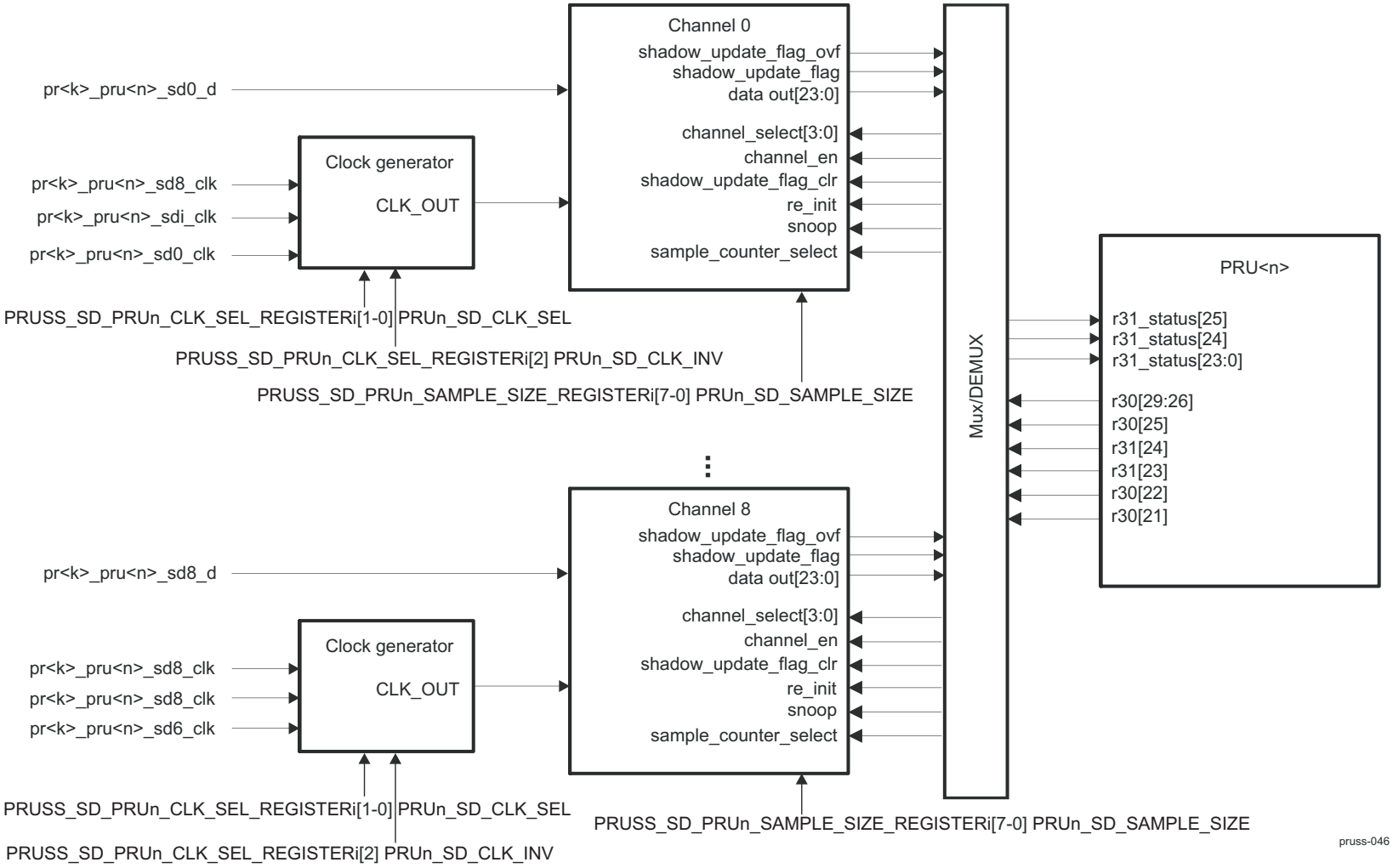
Table 6-87. PRU GPI Signals and Configurations for Sigma Delta (continued)

Signal Names at Device Level [1]	Sigma Delta (SD) Mode (PRUSS_GPCFGn[29:26] PR1_PRUn_GP_MUX_SEL = 3h)	
PR<k>_PRU<n>_GPI1	SD0_D	SD demodulator data channel 0
PR<k>_PRU<n>_GPI2	SD1_CLK	SD demodulator clock channel 1
PR<k>_PRU<n>_GPI3	SD1_D	SD demodulator data channel 1
PR<k>_PRU<n>_GPI4	SD2_CLK	SD demodulator clock channel 2
PR<k>_PRU<n>_GPI5	SD2_D	SD demodulator data channel 2
PR<k>_PRU<n>_GPI6	SD3_CLK	SD demodulator clock channel 3
PR<k>_PRU<n>_GPI7	SD3_D	SD demodulator data channel 3
PR<k>_PRU<n>_GPI8	SD4_CLK	SD demodulator clock channel 4
PR<k>_PRU<n>_GPI9	SD4_D	SD demodulator data channel 4
PR<k>_PRU<n>_GPI10	SD5_CLK	SD demodulator clock channel 5
PR<k>_PRU<n>_GPI11	SD5_D	SD demodulator data channel 5
PR<k>_PRU<n>_GPI12	SD6_CLK	SD demodulator clock channel 6
PR<k>_PRU<n>_GPI13	SD6_D	SD demodulator data channel 6
PR<k>_PRU<n>_GPI14	SD7_CLK	SD demodulator clock channel 7
PR<k>_PRU<n>_GPI15	SD7_D	SD demodulator data channel 7
PR<k>_PRU<n>_GPI16	SD8_CLK	SD demodulator clock channel 8
PR<k>_PRU<n>_GPI17	SD8_D	SD demodulator data channel 8
PR<k>_PRU<n>_GPI18		
PR<k>_PRU<n>_GPI19		

The PR<k>_PRU0_GPO1 signal (muxed with SD0_D) can be used as SD_CLKOUT when PRU-ICSS generates clock. This is a trade-off as PRU application will lose one SD channel. SD_CLKOUT needs to go through a clock generator chip if driving multiple sigma delta modulators and also be looped back into PRU-ICSS as SD_CLKIN, typically pru_gpi16.

Note to output the SD clock on PR<k>_PRU0_GPO1, this device requires that the PRU core be configured for both SD and shift out mode (PRUSS_GPCFG0/1[29-26] PR1_PRUn_GP_MUX_SEL = 3h and PRUSS_GPCFG0/1[14] PRU<n>_GPO_MODE = 1h). Be sure to configure the shift out mode's clock divisors before enabling shift out mode (PRUSS_GPCFG0/1[14] PRU<n>_GPO_MODE = 1h). Additionally, the PRU-ICSS_0 PRU0 SD clock is routed to both PR0_PRU0_GPO1 and PR0_PRU1_GPO1. Figure 6-40 shows a block diagram of the Sigma Delta implementation. Full description of the PRU R30 and R31 registers are shown in Table 6-89 and Table 6-90.

Figure 6-40. Sigma Delta Block Diagram



Note each channel can independently be configured to use one of three external clock sources. Table 6-88 shows the clock source options, selectable through PRUSS_SD_PRUn_CLK_SEL_REGISTERi[1:0] PRUn_SD_CLK_SEL (where n = 0 or 1).

Table 6-88. External Clock Sources

PRUn_SD_CLK_SEL value	Clock Source
0	pr<k>_pru<n>_sd8_clk
1	pr<k>_pru<n>_sd<m>_clk
2	pr<k>_pru<n>_sd0_clk for sd0, sd1, and sd2; pr<k>_pru<n>_sd3_clk for sd3, sd4, and sd5; pr<k>_pru<n>_sd6_clk for sd6, sd7, and sd8

6.4.5.2.2.3.5.2 PRU R30 / R31 Interface

The PRU uses the R30 and R31 registers to interface with the Sigma Delta interface. Tables Table 6-89 and Table 6-90 shows the R31 and R30 interface for the Sigma Delta mode. Note that only the parameters and data for one channel can be viewed at a time. The channel to be viewed is determined by the r30[29-26] (channel_select).

Table 6-89. Sigma Delta PRU Registers: R31

Bits	Field Name	Description
29:26	Reserved	
25	shadow_update_flag_ovf / shadow_update_flag_ovf_clr	Shadow update flag overflow, set when over sample count equals over sample size and shadow_update_flag is still set. Set this bit to clear the flag.
24	shadow_update_flag / shadow_update_flag_clr	Shadow update flag, set when over sample count equals over sample size and shadow_update_flag is still set. Set this bit to clear the flag.
23	re_init/data_out[23]	re_init (write): Set to reset all counters, flags, and shadow copy. Updates over_sample_size based on the current PRUSS_SD_PRUn_SAMPLE_SIZE_REGISTERi register (where n = 0 or 1 and i = 0 to 8) on the selected channel. data_out[23](read): most-significant bit of sample data
22:0	data_out[22:0]	Selected sample data excluding most-significant bit

Table 6-90. Sigma Delta PRU Registers: R30

Bits	Field Name	Description
31:30	Reserved	
29:26	channel_select[3:0]	Channel select 0h: Channel 0 ... 8h: Channel 8 9h: Reserved ... Fh: Reserved
25	channel_en	Global Channel enable (effects all 9 channels). 0h: All channels disabled. Counters/flags are cleared. 1h: All channels enabled.
24:23	Reserved	
22	snoop	Enable snoop (i.e. fetch data) on the selected channel. 0h: acc2/acc3 shadow copy 1h: current acc2/acc3
21	sample_counter_select	Read sample counter. 0h: Not selected 1h: Sample count selected
20:0	Reserved	

The PRU-ICSS CFG register space has additional registers for controlling the SD demodulator module:

- PRUSS_SD_PRUn_CLK_SEL_REGISTERi[4] PRUn_SD_ACC2_SEL (where n = 0 or 1, i = 0 to 8) - Selects accumulator 2 as source.
- PRUSS_SD_PRUn_CLK_SEL_REGISTERi[2] PRUn_SD_CLK_INV (where n = 0 or 1, i = 0 to 8) - Inverts clock.

- PRUSS_SD_PRUn_CLK_SEL_REGISTERi[1:0] PRUn_SD_CLK_SEL (where n = 0 or 1, i = 0 to 8) - Selects the clock source.
- PRUSS_SD_PRUn_SAMPLE_SIZE_REGISTERi[7:0] PRUn_SD_SAMPLE_SIZE (where n = 0 or 1, i = 0 to 8) - Selects number of samples to read before giving output.

6.4.5.2.2.3.5.3 Sigma Delta Description

Figure 6-41 shows a block diagram of the Sigma Delta hardware integrators and integration with the PRU R30 / R31 interface for a single channel.

Figure 6-41. Sigma Delta Hardware Integrators Block Diagram (snoop = 0)

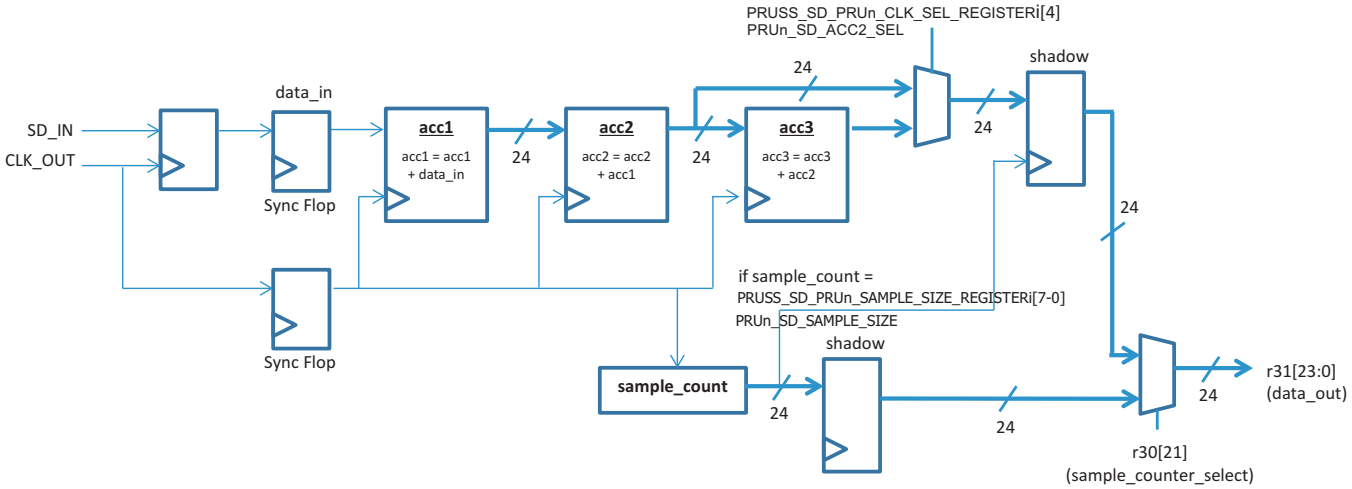
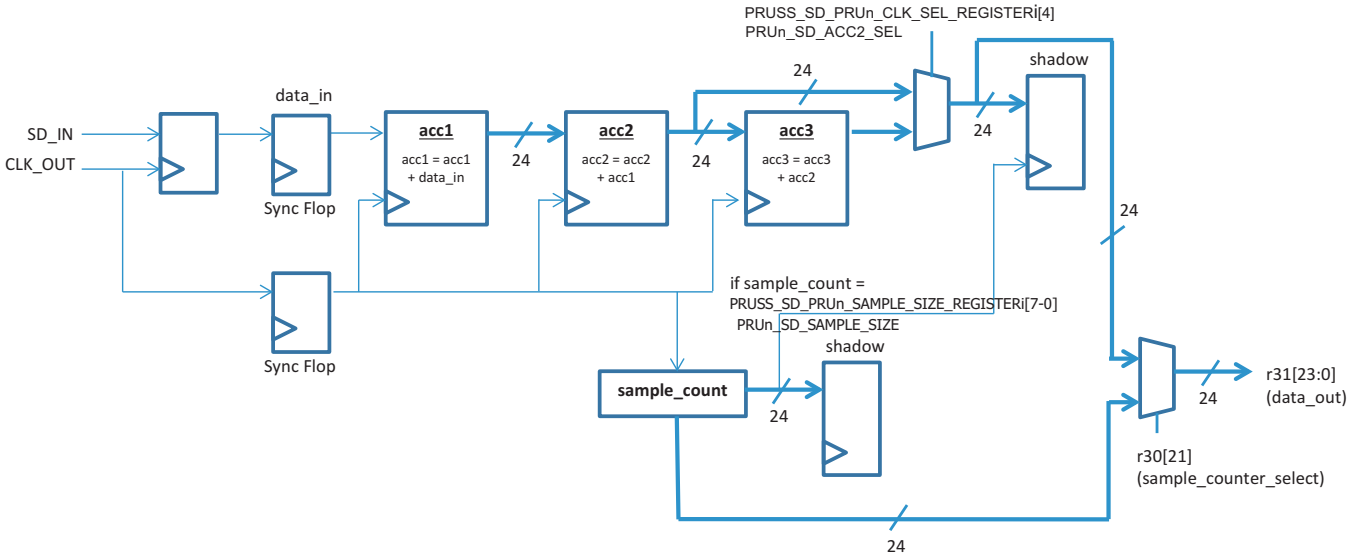


Figure 6-42. Sigma Delta Hardware Integrators Block Diagram (snoop = 1)



The three accumulators (acc1-acc3) for each channel are simple 24 bit adders. The input for acc1 is 1-bit, while the inputs for acc2 and acc3 are 24-bits. On each positive edge of the CLK_OUT, all three 24-bit counters (acc1-acc3) and the sample counter for each channel will get updated as follows:

```
acc1 = acc1 + data_in
acc2 = acc2 + acc1
acc3 = acc3 + acc2
sample_count = sample_count + 1
```

Each accumulator will rollover at 0xFF_FFFF. For example if acc2 = 0x10 and acc3 = 0xFF_FFFF, then acc3 will update to 0x00_0000F on the next clock event. Sample_count will rollover when it equals the defined sample size (PRUSS_SD_PRUn_SAMPLE_SIZE_REGISTERi[7-0] PRUn_SD_SAMPLE_SIZE).

Note that while the channels are not enabled, no operations are performed and all flags and counters are cleared. If a new sample size is to be loaded, the PRU firmware should assert re_init (r31[23]), and all stored count values are cleared to 0.

The Sigma Delta interface has two status flags:

- Shadow update flag (r31[24])
- Shadow update flag overflow (r31[25])

When sample_count equals the defined sample size (PRUSS_SD_PRUn_SAMPLE_SIZE_REGISTERi[7-0] PRUn_SD_SAMPLE_SIZE), then the acc2/acc3 shadow register copy will be updated, the shadow_update_flag (r31[24]) will be set, and sample_count will rollover to 0. The PRU firmware can clear this flag by writing '1' to shadow_update_flag_clr (r31[24]). If sample_count equals the defined sample size and the shadow_update_flag is still set, then shadow_update_flag_ovf (r31[25]) will be set. Similarly, the PRU firmware can clear this flag by writing '1' to shadow_update_flag_ovf_clr (r31[25]). Note that the clear operation for both flags has a higher priority than the set event.

The PRU firmware can monitor the acc2/acc3 and sample_count values through data_out[23:0] (r31[23:0]). Table 11-162 shows the configuration options for data_out[23:0].

Table 6-91. Data_out[23:0] Configuration Options

snoop (r30[22])	sample_counter_select (r30[21])	data_out (r31[23:0])
0	0	Reads acc2/acc3 shadow register copy. See Figure 6-41*Sigma Delta Hardware Integrators Block Diagram (snoop = 0).*
1	0	Reads acc2/acc3 directly. See Figure 6-42*Sigma Delta Hardware Integrators Block Diagram (snoop = 1).*
0	1	Reads sample_count shadow register copy. See Figure 6-41*Sigma Delta Hardware Integrators Block Diagram (snoop = 0).*
1	1	Reads sample_count directly. See Figure 6-42*Sigma Delta Hardware Integrators Block Diagram (snoop = 1).*

6.4.5.2.2.3.5.4 Basic Programming Example

The following programming example assumes that the PRU is configured for Sigma Delta Mode (PRUSS_GPCFG0/1[29-26] PR1_PRU<n>_GP_MUX_SEL = 3h).

1. Configure clock sources, accumulator source, and sample size:
   a. PRUSS_SD_PRUn_CLK_SEL_REGISTERi[1-0] PRUn_SD_CLK_SEL for clock source
   b. PRUSS_SD_PRUn_CLK_SEL_REGISTERi[2] PRUn_SD_CLK_INV for clock polarity
   c. PRUSS_SD_PRUn_CLK_SEL_REGISTERi[4] PRUn_SD_ACC2_SEL for accumulator source
   d. PRUSS_SD_PRUn_SAMPLE_SIZE_REGISTERi[7-0] PRUn_SD_SAMPLE_SIZE for sample size
2. Reinitialize all channels whose sample size was configured
   a. Select channel by writing to channel_select (r30[29-26])
   b. Delay at least 1 PRU cycle before executing re_int in step 2c.
   c. Reinitialize selected channel by writing to re_init (r31[23])
   d. Repeat steps 2a & 2b for all configured channels
3. Enable all channels by writing '1' to channel_en (r30[25])
4. Select channel by writing to channel_select (r30[29-26])
   a. Poll shadow_update_flag (r31[24]) to detect when acc2/acc3 shadow register copy data is ready to be ready
   b. Delay at least 1 PRU cycle before polling shadow_update_flag in Step 4c.

c. Read data_out[23:0] (r31[23:0])
d. Clear shadow_update_flag by writing '1' to r31[24]

5. Repeat step 4 for new channel

6.4.5.3 3 Channel Peripheral Interface

The 3 channel Peripheral Interface supports functionality for operations utilizing the EnDat 2.2 and BiSS protocols.

This module supports the following features:

- 3 channels with baud range from 100 kHz to 16 MHz
- ICSS_n_UART_CLK (default) or ICSS_n_VCLK_CLK master clock is an input to independent div16fr clock dividers to produce a 1X clock (ENDAT<m>_CLK) and oversampling clock
- Half-duplex (TX and RX are not supported concurrently)
- TX FIFO size of 32 bits
- RX FIFO size of 32 bits
- Configurable shift size/oversampling on RX
- Optional RX frame size auto shut off
- Programmable HW delay 1 (wire delay, controlling when the clock signal is first driven low) and delay 2 (tst delay, controlling when the clock signal is first driven high) on TX operation
- Optional programmable TX termination
- Individual TX channel start trigger (tx_channel_go) or simultaneous TX start trigger for all channels (tx_global_go)
- Flexible HW assisted clock output generation to allow free running, stop high and stop low (after last RX data), or stop high (after last TX data) operation with optional software clock override feature
- Optional SW direct snoop of data input

6.4.5.3.1 Block Diagram and Signal Configuration

The Peripheral Interface's I/Os are multiplexed with the PRU GPI/GPO signals, as shown in Table 6-92. The PR<k>_PRU<n>_GP_MUX_SEL bitfield in the PRUSS_GPCFG0/1 register must be set to 1h for configure the GPI/GPO signals for Peripheral I/F mode.

Table 6-92. PRU GPI/GPO Signals and Configurations for Peripheral I/F[1]

Pad Names at Device Level[2][3]	Peripheral I/F Mode (PRUSS_GPCFG0/1[29:26] PR1_PRUn_GP_MUX_SEL = 1h)
PR<k>_PRU<n>_GPI0	
PR<k>_PRU<n>_GPI1	
PR<k>_PRU<n>_GPI2	
PR<k>_PRU<n>_GPI3	
PR<k>_PRU<n>_GPI4	
PR<k>_PRU<n>_GPI5	
PR<k>_PRU<n>_GPI6	
PR<k>_PRU<n>_GPI7	
PR<k>_PRU<n>_GPI8	
PR<k>_PRU<n>_GPI9	ENDAT0_IN
PR<k>_PRU<n>_GPI10	ENDAT1_IN
PR<k>_PRU<n>_GPI11	ENDAT2_IN

(1) Usage of the Peripheral Interface signals are not restricted to only ENDAT interfaces.
(2) Note these signals are shared with the GP, MII and Sigma Delta modes. To configure for Peripheral I/F, PRUSS_GPCFG0/1[29:26] PR1_PRUn_GP_MUX_SEL needs to be set to 1h.
(3) Some devices may not pin out all 29 bits of R31 and all 32 bits of R30. For which pins are available on this device, see Section 6.4.2*PRU-ICSS Environment*. See the device Data Manual for device-specific pin mapping.

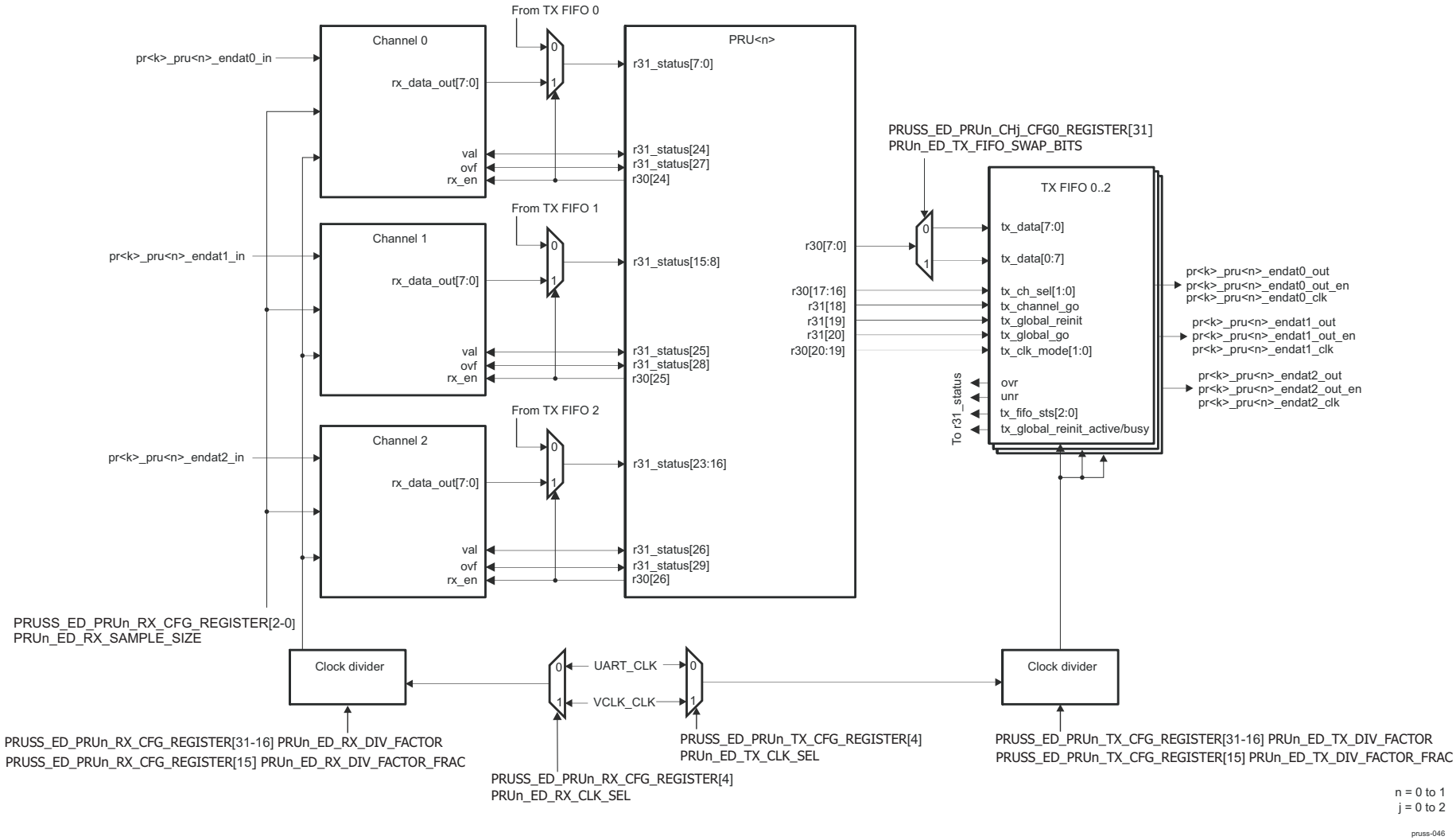
Table 6-92. PRU GPI/GPO Signals and Configurations for Peripheral I/F[1] (continued)

Pad Names at Device Level[2][3]	Peripheral I/F Mode (PRUSS_GPCFG0/1[29:26] PR1_PRUn_GP_MUX_SEL = 1h)
PR<k>_PRU<n>_GPI12	
PR<k>_PRU<n>_GPI13	
PR<k>_PRU<n>_GPI14	
PR<k>_PRU<n>_GPI15	
PR<k>_PRU<n>_GPI16	
PR<k>_PRU<n>_GPI17	
PR<k>_PRU<n>_GPI18	
PR<k>_PRU<n>_GPI19	
PR<k>_PRU<n>_GPO0	ENDAT0_CLK
PR<k>_PRU<n>_GPO1	ENDAT0_OUT
PR<k>_PRU<n>_GPO2	ENDAT0_OUT_EN
PR<k>_PRU<n>_GPO3	ENDAT1_CLK
PR<k>_PRU<n>_GPO4	ENDAT1_OUT
PR<k>_PRU<n>_GPO5	ENDAT1_OUT_EN
PR<k>_PRU<n>_GPO6	ENDAT2_CLK
PR<k>_PRU<n>_GPO7	ENDAT2_OUT
PR<k>_PRU<n>_GPO8	ENDAT2_OUT_EN
PR<k>_PRU<n>_GPO9	
PR<k>_PRU<n>_GPO10	
PR<k>_PRU<n>_GPO11	
PR<k>_PRU<n>_GPO12	
PR<k>_PRU<n>_GPO13	
PR<k>_PRU<n>_GPO14	
PR<k>_PRU<n>_GPO15	
PR<k>_PRU<n>_GPO16	
PR<k>_PRU<n>_GPO17	
PR<k>_PRU<n>_GPO18	
PR<k>_PRU<n>_GPO19	

A block diagram for the Peripheral I/F is included in Figure 6-43. As shown, each channel is composed of four I/Os:

- ENDAT<m>_IN - RX input data
- ENDAT<m>_CLK - Clock (CLK_OUT) generated by the 1x (or TX) clock. The default value is 1.
- ENDAT<m>_OUT - TX output data. The default value is 0.
- ENDAT<m>_OUT_EN - TX enable output (1 = TX mode, 0 = RX mode). The default value is 0. Note this signal is auto controlled by hardware.

Figure 6-43. Peripheral I/F Block Diagram



6.4.5.3.2 PRU R30 and R31 Interface

The PRU uses the R30 and R31 registers to interface with the Peripheral I/F. Table 6-93 shows the R31 and R30 interface for the Peripheral I/F RX mode, and Table 6-94 shows the comparable interface for the TX mode.

Table 6-93. Peripheral I/F RX

Register	Bits	Field name	Description
R31	31:30	Reserved	PRU Host Interrupts 1/0 from local INTC
	29	ovf2	Overflow Flag for Channel 2. Write 1 to clear.
	28	ovf1	Overflow Flag for Channel 1. Write 1 to clear.
	27	ovf0	Overflow Flag for Channel 0. Write 1 to clear.
	26	val2	Valid Flag for Channel 2. Write 1 to clear.
	25	val1	Valid Flag for Channel 1. Write 1 to clear.
	24	val0	Valid Flag for Channel 0. Write 1 to clear.
	23:16	rx_data_out2	Oversampled Data Output for Channel 2. Note these bits are shared with the TX Interface. When TX_FIFO has stopped transmission, RX data will be selected.
	15:8	rx_data_out1	Oversampled Data Output for Channel 1. Note these bits are shared with the TX Interface. When TX_FIFO has stopped transmission, RX data will be selected.
	7:0	rx_data_out0	Oversampled Data Output for Channel 0. Note these bits are shared with the TX Interface. When TX_FIFO has stopped transmission, RX data will be selected.
R30	31:27	Reserved	
	26	rx_en2	RX Enable for Channel 2. 0h: Channel not enabled, all counters/flags will get reset 1h: Channel is enabled
	25	rx_en1	RX Enable for Channel 1. 0h: Channel not enabled, all counters/flags will get reset 1h: Channel is enabled
	24	rx_en0	RX Enable for Channel 0. 0h: Channel not enabled, all counters/flags will get reset 1h: Channel is enabled
	23:0	Reserved	

Table 6-94. Peripheral I/F TX

Register	Bits	Field name	Description
R31	31:30	Reserved	
	29:22	Reserved	
	21	tx_global_reinit_active/ busy2	Tx_global_reinit action has some latency do to clocking. This status shows if action is completed. 1h: Active 0h: Done For non reinit case, this bit states that last bit is on tx wire. It does not mean the clock is off. 1h: Last bit is not done 0h: Last bit on tx wire Note that by using rx auto arm feature, the observation is lost at rx enable. This can be used to determine when to enable rx during non-auto arm case.
	20	tx_global_go	TX global start of all channels. Note: FIFO must not be empty. If empty, transmit will not start.
	19	tx_global_reinit	Reinit all channels into default mode. This clears all flags and state machines for all channels. Note: Sequence should be assert tx_global_reinit then de-assert rx_en. This will ensure TX and RX are in reset/default state. User must assert this after the frame has been sent and TX is not busy.
	18	tx_channel_go	TX start the channel transmit (selected by tx_ch_sel). Note: FIFO must not be empty.
	17	unr2	Under Run Flag for Channel 2. This flag is only set when the tx_frame_count is nonzero and FIFO is empty at time to send data.

Table 6-94. Peripheral I/F TX (continued)

Register	Bits	Field name	Description
	16	ovr2	Over Run Flag for Channel 2
	15:14	Reserved	
	13	tx_global_reinit_active/ busy1	Tx_global_reinit action has some latency do to clocking. This status shows if action is completed. 1h: Active 0h: Done For non reinit case, this bit states that last bit is on tx wire. It does not mean the clock is off. 1h: Last bit is not done 0h: Last bit on tx wire Note that by using rx auto arm feature, the observation is lost at rx enable. This can be used to determine when to enable rx during non-auto arm case.
	12:10	tx_fifo_sts1	TX FIFO occupancy status for Channel 1. 0 :0 Empty 1h: 1 word 2h: 2 words 3h: 3 words 4h: Full 5h-7h: Reserved
	9	unr1	Under Run Flag for Channel 1. This flag is only set when the tx_frame_count is nonzero and FIFO is empty at time to send data.
	8	ovr1	Over Run Flag for Channel 1
	7:6	Reserved	
	5	tx_global_reinit_active/ busy0	Tx_global_reinit action has some latency do to clocking. This status shows if action is completed. 1h: Active 0h: Done For non reinit case, this bit states that last bit is on tx wire. It does not mean the clock is off. 1h: Last bit is not done 0h: Last bit on tx wire Note that by using rx auto arm feature, the observation is lost at rx enable. This can be used to determine when to enable rx during non-auto arm case.
	4:2	tx_fifo_sts0	TX FIFO occupancy status for Channel 0. 0h: Empty 1h: 1 word 2h: 2 words 3h: 3 words 4h: Full 5h-7h: Reserved
	1	unr0	Under Run Flag for Channel 0. This flag is only set when the tx_frame_count is nonzero and FIFO is empty at time to send data.
	0	ovr0	Over Run Flag for Channel 0
R30	31:21	Reserved	
	20:19	clk_mode	CLK_OUT mode. 0h: Free-running/stop-low. Clock will remain free-running until the receive module has received the number of bits indicated in rx_frame_counter and then the clock will stop low. 1h: Free-running/stop-high (default). Clock will remain free-running until the receive module has received the number of bits indicated in rx_frame_counter and then the clock will stop high. Note this is the default/reset state, and a hardware reset or reinit will return clk_mode to this state. Note the initial state of the clock will be high, but the clock will not start until TX GO event. 2h: Free-run. NOTE: You must do a reinit to get out of this clock mode then you can update clk_mode to a different mode. Also if you do multiple TX GO, the 2nd go should have tst_delay and wire_delay zero since the clock is free running after the first go. 3h: Stop high after transmit. Clock will run until the last TX bit is sent and stops high.
	18	Reserved	

Table 6-94. Peripheral I/F TX (continued)

Register	Bits	Field name	Description
	17:16	tx_ch_sel	TX channel select. 0h: Channel 0 1h: Channel 1 2h: Channel 2 3h: Reserved
	15:9	Reserved	
	7:0	tx_data	TX data for FIFO. Notes: FIFO transmits MSB first and is 32-bits deep. TX_FIFO_SWAP_BITS bit in the PRU-ICSS CFG register space can be used to flip the load order of bits. The FIFO has 2 modes of operation: 1. Preload and Go. This should be done for EnDAT and frames less than 32-bits. 2. Continuous mode. This should be done for frames bigger than 32-bits. In continuous mode, software needs to keep up with the line rate and ensure that the FIFO is never empty. When the FIFO is at 2 byte level, software needs to load the next 2 bytes. If software waits till the end of the empty state, it is possible to get the TX into a bad state. The FIFO state can be recovered via re-init.

NOTE: The PRU-ICSS CFG register space has additional registers for controlling the Peripheral I/F module.

6.4.5.3.3 Clock Generation

6.4.5.3.3.1 Configuration

The Peripheral I/F module has two source clock options, ICSS_n_UART_CLK (default) and ICSS_n_VCLK_CLK. There are two independent clock dividers (div16) for the 1x and oversampling (OS) clocks, and each clock divider is configurable by two cascading dividers:

- [31-16]PRUn_ED_TX_DIV_FACTOR and [15]PRUn_ED_TX_DIV_FACTOR_FRAC for the 1x clock
- [31-16]PRUn_ED_RX_DIV_FACTOR and [15]PRUn_ED_RX_DIV_FACTOR_FRAC for the OS clock

The 1x clock is output on the ENDAT<m>_CLK signal. In TX mode, the output data is read from the TX FIFO at this 1x clock rate. The default value of this clock is high and the start and stop conditions for this clock are described in Section 6.4.5.3.3.2 *Clock Output Start Conditions* and Section 6.4.5.3.3.3 *Stop Conditions*.

In RX mode, the input data is sampled at the OS clock rate. Note the OS clock rate divided by the 1x clock rate must equal [2-0]PRUn_ED_RX_SAMPLE_SIZE.

Example clock rates and divisor values relative to the 192-MHz ICSS_n_UART_CLK source are shown in Table 6-95.

Table 6-95. Clock Rate Examples for 192-MHz PRUSSn_UART_GFCLK Clock Source

TX_DIV_FACTOR	1x Clock	RX_DIV_FACTOR	RX_DIV_FACTOR_FRAC	OS Clock	Oversample Factor
12	16 MHz	1	1.5	128 MHz	8x
16	12 MHz	2	1	96 MHz	8x
24	8 MHz	3	1	64 MHz	8x
32	6 MHz	4	1	48 MHz	8x
48	4 MHz	6	1	32 MHz	8x
96	2 MHz	12	1	16 MHz	8x
192	1 MHz	24	1	8 MHz	8x

6.4.5.3.3.2 Clock Output Start Conditions

This section describes the configurable start conditions for the ENDAT<m>_CLK. The software can completely control via PRUSS_ED_PRUn_CHj_CFG0_REGISTER when bit [29]PRUn_ED_CLK_OUT_OVR_EN = 1h. By default however, the PRU hardware will control the clocks as described in the following sections.

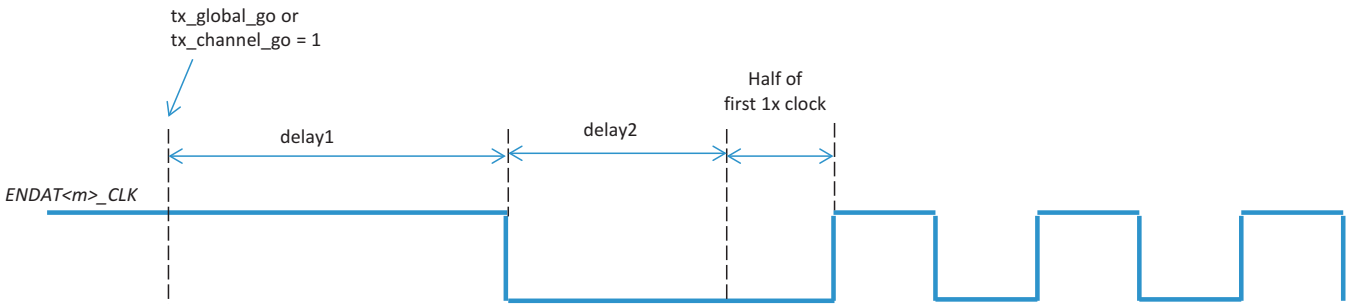
6.4.5.3.3.2.1 TX Mode (RX_EN = 0)

In TX mode, the ENDAT<m>_CLK begins after the firmware loads the TX FIFO and sets either r30[20] (tx_global_go) or r30[17:16] (tx_channel_go) to 1h. After the “go” bit is set, the delay1 (wire delay) compensation counter for each channel begins. After delay1 is complete, ENDAT<m>_CLK is driven low and then the delay2 (tst) counter begins. After the delay2 counter expires, the ENDAT<m>>_CLK starts running (first low and then high). Therefore, first rising edge of ENDAT<m>_CLK (measured from the go bit) = delay1 (tx wire delay) + delay2 (tst_counter delay) + half of the 1x clock frequency (since the clock starts low).

Figure 6-44 shows the start condition for TX mode. As shown in the figure, the default value of clock is high. The PRU-ICSS CFG register space has additional registers for controlling the TX start timing delay values:

- Delay 1: PRUSS_ED_PRUn_CHj_CFG0_REGISTER[10-0] PRUn_ED_TX_WDLY
- Delay 2: PRUSS_ED_PRUn_CHj_CFG0_REGISTER[15-0] PRUn_ED_TST_DELAY_COUNTER

Figure 6-44. TX Mode Start Condition



6.4.5.3.3.2.2 RX Mode (RX_EN = 1)

In RX mode, the ENDAT<m>_CLK will start running whenever the RX_EN is set. Note that the PRU firmware in this mode is responsible for any delay conditions.

The hardware can also auto-enable RX mode at the end of a TX transaction. The PRUSS_ED_PRUn_CHj_CFG1_REGISTER[31-16] PRUn_ED_RX_EN_COUNTER is used to program a delay between the last TX bit sent and when the RX_EN is set.

6.4.5.3.3.3 Stop Conditions

The r30[20:19] (clk_mode[1:0]) value determines the stop condition for ENDAT<m>_CLK. There are 4 options available:

clk_mode_value	Description
0	Stop low on last RX frame
1	Stop high on last RX frame
2	Run continuously
3	Stop high on last TX bit

The last RX frame is configured by PRUSS_ED_PRUn_CHj_CFG0_REGISTER[27-16] PRUn_ED_RX_FRAME_SIZE, and the last TX bit is configured by PRUSS_ED_PRUn_CHj_CFG0_REGISTER[15-11] PRUn_ED_TX_FRAME_SIZE. Each stop condition is shown in Figure 6-45 through Figure 6-48.

Figure 6-45. ENDAT<m>_CLK Stop High on Last RX Frame

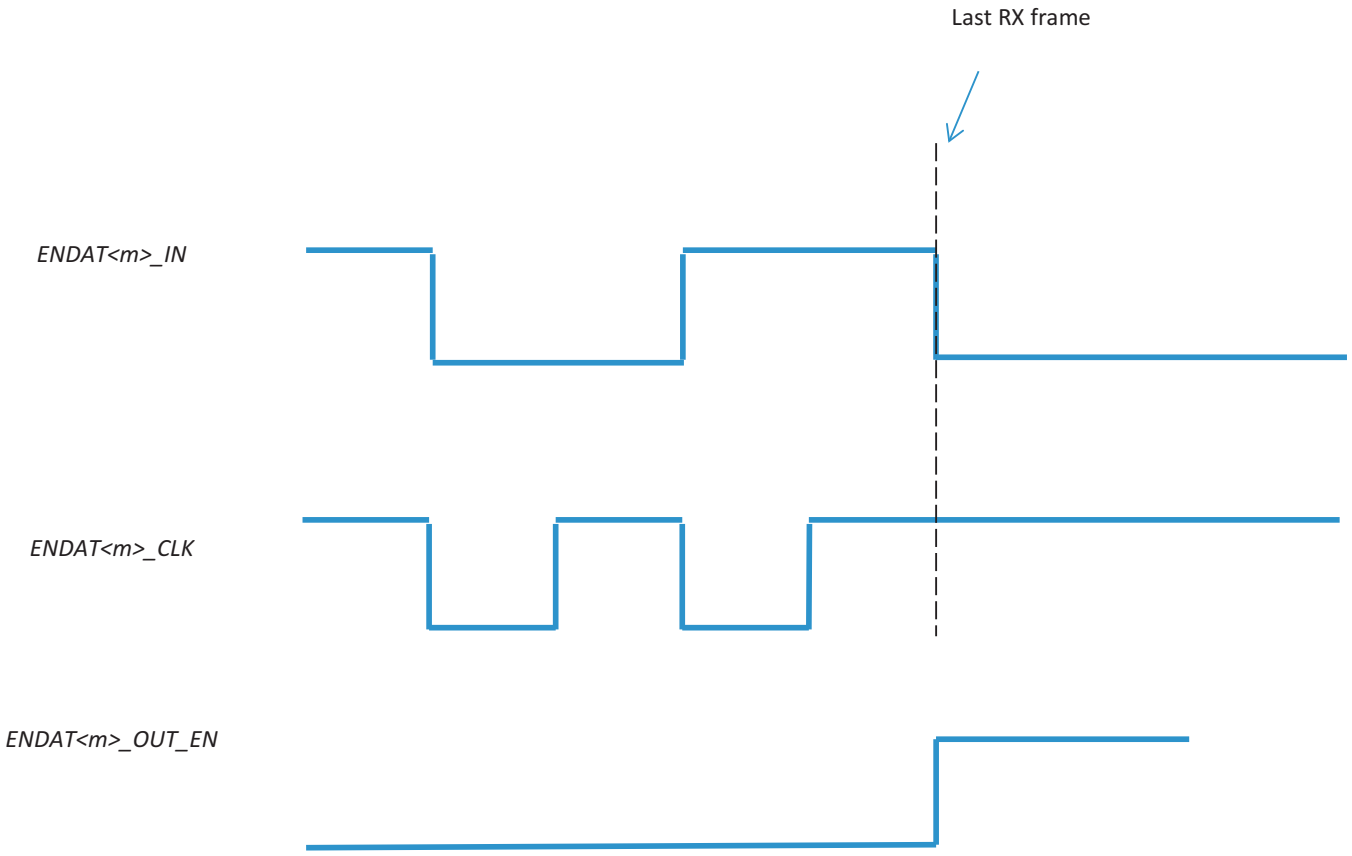


Figure 6-46. ENDAT<m>_CLK Stop Low on Last RX Frame

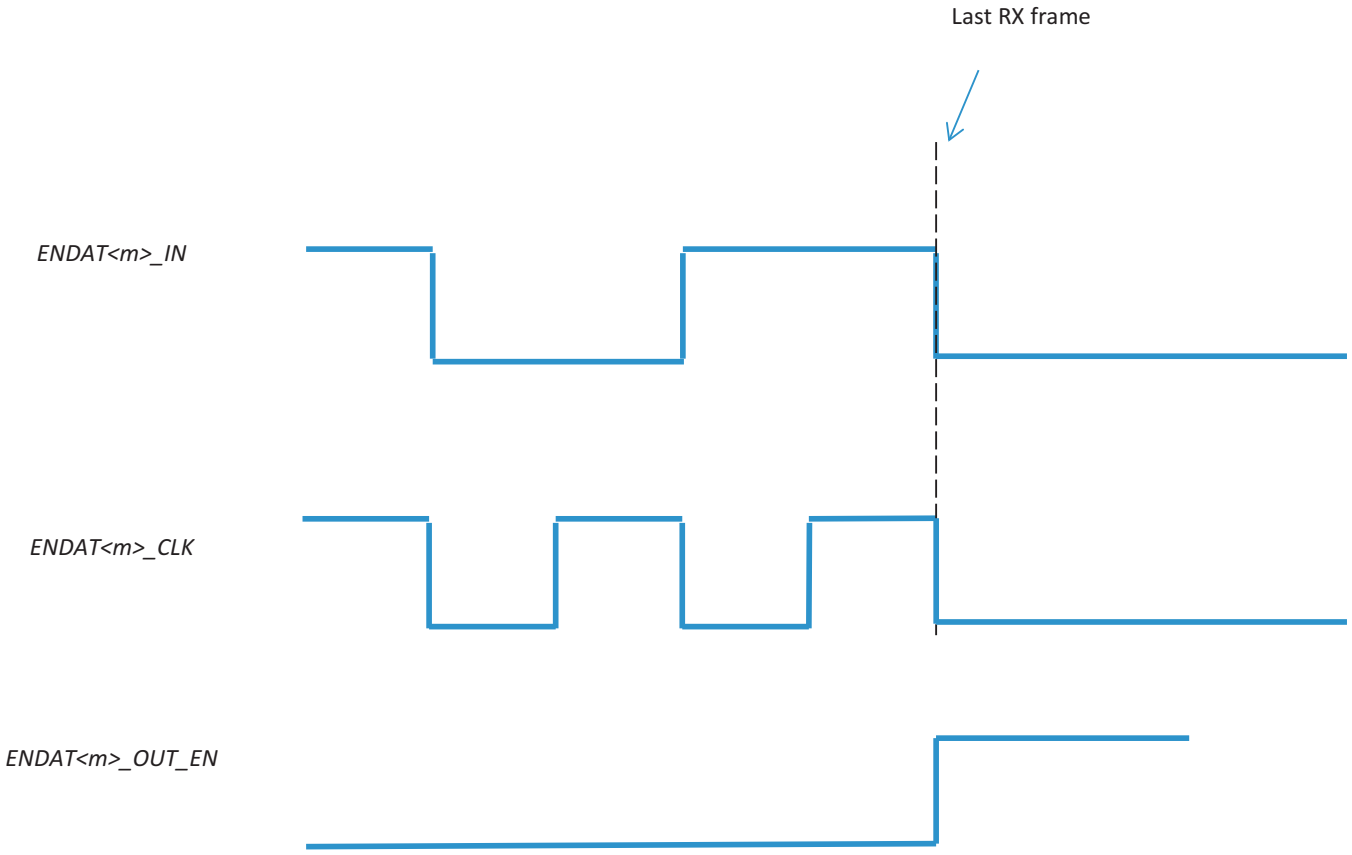


Figure 6-47. ENDAT<m>_CLK Run Continuously

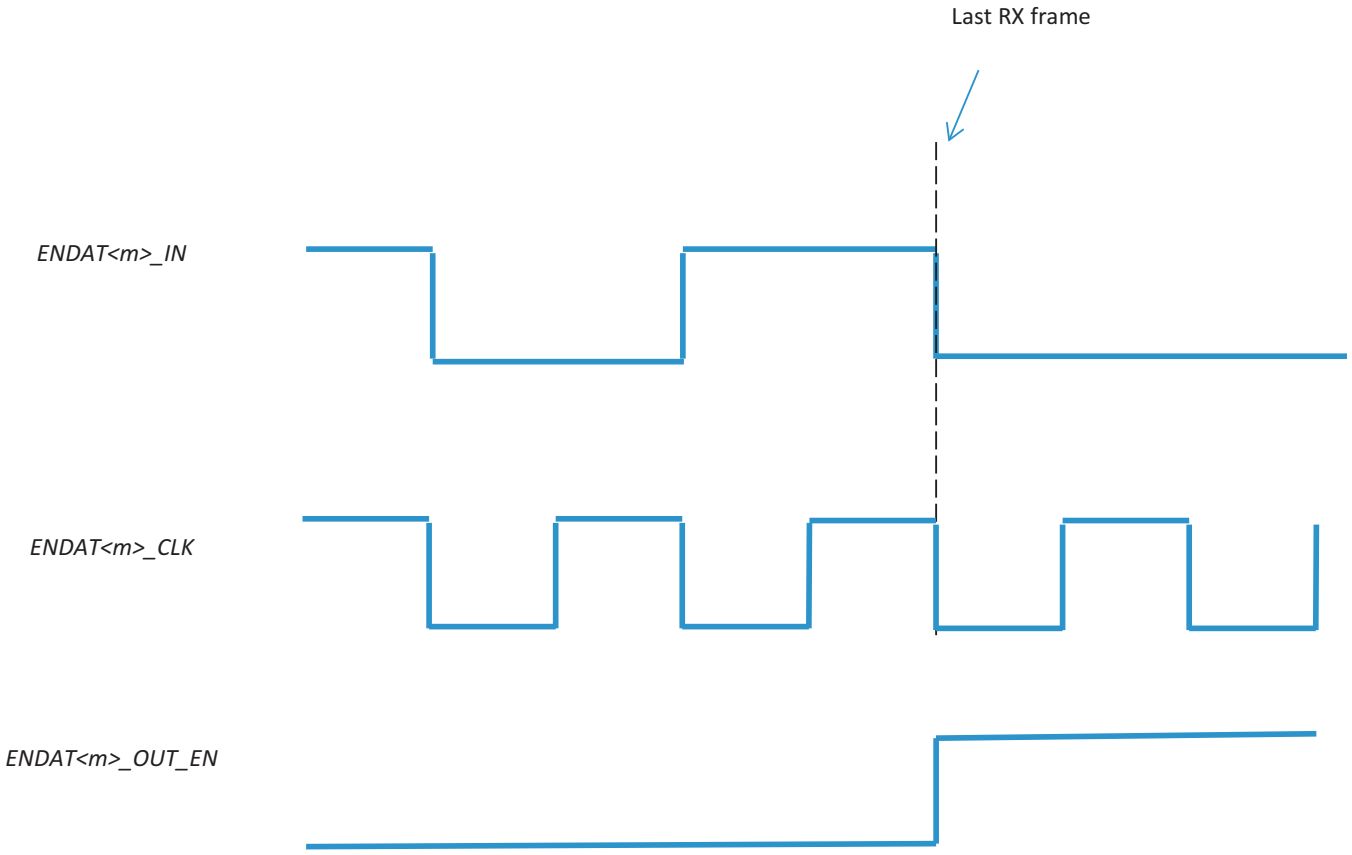
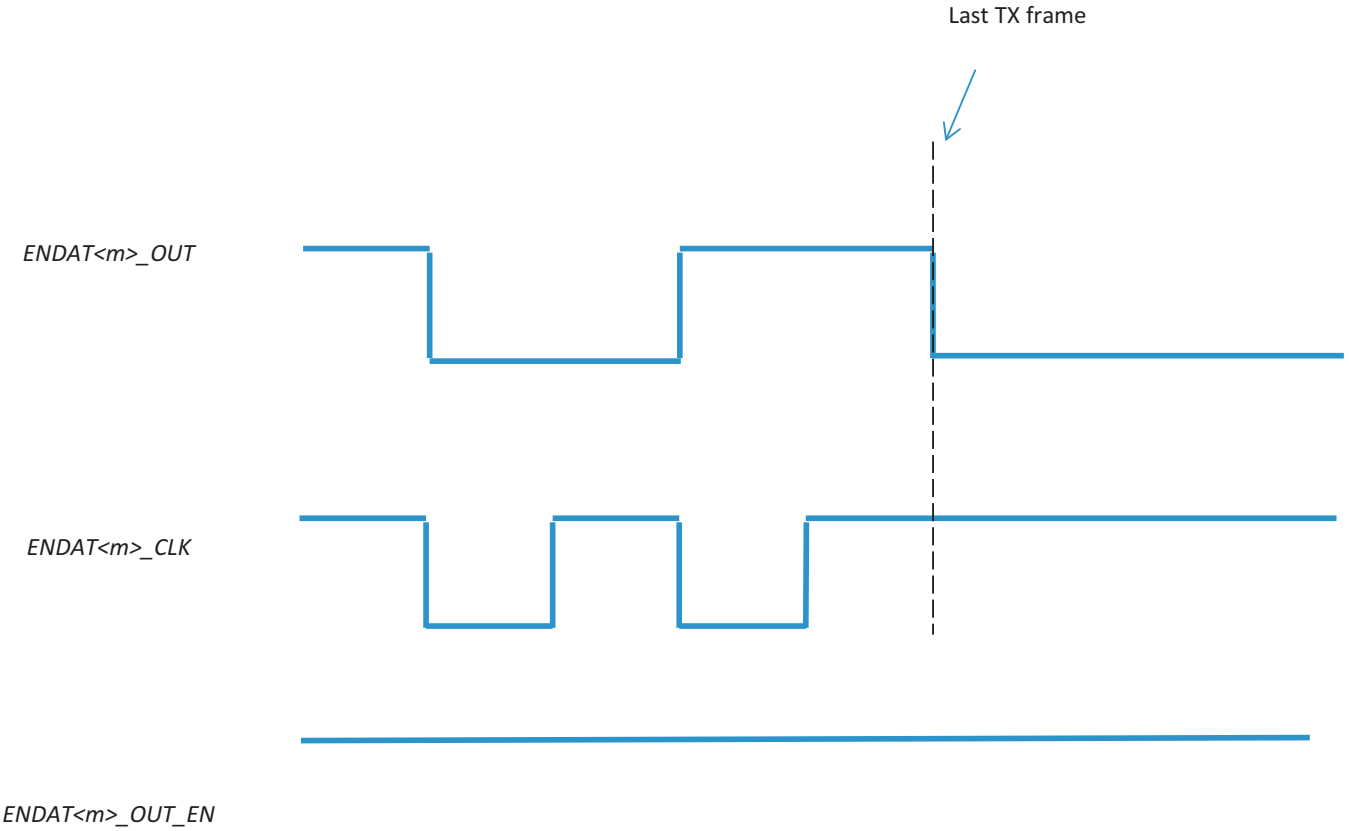


Figure 6-48. ENDAT<m>_CLK Stop High on Last TX Bit



6.4.5.3.4 Basic Programming Model

The following programming models assume that the PRU is configured for 3 Peripheral Mode (PRUSS_GPCFG0/1[29-26] PR1_PRUn_GP_MUX_SEL = 1h).

6.4.5.3.4.1 Clock Generation

Follow these steps to configure Peripheral I/F clocks using the HW control of the clock:

1. Select TX and RX clock sources:
   a. PRUSS_ED_PRUn_TX_CFG_REGISTER[4] PRUn_ED_TX_CLK_SEL for the TX clock source
   b. PRUSS_ED_PRUn_RX_CFG_REGISTER[4] PRUn_ED_RX_CLK_SEL for the RX clock source
2. Configure the 1x (TX) clock frequency:
   a. Write Division Factor to PRUSS_ED_PRUn_TX_CFG_REGISTER[31-16] PRUn_ED_TX_DIV_FACTOR
   b. Write Fraction division factor to PRUSS_ED_PRUn_TX_CFG_REGISTER[15] PRUn_ED_TX_DIV_FACTOR_FRAC
3. Configure the oversampling (RX) frequency and oversample size:
   a. Write Division Factor to PRUSS_ED_PRUn_RX_CFG_REGISTER[31-16] PRUn_ED_RX_DIV_FACTOR
   b. Write Fraction division factor to PRUSS_ED_PRUn_RX_CFG_REGISTER[15] PRUn_ED_RX_DIV_FACTOR_FRAC
   c. Write RX oversample size to PRUSS_ED_PRUn_RX_CFG_REGISTER[2-0] PRUn_ED_RX_SAMPLE_SIZE
4. Select the clk_mode to configure how the ENDAT<m>_CLK signal ends after TX/RX:
   a. Write to r30[20:19] (clk_mode). Note the clk_mode setting can also be changed per transaction.

5. Configure the wire, tst, and rx_en_counter delay values:
   a. PRUSS_ED_PRUn_CHj_CFG0_REGISTER[10-0] PRUn_ED_TX_WDLY for wire delay
   b. PRUSS_ED_PRUn_CHj_CFG1_REGISTER[15-0] PRUn_ED_TST_DELAY_COUNTER for tst delay
   c. PRUSS_ED_PRUn_CHj_CFG1_REGISTER[31-16] PRUn_ED_RX_EN_COUNTER for auto-delay between TX and RX

6.4.5.3.4.2 TX - Single Shot

Follow these steps to configure the Peripheral I/F channel(s) for a single shot transmission:

1. (Optional) Configure TX FIFO for MSB (default) or LSB:
   1. PRUSS_ED_PRUn_CHj_CFG0_REGISTER[31] PRUn_ED_TX_FIFO_SWAP_BITS
2. Pre-load TX FIFO:
   1. Select TX channel by writing the desired channel number to R30[17:16] (tx_ch_sel)
   2. Write 1-4 bytes of data to r30[7:0] (tx_data). At each r30[7:0] write, data will be pushed into the FIFO.
   3. Repeat Steps 2a and 2b for all desired channels.
3. Configure TX frame size if less than 4 full bytes loaded into FIFO:
   1. PRUSS_ED_PRUn_CHj_CFG0_REGISTER[15-11] PRUn_ED_TX_FRAME_SIZE
4. Push TX FIFO data to ENDAT<m>_OUT (see Section 6.4.5.3.3.2 for the ENDAT<m>_CLK and ENDAT<m>_OUT start time relationship);
   1. To start TX on all channels, set r31[20] = 1 (tx_global_go).
   2. To start TX on individual channel:
      i. Select TX channel by writing the desired channel number to R30[17:16] (tx_ch_sel)
      ii. Set R31[18] = 1 (tx_channel_go)
5. If PRUSS_ED_PRUn_CHj_CFG0_REGISTER[31-16] PRUn_ED_RX_EN_COUNTER > 0, then the channel will automatically switch into RX mode. See Section 6.4.5.3.4.4 for an example of how to program and configure RX content.
6. If PRUSS_ED_PRUn_CHj_CFG0_REGISTER[31-16] PRUn_ED_RX_EN_COUNTER = 0, poll either r31[21, 13, or 5] (tx_global_reinit_active/busy[2,1,0]) or PRUSS_ED_PRUn_TX_CFG_REGISTER[7, 6, or 5] PRUn_ED_BUSY_i (where i = 0 to 2, indicates channel number) for when TX is complete

NOTE: The ENDAT<m>_CLK Peripheral I/F requires that ENDAT<m>_CLK be in a high state at the beginning of a new transaction. If the clock ended the single shot transmission in low state, then the clock needs to be reset before sending more data. The steps to reset ENDAT<m>_CLK are:

1. Set R31[19] = 1 (tx_global_reinit) to reset clock high
2. Wait until tx_busy<m> is cleared
3. Re-configure R30[20:19] (clk_mode), since reinit will reset the clk_mode to "Free-running/stop-high" mode

6.4.5.3.4.3 TX - Continuous FIFO Loading

Follow these steps to configure the Peripheral I/F channel(s) for a continuous loading transmission:

1. (Optional) Configure TX FIFO for MSB (default) or LSB:
   a. PRUSS_ED_PRUn_CHj_CFG0_REGISTER[31] PRUn_ED_TX_FIFO_SWAP_BITS
2. Pre-load TX FIFO:
   a. Select TX channel by writing the desired channel number to r30[17:16] (tx_ch_sel)
   b. Write 1-4 bytes of data to r30[7:0] (tx_data). At each r30[7:0] write, data will be pushed into the FIFO.

c. Repeat Steps 2a and 2b for all desired channels.

3. Configure TX frame size to continuously transmit the TX FIFO until empty:
   a. Set PRUSS_ED_PRUn_CHj_CFG0_REGISTER[15-11] PRUn_ED_TX_FRAME_SIZE = 0h
4. Push TX FIFO data to ENDAT<m>_OUT (see Section 6.4.5.3.3.2 for the ENDAT<m>_CLK and ENDAT<m>_OUT start time relationship):
   a. To start TX on all channels, set r31[20] = 1 (tx_global_go).
   b. To start TX on individual channel:
      i. Select TX channel by writing the desired channel number to r30[17:16] (tx_ch_sel)
      ii. Set r31[18] = 1 (tx_channel_go)
5. Monitor line rate and reload FIFO:
   a. Polling r31[xx, 12:10, 4:2] (tx_fifo_sts<m>)
   b. When FIFO level is at 2 bytes, load next 2 bytes of data (see Step 2). Do not let the FIFO get close to 0. Once the FIFO runs empty, the hardware will assume the PRU has reached end of the last transmit. Any new writes to the FIFO will NOT be sent until the software sends another tx_channel_go bit. Note there are also underrun and overrun error flags that can be monitored.
6. To end TX operation, do not send any new data to FIFO.
   a. If PRUSS_ED_PRUn_CHj_CFG1_REGISTER[31-16] PRUn_ED_RX_EN_COUNTER > 0, then the channel will automatically switch into RX mode. See Section 6.4.5.3.4.4 for an example of how to program and configure RX content.
   b. If PRUSS_ED_PRUn_CHj_CFG1_REGISTER[31-16] PRUn_ED_RX_EN_COUNTER = 0, poll either r31[21, 13, or 5] (tx_global_reinit_active/busy[2,1,0]) or PRUSS_ED_PRUn_TX_CFG_REGISTER[7, 6, or 5] PRUn_ED_BUSY_i (where i = 0 to 2, indicates channel number) for when TX is complete

NOTE: The ENDAT<m>_CLK Peripheral I/F requires that ENDAT<m>_CLK be in a high state at the beginning of a new transaction. If the clock ended the continuous loading transmission in low state, then the clock needs to be reset before sending more data. The steps to reset ENDAT<m>_CLK are:

1. Set R31[19] = 1 (tx_global_reinit) to reset clock high
2. Wait until tx_busy<m> is cleared
3. Re-configure R30[20:19] (clk_mode), since reinit will reset the clk_mode to "Free-running/stop-high" mode

6.4.5.3.4.4 RX - Auto Arm or Non-Auto Arm

Follow these steps to configure the Peripheral I/F channel(s) to receive data:

1. Configure RX and frame size:
   a. PRUSS_ED_PRUn_CHj_CFG0_REGISTER[27-16] PRUn_ED_RX_FRAME_SIZE
2. To start ENDAT<m>_CLK:
   a. For the non-auto arm use case, set r30[26, 25, 24] = 1 (rx_en<m>)
   b. For the auto arm use case, rx_en<m> will be automatically enabled at the end of a TX operation when PRUSS_ED_PRUn_CHj_CFG1_REGISTER[31-16] PRUn_ED_RX_EN_COUNTER > 0
3. RX FIFO will start filling on the first start bit (ENDAT<m>_IN = 1). The data will be captured on the positive edge of the ENDAT<m>_CLK and shifted into the LSB position of the 8-bit shadow register.
4. Poll for r31[26, 25, 24] (val<m>) assertion. The valid flag will be asserted when n bits of data (determined by PRUSS_ED_PRUn_RX_CFG_REGISTER[2-0] PRUn_ED_RX_SAMPLE_SIZE) have been collected.
5. Fetch data by reading r31[23-16, 15-8, 7-0] (rx_data_out<m>). The data will remain constant for one data frame, and PRU must read data and clear valid flag within this time. Otherwise, an overflow will occur – r31[29, 28, 27] (ovf<m>) = 1 - indicating that val<m> has been continuously asserted for longer than one data frame.

6. The clock will be stopped based on the r30[20:19] (clk_mode) configured before the start of the RX operation.
7. Clear r30[26, 25, 24] (rx_en<m>) to disable RX mode. All counters and flags will be reset.

6.4.5.4 PRU Multiplier with Accumulation (MPY/MAC)

This section describes the MAC (multiplier with accumulation) module integrated to PRU0 and PRU1 cores of PRU-ICSS_0/PRU-ICSS_1.

6.4.5.4.1 PRU MACs Overview

Each of the two PRU cores (PRU0 and PRU1) has a designated unsigned multiplier with accumulation (MPY/MAC). The MAC supports two modes of operation: Multiply Only and Multiply and Accumulate.

The MAC is directly connected with the PRU internal registers R25-R29 and uses the broadside load/store PRU interface and XFR instructions to both control and mode of the MAC and import the multiplication results into the PRU.

6.4.5.4.1.1 PRU MAC Key Features

The MPY/MAC features are:

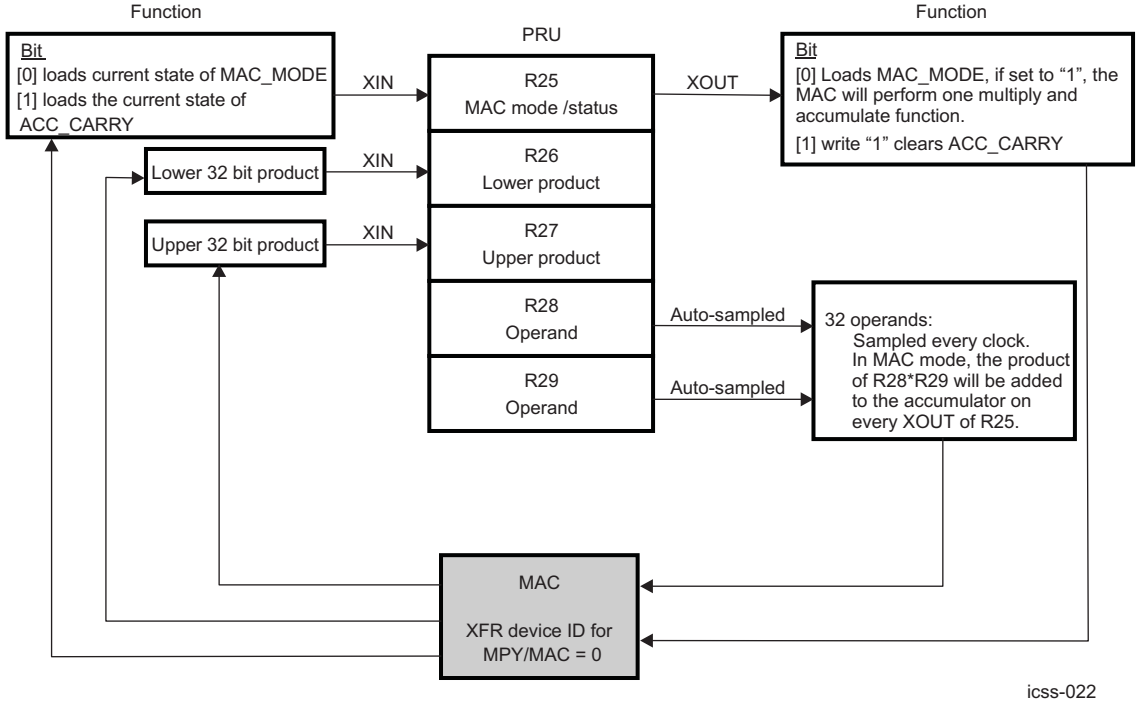
- Configurable Multiply Only and Multiply and Accumulate functionality via PRU register R25
- 32-bit operands with direct connection to PRU registers R28 and R29
- 64-bit result (with carry flag) with direct connection to PRU registers R26 and R27
- PRU broadside interface and XFR instructions (XIN, XOUT) allow for importing multiplication results and initiating accumulate function

6.4.5.4.1.2 PRU MAC Operations

6.4.5.4.1.2.1 PRU versus MAC Interface

The MAC directly connects with the PRU internal registers R25-R29 through use of the PRU broadside interface and XFR instructions. Figure 6-49 shows the functionality of each register.

Figure 6-49. Integration of the PRU and MPY/MAC



The XFR instructions (XIN and XOUT) are used to load/store register contents between the PRU core and the MAC. These instructions define the start, size, direction of the operation, and device ID. The device ID number corresponding to the MPY/MAC is shown in Table 6-96.

Table 6-96. MPY/MAC XFR ID

Device ID	Function
0	Selects MPY/MAC

The PRU register R25 is mapped to the MAC_CTRL_STATUS register (Table 6-97). The MAC's current status (MAC_MODE and ACC_CARRY states) is loaded into R25 using the XIN command on R25. The PRU sets the MAC's mode and clears the ACC_CARRY using the XOUT command on R25.

Table 6-97. MAC_CTRL_STATUS Register (R25) Field Descriptions

Bit	Field	Description
7-2	RESERVED	Reserved
1	ACC_CARRY	Write 1 to clear. 0h: 64-bit accumulator carry has not occurred 1h: 64-bit accumulator carry occurred
0	MAC_MODE	0h: Accumulation mode disabled and accumulator is cleared 1h: Accumulation mode enabled

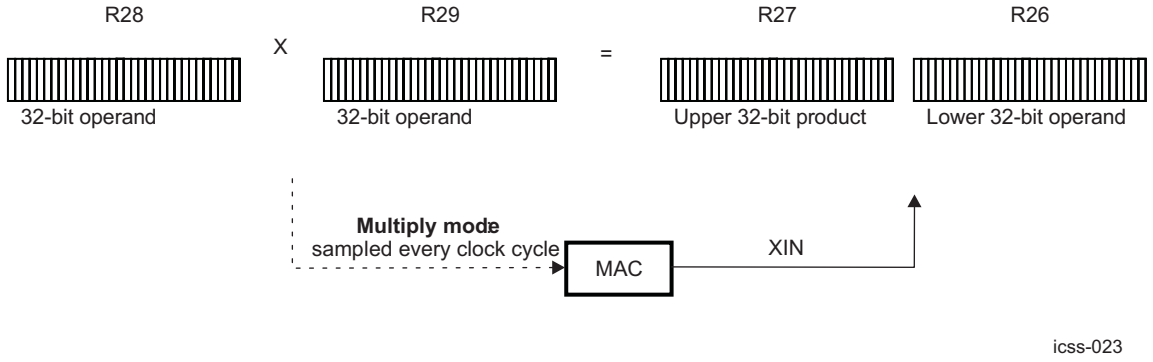
The two 32-bit operands for the multiplication are loaded into R28 and R29. These registers have a direction connection with the MAC. Therefore, XOUT is not required to load the MAC. In multiply mode, the MAC samples these registers every clock cycle. In multiply and accumulate mode, the MAC samples these registers every XOUT R25[7:0] transaction when MAC_MODE = 1.

The product from the MAC is linked to R26 (lower 32 bits) and R27 (upper 32 bits). The product is loaded into register R26 and R27 using XIN.

6.4.5.4.1.2.2 Multiply only mode(default state), MAC_MODE = 0

The Figure 6-50 summarizes the MAC operation in "Multiply-only"mode, in which the MAC multiplies the contents of R28 and R29 on every clock cycle.

Figure 6-50. MAC Multiply-only Mode- Functional Diagram



6.4.5.4.1.2.2.1 Programming PRU MAC in "Multiply-ONLY" mode

The following steps are performed by the PRU firmware for multiply-only mode:

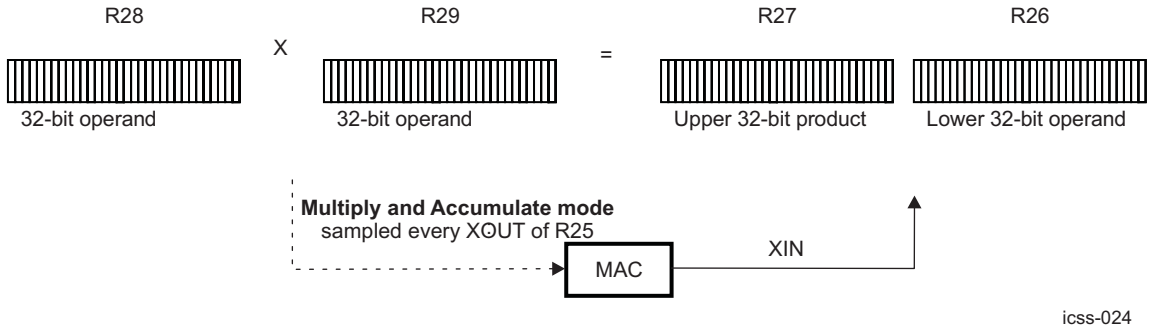
1. Enable multiply only MAC_MODE.
   (a) Clear R25[0] for multiply only mode.
   (b) Store MAC_MODE to MAC using XOUT instruction with the following parameters:
   - Device ID = 0
   - Base register = R25
   - Size = 1
2. Load operands into R28 and R29.
3. Delay at least 1 PRU cycle before executing XIN in step 4.
4. Load product into PRU using XIN instruction on R26, R27.

Repeat steps 2 and 4 for each new operand.

6.4.5.4.1.2.3 Multiply and Accumulate Mode, MAC_MODE = 1

The Figure 6-51 summarizes the MAC operation in "Multiply and Accumulate" mode. On every XOUT R25_REG[7:0] transaction, the MAC multiplies the contents of R28 and R29, adds the product to its accumulated result, and sets ACC_CARRY if an accumulation overflow occurs.

Figure 6-51. MAC Multiply and Accumulate Mode Functional Diagram



6.4.5.4.1.2.3.1 Programming PRU MAC in Multiply and Accumulate Mode

The following steps are performed by the PRU firmware for multiply and accumulate mode:

1. Enable multiply and accumulate MAC_MODE.

(a) Set R25[1:0] = 1 for accumulate mode.
(b) Store MAC_MODE to MAC using XOUT instruction with the following parameters:
- Device ID = 0
- Base register = R25
- Size = 1

2. Clear accumulator and carry flag.
(a) Set R25[1:0] = 3 to clear accumulator (R25[1]=1) and preserve accumulate mode (R25[0]=1).
(b) Store accumulator to MAC using XOUT instruction on R25.

3. Load operands into R28 and R29.

4. Multiply and accumulate, XOUT R25[1:0] = 1

Repeat step 4 for each multiply and accumulate using same operands.

Repeat step 3 and 4 for each multiply and accumulate for new operands.

5. Load the accumulated product into R26, R27, and the ACC_CARRY status into R25 using the XIN instruction.

NOTE: Steps one and two are required to set the accumulator mode and clear the accumulator and carry flag.

6.4.5.5 CRC16/32

The PRU0 and PRU1 cores of PRU-ICSS_0/PRU-ICSS_1 each have a designated CRC16/32 module.

In general, CRC adds error detection capability to communication systems. The CRC encoder appends redundant bits (or CRC bits) to the systematic data message. During reception of the data message, the received data is also encoded with the same CRC encoder. The 2 sets of CRC bits are compared together. If they match, there were no transmission errors; and if they don't match, a transmission error has been detected.

6.4.5.5.1 Features

CRC16/32 supports the following features:

- Supports CRC32:
  - $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x+1$
- Supports CRC16:
  - $x^{16}+x^{15}+x^{2}+1$
- PRU broadside interface and XFR instructions (XIN, XOUT) allow for importing CRC results and executing accumulate function

6.4.5.5.2 PRU and CRC16/32 Interface

The CRC16/32 module directly connects with the PRU internal registers R25-R29 through use of the PRU broadside interface and XFR instructions. shows the functionality of each register.

The XFR instructions (XIN and XOUT) are used to load/store register contents between the PRU core and the CRC16/32 module. These instructions define the start, size, direction of the operation, and device ID. The XFR device ID number corresponding to the CRC16/32 module is 1.

Table 6-98. CRC Register to PRU Port Mapping

CRC Register	R/W	Description	PRU Mapping
CRC_CFG	W	bit [0] CRC32_ENABLE: 0: CRC16 mode is selected. Hardware will auto-set init state of CRC_SEED to 0000_0000h. Note CRC16 result value is only 16-bits 1: CRC32 mode is selected. Hardware will auto-set init state of CRC_SEED will be FFFF_FFFFh. Always write all 4 bytes.	R25_reg
CRC_DATA_8_BFLIP	R	8-bit flip of CRC_DATA. CRC_DATA_8_BFLIP has the same byte order as CRC_DATA[31:0], but each byte has all bits flipped. CRC_DATA_32_FLIP[7:0] = CRC_DATA[0:7] CRC_DATA_32_FLIP[15:8] = CRC_DATA[8:15] CRC_DATA_32_FLIP[23:16] = CRC_DATA[16:23] CRC_DATA_32_FLIP[31:24] = CRC_DATA[24:31] For CRC16, only CRC_DATA_8_BFLIP[15:0] are valid. No auto reset on CRC_DATA_8_BFLIP read.	R27_reg
CRC_SEED	W	CRC SEED value. Hardware will auto-initialize the CRC_SEED value to 0000_0000h for CRC16 and FFFF_FFFFh for CRC32. Software only needs to initialize CRC_SEED if a different default value is required. Always write 4 bytes. Note when CRC_CFG[CRC32_ENABLE] is enabled, the hardware will switch the CRC_SEED value to FFFF_FFFFh. Reading the CRC_DATA register will reset the CRC value to the CRC_SEED state.	R28_reg
CRC_DATA_32_BFLIP	R	Full 32-bit flip of CRC_DATA CRC_DATA_32_BFLIP[0] = CRC_DATA[31] … CRC_DATA_32_BFLIP[31] = CRC_DATA[0] For CRC16, only CRC_DATA_32_BFLIP[31:16] are valid. No auto reset on CRC_DATA_32_BFLIP read.	R28_reg
CRC_DATA	RW	For Write, must use a fixed width throughout the session. The CRC module supports lower 8-bit, or lower 16-bit, or full 32-bit data widths. For Read, LSB or CRC_DATA[0] is first bit on the wire. Note for CRC16, only CRC_DATA[15:0] is valid. Hardware will delay CRC_DATA read operation up to 1 clock if it occurs back to back with a CRC_DATA write.	R29_reg

6.4.5.5.3 Programming Model

The following steps are performed by the PRU firmware to use the CRC module:

Step1: Configuration (optional)

1. Configure CRC type:
   For CRC32 operation, set CRC32_ENABLE using XOUT instruction with the following parameters:
   - Device ID = 1
   - Base register = R25
   - Size = 1
2. Update CRC_SEED, if required using XOUT with the following parameters:
   - Device ID = 1
   - Base register = R28
   - Size = 1 to 4

Step 2:

1. Load new CRC data into R29
2. Push CRC data to the CRC16/32 module using XOUT with the following parameters:
   - Device ID = 1
   - Base register = R29
   - Size = 1 to 4

3. Load the accumulated CRC result into the PRU using the XIN instruction with the following parameters:
   - Device ID = 1
   - Base register = R29
   - Size = 4

Repeat Step 2, numbers 1 and 2 for each new CRC data.

NOTE: When a session starts, the PRU firmware must use the same write data width throughout the session.

6.4.5.6 PRU0 and PRU1 Scratch Pad Memory

The PRU-ICSS supports a scratch pad with three independent banks accessible by the PRU cores. The PRU cores interact with the scratch pad through broadside load/store PRU interface and XFR instructions. The scratch pad can be used as a temporary place holder for the register contents of the PRU cores. Direct connection between the PRU cores is also supported for transferring register contents directly between the cores.

This section describes the Scratch Pad Memory shared between and directly accessible by the PRU0 and PRU1 cores, as well as the XFR direct method used by the PRU cores of the PRU-ICSS_0 /PRU-ICSS_1.

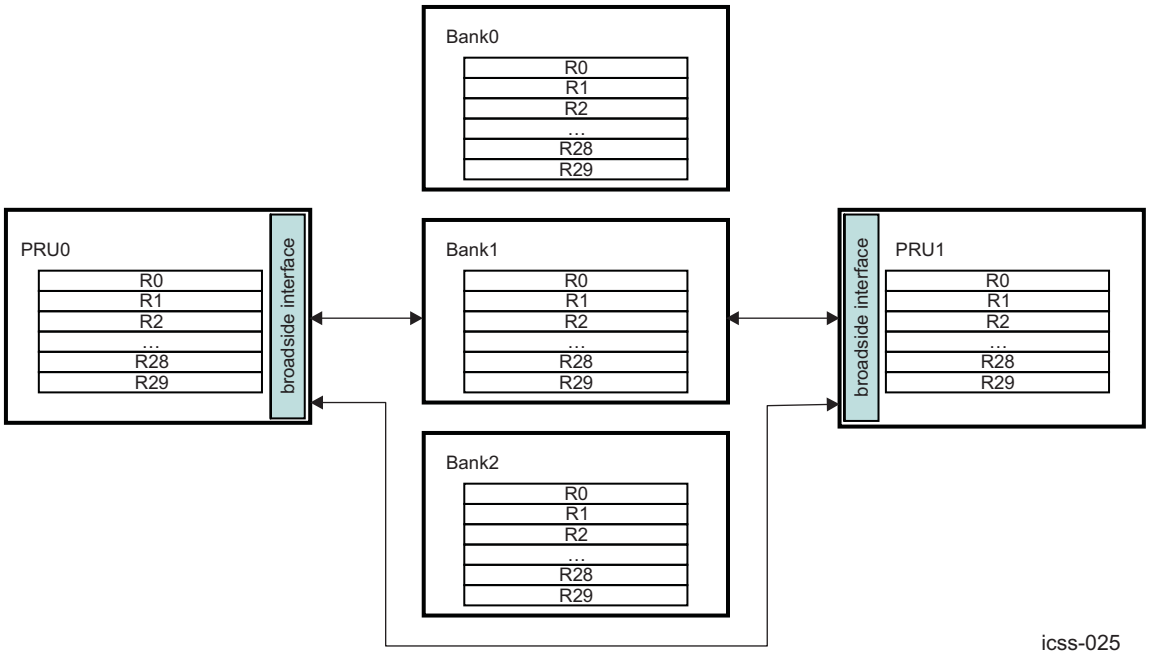
6.4.5.6.1 PRU0/1 Scratch Pad Overview

The PRU-ICSS scratch pad supports the following features:

- Three scratch pad banks of 30, 32-bit registers (R29:0)
- Flexible load/store options
  - User-defined start byte and length of the transfer
  - Length of transfer ranges from one byte of a register to the entire register content (R29 to R0)
  - Simultaneous transaction supported between PRU0 <-> Bank <n> and PRU1 <-> Bank <m>
  - Direct connection of PRU0->PRU1 or PRU1->PRU0 for all registers R29-R0
- XFR instructions operate in one clock cycle
- Optional XIN/XOUT shift functionality allows remapping of registers (R<n> -> R<m>) during load store operation

Figure 6-52 shows a simplified model of the ScratchPad integration.

Figure 6-52. ScratchPad and PRU Integration



6.4.5.6.2 PRU0 /1 Scratch Pad Operations

XFR instructions are used to load/store register contents between the PRU cores and the scratch pad banks. These instructions define the start, size, direction of the operation, and device ID. The device ID corresponds to the external source or destination (either a scratch pad bank or the other PRU core). The device ID numbers are shown in Table 6-99. Note the direct connect mode (device ID 14) can be used to synchronize the PRU cores. This mode requires the transmitting PRU core to execute XOUT and the receiving PRU core to execute XIN.

Table 6-99. Scratch Pad XFR ID

Device ID	Function/Operation
10	Selects Bank0
11	Selects Bank1
12	Selects Bank2
13	Reserved
14	Selects other PRU core (Direct connect mode)

A collision occurs when two XOUT commands simultaneously access the same asset or device ID. Table 6-100 shows the priority assigned to each operation when a collision occurs. In direct connect mode (device ID 14), any PRU transaction will be terminated if the stall is greater than 1024 cycles. This will generate the event pr<k>_xfr_timeout that is connected to INTC.

Table 6-100. Scratch Pad XFR Collision and Stall Conditions

Operation	Collision and Stall Handling
PRU<n> XOUT (->) bank[j]	If both PRU cores access the same bank simultaneously, PRU0 is given priority. PRU1 will temporarily stall until the PRU0 operation completes.
PRU<n> XOUT (->) PRU<m>	Direct connect mode requires the transmitting core (PRU<n>) to execute XOUT and the receiving core (PRU<m>) to execute XIN. If PRU<n> executes XOUT before PRU<m> executes XIN, then PRU<n> will stall until either PRU<m> executes XIN or the stall is greater than 1024 cycles.

Table 6-100. Scratch Pad XFR Collision and Stall Conditions (continued)

Operation	Collision and Stall Handling
PRU<m> XIN (<-) PRU<n>	Direct connect mode requires the transmitting core (PRU<n>) to execute XOUT and the receiving core (PRU<m>) to execute XIN. If PRU<m> executes XIN before PRU<n> executes XOUT, then PRU<m> will stall until either PRU<n> executes XOUT or the stall is greater than 1024 cycles.

6.4.5.6.2.1 Optional XIN/XOUT Shift

The optional XIN/XOUT shift functionality allows register contents to be remapped or shifted within the destination's register space. For example, the contents of PRU0 R6-R8 could be remapped to Bank1 R10-12. The XIN/XOUT shift feature is not supported for direct connect mode, only for transfers between a PRU core and scratch pad bank.

The shift feature is enabled or disabled through the PRU subsystem level register PRUSS_SPP[1] XFR_SHIFT_EN bit. When enabled, R0[4-0] (internal to the PRU) defines the number of 32-bit registers in which content is shifted in the scratch pad bank. Note that scratch pad banks do not have registers R30 or R31.

6.4.5.6.2.2 Example Scratch Pad Operations

The following PRU firmware examples demonstrate the shift functionality. Note these assume the XFR_SHIFT_EN bit of the PRUSS_SPP register of the PRU-ICSS CFG register space has been set.

XOUT Shift By 4 Registers

Store R4:R7 to R8:R11 in bank0:

- Load 4 into R0.b0
- XOUT using the following parameters:
  - Device ID = 10
  - Base register = R4
  - Size = 16

XOUT Shift By 9 Registers, With Wrap Around

Store R25:R29 to R4:R8 in bank1:

- Load 9 into R0.b0
- XOUT using the following parameters:
  - Device ID = 11
  - Base register = R25
  - Size = 20

XIN Shift By 10 Registers

Load R14:R16 from bank2 to R4:R6:

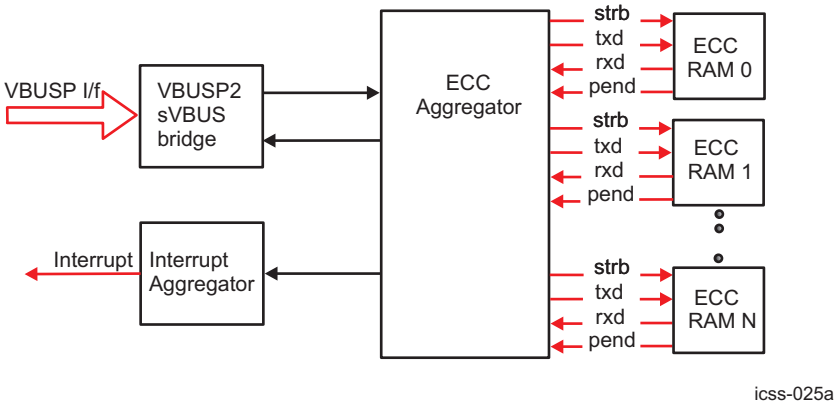
- Load 10 into R0.b0
- XIN using the following parameters:
  - Device ID = 12
  - Base register = R4
  - Size = 12

6.4.5.7 ECC Aggregator

This section describes the architecture and functional details of the ECC Aggregator module.

Figure 6-53 shows a Block Diagram of the ECC Aggregator module.

Figure 6-53. ECC Aggregator Block Diagram



6.4.5.7.1 ECC Aggregator module supported features

The ECC Aggregator module provides the following features:

- Aggregates level pending status from the ECC RAMs into a single interrupt to the host.
- Provides a mechanism to control and monitor the ECC RAMs in the ICSS subsystem via a standard 32-bit VBUSP interface.
- Supports software readable status of ECC single/double-bit errors and associated info such as RAM address and data bit(s) that are in error.
- Uses a low pin-count serial VBUSP protocol to communicate to all the ECC RAMs while bridging the standard VBUSP protocol to the serial protocol on the host interface.

6.4.5.7.2 ECC Aggregator module not supported features

The ECC Aggregator module does not support the following feature:

- Statistics such as tracking the number of single and double-bit errors. If needed these operations can be handled by software.

6.4.5.7.3 Functional operation of the ECC Aggregator module

The ECC Aggregator module is responsible for the following functions:

- Providing software access to all the ECC related registers via a standard 32-bit VBUSP interface.
- Aggregating the level pending interrupts from the ECC RAMs to the standard EOI-handshake based interrupt to the host.

6.4.5.7.3.1 ECC Aggregator Registers

The ECC Aggregator module supports all the ECC control, status and interrupt registers for all the ECC RAMs in the PRU-ICSS_0/1 subsystem. The ECC Aggregator registers are memory mapped to a 1 KB address space in the PRU-ICSS_0/1 module. There are 3 types of register accesses in the ECC Aggregator module:

- Global registers that are common to all ECC RAMs. This includes the ECC_REVISION and ECC_VECTOR registers. Every ECC RAM serviced by the Aggregator module is assigned a unique ID by the module. Writing this ID to the ECC_VECTOR register selects the ECC RAM to be controlled or read the ECC status from.
- ECC control and status registers. These are specific to each ECC RAM and selected by the bit field [10:0] RAM_ID written to the ECC_VECTOR register. The accesses to these registers are done via the serial VBUSP interface.
- Interrupt registers. These include the standard interrupt status, interrupt enable, clear and EOI registers that are part of a standard interrupt module.

6.4.5.7.3.2 Reads to ECC control and status registers

The reads to the ECC control and status registers for each ECC RAM are triggered by writing a 1h to the ECC_VECTOR[15] TRIGGER_READ register as described below:

1. Software writes the ECC ID (bits [10:0] RAM_ID), read trigger (bit [15] TRIGGER_READ), read address (bits [23:16] READ_ADDRESS) to the ECC_VECTOR register. The read address (bits [23:16] READ_ADDRESS) corresponds to any of the ECC control or status registers which require serial VBUS access to the ECC RAMs.
2. Software then polls the ECC_VECTOR[24] READ_DONE bit. If [24] READ_DONE bit is set to 1h, this indicates that the read operation has completed on the serial VBUS.
3. Software reads the data from the ECC control or status register. Read data is returned the following clock cycle.

6.4.5.7.3.3 ECC Aggregator Interrupts

The ECC Aggregator module aggregates all the level pending status from the ECC RAMs to a single EOI-handshake based interrupt to the host. Software is expected to follow the sequence described below:

1. Software enables the interrupts for the ECC RAMs by writing 1h to the [4-0]ENABLE bit in the ECC_INT_ENABLE_n register (where n = 0 to 15).
2. On receiving an interrupt, software checks the ECC_INT_STATUS_n register (where n = 0 to 15) to see which ECC RAMs caused the error. Note that this status read is not a serial VBUS operation.
3. Software writes the ECC_VECTOR[10:0] RAM_ID bit field along with the read message that includes setting bit ECC_VECTOR[15] TRIGGER_READ (to trigger the read), writing the ECC_ERROR_STATUS1 register address to the ECC_VECTOR[23:16] READ_ADDRESS bit field. Software will need to load the read message in the ECC_VECTOR register again if it needs to read the ECC_ERROR_STATUS2 register.
4. Software polls the [24] READ_DONE bit in the ECC_VECTOR register. When the READ_DONE bit is set to 1h, it does a read of the ECC_ERROR_STATUS1 and ECC_ERROR_STATUS2 registers.
5. After the interrupt has been serviced, software will clear the interrupt status by writing to bits [8] CLR_ECC_SEC, [9] CLR_ECC_DED or [10] CLR_ECC_OTHER depending on the type of ECC error in the ECC_ERROR_STATUS1 register.
6. Software has to poll the ECC_ERROR_STATUS1 register to guarantee that the status bit has been cleared. Otherwise there is no indication that the write has actually completed over the serial VBUS.
7. Software will write to the ECC_EOI register to clear the interrupt.

6.4.5.7.3.4 RAM Index Allocation

Table 6-101 shows the mapping of the RAM IDs to the ECC RAMs serviced by the aggregator.

Table 6-101. Mapping of the RAM IDs to the ECC RAMs

RAM Index	ECC RAM Prefix	Description
0	pr<k>_dram0_	Data RAM0 (8KB)
1	pr<k>_dram1_	Data RAM1 (8KB)
2	pr<k>_ram_	Shared Data RAM2 (64KB)
3	pr<k>_pru0_iram_	PRU0 Instruction Memory (16KB)
4	pr<k>_pru1_iram_	PRU1 Instruction Memory (16KB)

6.4.5.7.4 PRU-ICSS_ECC_CFG Registers

Table 6-103 lists the memory-mapped registers for the PRU-ICSS_ECC_CFG. All register offset addresses not listed in Table 6-103 should be considered as reserved locations and the register contents should not be modified.

Table 6-102. PRU-ICSS_ECC_CFG Instances

Module Name	Base Address
PRU-ICSS_0_ECC_CFG	20AA 7000h
PRU-ICSS_1_ECC_CFG	20AE 7000h

Table 6-103. PRU-ICSS_ECC_CFG Registers

Offset	Acronym	Register Name	PRU-ICSS_0_ ECC_CFG Physical Address	PRU-ICSS_1_ ECC_CFG Physical Address	Section
0h	ECC_REVISION	The Revision Register contains the ID and revision information for the ECC Aggregator module. It does not support byte access.	20AA 7000h	20AE 7000h	Section 11.13.5.1.6.1
8h	ECC_VECTOR	ECC RAM ID to select the ECC RAM to control or read status.	20AA 7008h	20AE 7008h	Section 11.13.5.1.6.2
Ch	ECC_MISC_STATUS	Miscellaneous status register.	20AA 700Ch	20AE 700Ch	Section 11.13.5.1.6.3
10h	ECC_WRAPPER_REVISION	The Revision Register contains the ID and revision information for the ECC wrapper.	20AA 7010h	20AE 7010h	Section 11.13.5.1.6.4
14h	ECC_CONTROL	The Control Register controls the ECC control bits for the selected ECC RAM.	20AA 7014h	20AE 7014h	Section 11.13.5.1.6.5
18h	ECC_ERROR_CONTROL1	This register contains ECC error control bits for the selected ECC RAM.	20AA 7018h	20AE 7018h	Section 11.13.5.1.6.6
1Ch	ECC_ERROR_CONTROL2	This register contains ECC error control bits for the selected ECC RAM.	20AA 701Ch	20AE 701Ch	Section 11.13.5.1.6.7
20h	ECC_ERROR_STATUS1	This register contains ECC status bits for the selected ECC RAM.	20AA 7020h	20AE 7020h	Section 11.13.5.1.6.8
24h	ECC_ERROR_STATUS2	This register contains ECC status bits for the selected ECC RAM.	20AA 7024h	20AE 7024h	Section 11.13.5.1.6.9
3Ch	ECC_EOI	This is the ECC_EOI register for the interrupt to the host.	20AA 703Ch	20AE 703Ch	Section 11.13.5.1.6.10
40h to 7Ch	ECC_INT_STATUS_0 to ECC_INT_STATUS_15	These are the raw level interrupt status bits where each bit corresponds to the pending status from an ECC RAM.	20AA 7040h to 20AA 707Ch	20AE 7040h to 20AE 707Ch	Section 11.13.5.1.6.11
80h to BCh	ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15	These are interrupt enables associated with the interrupt from each of the ECC RAMs.	20AA 7080h to 20AA 70BCh	20AE 7080h to 20AE 70BCh	Section 11.13.5.1.6.12
C0h to FCh	ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15	These are interrupt enable clear bits associated with the interrupt from each of the ECC RAMs.	20AA 70C0h to 20AA 70FCh	20AE 70C0h to 20AE 70FCh	Section 11.13.5.1.6.13

6.4.5.7.4.1 ECC_REVISION Register (Offset = 0h) [reset = 0h]

ECC_REVISION is shown in and described in Table 11-2375.

The Revision Register contains the ID and revision information for the module.

Table 6-104. ECC_REVISION Instances

Instance	Physical Address
PRU-ICSS_0_ECC_CFG	20AA 7000h
PRU-ICSS_1_ECC_CFG	20AE 7000h

Figure 6-54. ECC_REVISION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 4E8A0107h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-105. ECC_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R		TI internal data. Identifies revision of peripheral.

Table 6-106. Register Call Summary for ECC_REVISION

PRU-ICSS PRU Cores

- PRU-ICSS_ECC_CFG Registers: [0]
- ECC_REVISION Register (Offset = 0h) [reset = 0h]: [0]
- ECC Aggregator Registers: [0]

6.4.5.7.4.2 ECC_VECTOR Register (Offset = 8h) [reset = 0h]

ECC_VECTOR is shown in Figure 11-1034 and described in Table 11-2378.

ECC RAM ID to select the ECC RAM to control or read status.

Table 6-107. ECC_VECTOR Instances

Instance	Physical Address
PRU-ICSS_0_ECC_CFG	20AA 7008h
PRU-ICSS_1_ECC_CFG	20AE 7008h

Figure 6-55. ECC_VECTOR Register

31	30	29	28	27	26	25	24
RESERVED							READ_DONE
R-							R-
23	22	21	20	19	18	17	16
READ_ADDRESS							
R/W-							
15	14	13	12	11	10	9	8
TRIGGER_READ	RESERVED					RAM_ID	
R/W-	R-					R/W-	
7	6	5	4	3	2	1	0
RAM_ID							
R/W-							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-108. ECC_VECTOR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24	READ_DONE	R	0h	Status to indicate if read on the serial VBUS is complete.
23-16	READ_ADDRESS	R/W	0h	Read address. Can be any of the registers (10h – 24h).
15	TRIGGER_READ	R/W	0h	Write 1 to trigger a read on the serial VBUS.
14-11	RESERVED	R	0h	Reserved
10-0	RAM_ID	R/W	0h	Value written to select the corresponding ECC RAM for control or status. • 0h: 8KB Data RAM0 • 1h: 8KB Data RAM1 • 2h: 64KB RAM • 3h: 16KB IRAM0 • 4h: 16KB IRAM1 • 5h: Reserved

Table 6-109. Register Call Summary for ECC_VECTOR

PRU-ICSS PRU Cores
- PRU-ICSS_ECC_CFG Registers: [0]
- ECC_VECTOR Register (Offset = 8h) [reset = 0h]: [0]
- ECC Aggregator Registers: [0][1][2]
- Reads to ECC control and status registers: [0][1][2]
- ECC Aggregator Interrupts: [0][1][2][3][4]

6.4.5.7.4.3 ECC_MISC_STATUS Register (Offset = Ch) [reset = 0h]

ECC_MISC_STATUS is shown in Figure 11-1035 and described in Table 11-2381.

Miscellaneous status register.

Table 6-110. ECC_MISC_STATUS Instances

Instance	Physical Address
PRU-ICSS_0_ECC_CFG	20AA 700Ch
PRU-ICSS_1_ECC_CFG	20AE 700Ch

Figure 6-56. ECC_MISC_STATUS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					NUM_RAMS										
R-0h																					R-5h										

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-111. ECC_MISC_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Reserved
10-0	NUM_RAMS	R	5h	Indicates the number of RAMs serviced by the ECC aggregator.

Table 6-112. Register Call Summary for ECC_MISC_STATUS

PRU-ICSS PRU Cores

- PRU-ICSS_ECC_CFG Registers: [0]
- ECC_MISC_STATUS Register (Offset = Ch) [reset = 0h]: [0]

6.4.5.7.4.4 ECC_WRAPPER_REVISION Register (Offset = 10h) [reset = 0h]

ECC_WRAPPER_REVISION is shown in Figure 6-57 and described in Table 11-2384.

The Revision Register contains the ID and revision information for the ECC wrapper.

Table 6-113. ECC_WRAPPER_REVISION Instances

Instance	Physical Address
PRU-ICSS_0_ECC_CFG	20AA 7010h
PRU-ICSS_1_ECC_CFG	20AE 7010h

Figure 6-57. ECC_WRAPPER_REVISION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R -																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-114. ECC_WRAPPER_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R		TI internal data. Identifies revision of peripheral.

Table 6-115. Register Call Summary for ECC_WRAPPER_REVISION

PRU-ICSS PRU Cores

- PRU-ICSS_ECC_CFG Registers: [0]
- ECC_WRAPPER_REVISION Register (Offset = 10h) [reset = 0h]: [0]

6.4.5.7.4.5 ECC_CONTROL Register (Offset = 14h) [reset = 0h]

ECC_CONTROL is shown in Figure 11-1037 and described in Table 11-2387.

The Global Control Register controls the ECC control bits for the selected ECC RAM.

Table 6-116. ECC_CONTROL Instances

Instance	Physical Address
PRU-ICSS_0_ECC_CFG	20AA 7014h
PRU-ICSS_1_ECC_CFG	20AE 7014h

Figure 6-58. ECC_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED	ERROR_ONCE	FORCE_N_ROW	FORCE_DED	FORCE_SEC	ENABLE_RMW	ECC_CHECK	ECC_ENABLE
R-	R/W-	R/W-	R/W-	R/W-	R/W-	R/W-	R/W-

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-117. ECC_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	ERROR_ONCE	R/W	0h	If this bit is set, the FORCE_SEC/ FORCE_DED will inject an error to the specified row only once. The FORCE_SEC bit will be cleared once a writeback happens. If writeback is not enabled, this error will be cleared the cycle following the read when the data is corrected. For double-bit errors, the FORCE_DED bit will be cleared the cycle following the double-bit error. Any subsequent reads will not force an error.
5	FORCE_N_ROW	R/W	0h	Force single/double-bit error on the next RAM read.
4	FORCE_DED	R/W	0h	Force double-bit error. Cleared the cycle following the error if ERROR_ONCE is asserted.
3	FORCE_SEC	R/W	0h	Force single-bit error. Cleared on a writeback or the cycle following the error if ERROR_ONCE is asserted.
2	ENABLE_RMW	R/W	1h	Enable read-modify-write on partial word writes.
1	ECC_CHECK	R/W	1h	Enable ECC check. ECC is completely bypassed if both ECC_ENABLE and ECC_CHECK are 0.
0	ECC_ENABLE	R/W	1h	Enable ECC generation.

Table 6-118. Register Call Summary for ECC_CONTROL

PRU-ICSS PRU Cores
- ECC_CONTROL Register (Offset = 14h) [reset = 0h]: [0]
- PRU-ICSS_ECC_CFG Registers: [0]

6.4.5.7.4.6 ECC_ERROR_CONTROL1 Register (Offset = 18h) [reset = 0h]

ECC_ERROR_CONTROL1 is shown in Figure 11-1038 and described in Table 11-2390.

This register contains ECC error control bits for the selected ECC RAM.

Table 6-119. ECC_ERROR_CONTROL1 Instances

Instance	Physical Address
PRU-ICSS_0_ECC_CFG	20AA 7018h
PRU-ICSS_1_ECC_CFG	20AE 7018h

Figure 6-59. ECC_ERROR_CONTROL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_BIT1																ECC_ROW															
R/W-																R/W-															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-120. ECC_ERROR_CONTROL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	ECC_BIT1	R/W	0h	Column/ Data bit that needs to be flipped when FORCE_SEC or FORCE_DED is set.
15-0	ECC_ROW	R/W	0h	Row address where FORCE_SEC or FORCE_DED needs to be applied. This is ignored if FORCE_N_ROW is set.

Table 6-121. Register Call Summary for ECC_ERROR_CONTROL1

PRU-ICSS PRU Cores

- PRU-ICSS_ECC_CFG Registers: [0]
- ECC_ERROR_CONTROL1 Register (Offset = 18h) [reset = 0h]: [0]

6.4.5.7.4.7 ECC_ERROR_CONTROL2 Register (Offset = 1Ch) [reset = 0h]

ECC_ERROR_CONTROL2 is shown in Figure 11-1039 and described in Table 11-2393.

This register contains ECC error control bits for the selected ECC RAM.

Table 6-122. ECC_ERROR_CONTROL2 Instances

Instance	Physical Address
PRU-ICSS_0_ECC_CFG	20AA 701Ch
PRU-ICSS_1_ECC_CFG	20AE 701Ch

Figure 6-60. ECC_ERROR_CONTROL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ECC_BIT2															
R-																R/W-															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-123. ECC_ERROR_CONTROL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	ECC_BIT2	R/W	0h	Data bit that needs to be flipped when FORCE_DED is set

Table 6-124. Register Call Summary for ECC_ERROR_CONTROL2

PRU-ICSS PRU Cores
- PRU-ICSS_ECC_CFG Registers: [0]
- ECC_ERROR_CONTROL2 Register (Offset = 1Ch) [reset = 0h]: [0]

6.4.5.7.4.8 ECC_ERROR_STATUS1 Register (Offset = 20h) [reset = 0h]

ECC_ERROR_STATUS1 is shown in Figure 11-1040 and described in Table 11-2396.

This register contains ECC status bits for the selected ECC RAM.

Table 6-125. ECC_ERROR_STATUS1 Instances

Instance	Physical Address
PRU-ICSS_0_ECC_CFG	20AA 7020h
PRU-ICSS_1_ECC_CFG	20AE 7020h

Figure 6-61. ECC_ERROR_STATUS1 Register

31	30	29	28	27	26	25	24
ECC_ROW							
R-							
23	22	21	20	19	18	17	16
ECC_ROW							
R-							
15	14	13	12	11	10	9	8
RESERVED					CLR_ECC_OTHER	CLR_ECC_DED	CLR_ECC_SEC
R-					R/W1C	R/W1C	R/W1C
7	6	5	4	3	2	1	0
RESERVED					ECC_OTHER	ECC_DED	ECC_SEC
R-					R/W1S	R/W1S	R/W1S

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-126. ECC_ERROR_STATUS1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	ECC_ROW	R	0h	Indicates the row/address where the single or double-bit error occurred.
15-11	RESERVED	R	0h	Reserved
10	CLR_ECC_OTHER	R/W1C	0h	1 indicates a successive single-bit error. Writing a 1 clears the status bit.
9	CLR_ECC_DED	R/W1C	0h	1 indicates a pending double-bit error. Writing a 1 clears the status bit.
8	CLR_ECC_SEC	R/W1C	0h	1 indicates a pending single-bit error. Writing a 1 clears the status bit.
7-3	RESERVED	R	0h	Reserved
2	ECC_OTHER	R/W1S	0h	1 indicates that successive single-bit errors have occurred while a writeback is still pending. Software can also write a 1 to set the pending status and write a ‘1’ to the corresponding clear bit to clear the status.
1	ECC_DED	R/W1S	0h	1 indicates pending double-bit error. Since the double-bit error from the ECC logic is a pulsed interrupt, this is also a status set register. The software can also write a ‘1’ to set the pending status and write a ‘1’ to the corresponding clear bit to clear the status.
0	ECC_SEC	R/W1S	0h	1 indicates pending single-bit error status. Since the single-bit error from the ECC logic is a pulsed interrupt, this is also a status set register. The software can also write a ‘1’ to set the pending status and write a ‘1’ to the corresponding clear bit to clear the status.

Table 6-127. Register Call Summary for ECC_ERROR_STATUS1

PRU-ICSS PRU Cores
• PRU-ICSS_ECC_CFG Registers: [0] • ECC_ERROR_STATUS1 Register (Offset = 20h) [reset = 0h]: [0] • ECC Aggregator Interrupts: [0][1][2][3]

6.4.5.7.4.9 ECC_ERROR_STATUS2 Register (Offset = 24h) [reset = 0h]

ECC_ERROR_STATUS2 is shown in Figure 11-1041 and described in Table 11-2399.

This register contains ECC status bits for the selected ECC RAM.

Table 6-128. ECC_ERROR_STATUS2 Instances

Instance	Physical Address
PRU-ICSS_0_ECC_CFG	20AA 7024h
PRU-ICSS_1_ECC_CFG	20AE 7024h

Figure 6-62. ECC_ERROR_STATUS2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ECC_BIT1															
R-																R-															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-129. ECC_ERROR_STATUS2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	ECC_BIT1	R	0h	Indicates the bit position in the RAM data that is in error. For eg: a value of 1 indicates that bit 1 in the data is in error. This is valid only for single bit errors (sec).

Table 6-130. Register Call Summary for ECC_ERROR_STATUS2

PRU-ICSS PRU Cores

- PRU-ICSS_ECC_CFG Registers: [0]
- ECC Aggregator Interrupts: [0][1]
- ECC_ERROR_STATUS2 Register (Offset = 24h) [reset = 0h]: [0]

6.4.5.7.4.10 ECC_EOI Register (Offset = 3Ch) [reset = 0h]

ECC_EOI is shown in Figure 11-1042 and described in Table 11-2402.

This is the ECC_EOI register for the interrupt to the host.

Table 6-131. ECC_EOI Instances

Instance	Physical Address
PRU-ICSS_0_ECC_CFG	20AA 703Ch
PRU-ICSS_1_ECC_CFG	20AE 703Ch

Figure 6-63. ECC_EOI Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED							EOI_WR
R-							R/W1S

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -*n* = value after reset

Table 6-132. ECC_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EOI_WR	R/W1S	0h	Write to this register indicates that software has acknowledged the pending interrupt and the next interrupt can be sent to the host.

Table 6-133. Register Call Summary for ECC_EOI

PRU-ICSS PRU Cores

- PRU-ICSS_ECC_CFG Registers: [0][1]
- ECC Aggregator Interrupts: [0]
- ECC_EOI Register (Offset = 3Ch) [reset = 0h]: [0][1]

6.4.5.7.4.11 ECC_INT_STATUS_0 to ECC_INT_STATUS_15 Register (Offset = 40h to 7Ch) [reset = 0h]

ECC_INT_STATUS_0 to ECC_INT_STATUS_15 is shown in Figure 11-1043 and described in Table 11-2405.

These are the raw level interrupt status bits where each bit corresponds to the pending status from an ECC RAM. The bit associations with the ECC RAMs are assigned by the ECC aggregator module. There is 1 register for upto 32 ECC RAMs. The addresses increment for every additional 32-bit register required to hold the interrupt status for all the ECC RAMs (0 to N-1).

Table 6-134. ECC_INT_STATUS_0 to ECC_INT_STATUS_15 Instances

Instance	Physical Address
PRU-ICSS_0_ECC_CFG	20AA 7040h to 20AA 707Ch
PRU-ICSS_1_ECC_CFG	20AE 7040h to 20AE 707Ch

Figure 6-64. ECC_INT_STATUS_0 to ECC_INT_STATUS_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											SRC_INTR					
R-																											R-					

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-135. ECC_INT_STATUS_0 to ECC_INT_STATUS_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4-0	SRC_INTR	R	0h	Level interrupt status from each ECC RAM. • 0h: Not pending status. • 1h: Pending status.

Table 6-136. Register Call Summary for ECC_INT_STATUS_0

PRU-ICSS PRU Cores
- ECC_INT_STATUS_0 to ECC_INT_STATUS_15 Register (Offset = 40h to 7Ch) [reset = 0h]: [0]
- PRU-ICSS_ECC_CFG Registers: [0]

6.4.5.7.4.12 ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15 Register (Offset = 80h to BCh) [reset = 0h]

ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15 is shown in Figure 11-1044 and described in Table 11-2408.

These are interrupt enables associated with the interrupt from each of the ECC RAMs. Writing a 1 to a bit position in the register enables the interrupt from the associated ECC RAM. There is 1 register for upto 32 ECC RAMs. The addresses increment for every additional 32-bit register required for all ECC RAMs (0 to N-1).

Table 6-137. ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15 Instances

Instance	Physical Address
PRU-ICSS_0_ECC_CFG	20AA 7080h to 20AA 70BCh
PRU-ICSS_1_ECC_CFG	20AE 7080h to 20AE 70BCh

Figure 6-65. ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														ENABLE	
R-																														R/W1S	

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-138. ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4-0	ENABLE	R/W1S	0h	Write 1 to enable interrupt from the associated ECC RAM.

Table 6-139. Register Call Summary for ECC_INT_ENABLE_0

PRU-ICSS PRU Cores
- PRU-ICSS_ECC_CFG Registers: [0]
- ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15 Register (Offset = 80h to BCh) [reset = 0h]: [0]

6.4.5.7.4.13 ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15 Register (Offset = C0h to FCh) [reset = 0h]

ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15 is shown in Figure 11-1045 and described in Table 11-2411.

These are interrupt enable clear bits associated with the interrupt from each of the ECC RAMs. Writing a 1 to a bit position in the register disables the interrupt from the associated ECC RAM. There is 1 register for upto 32 ECC RAMs. The addresses increment for every additional 32-bit register required for all the ECC RAMs (0 to N-1).

Table 6-140. ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15 Instances

Instance	Physical Address
PRU-ICSS_0_ECC_CFG	20AA 70C0h to 20AA 70FCh
PRU-ICSS_1_ECC_CFG	20AE 70C0h to 20AE 70FCh

Figure 6-66. ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														ENABLE_CLEAR	
R-																														R/W1C	

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-141. ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4-0	ENABLE_CLEAR	R/W1C	0h	Write 1 to disable interrupt from the associated ECC RAM.

Table 6-142. Register Call Summary for ECC_INT_CLEAR_0

PRU-ICSS PRU Cores
- PRU-ICSS_ECC_CFG Registers: [0]
- ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15 Register (Offset = C0h to FCh) [reset = 0h]: [0]

6.4.5.8 PRU-ICSS_PRU_CTRL Registers

Table 6-144 lists the memory-mapped registers for the PRU-ICSS PRU0 and PRU1 cores. All register offset addresses not listed in Table 6-144 should be considered as reserved locations and the register contents should not be modified.

Table 6-143. PRU-ICSS_PRU_CTRL Instances

Instance	Base Address
PRU_ICSS_0_PRU0_CTRL	20AA 2000h
PRU_ICSS_0_PRU1_CTRL	20AA 4000h
PRU_ICSS_1_PRU0_CTRL	20AE 2000h
PRU_ICSS_1_PRU1_CTRL	20AE 4000h

Table 6-144. PRU-ICSS_PRU_CTRL Registers

Offset	Acronym	Register Name	PRU_ICSS_0_PRU0_CTRL Physical Address	PRU_ICSS_0_PRU1_CTRL Physical Address	PRU_ICSS_1_PRU0_CTRL Physical Address	PRU_ICSS_1_PRU1_CTRL Physical Address	Section
0h	PRU_CONTROL	Control register	20AA 2000h	20AA 4000h	20AE 2000h	20AE 4000h	Section 6.4.5.8.1
4h	PRU_STATUS	Status register	20AA 2004h	20AA 4004h	20AE 2004h	20AE 4004h	Section 6.4.5.8.2
8h	PRU_WAKEUP_EN	Wakeup enable register	20AA 2008h	20AA 4008h	20AE 2008h	20AE 4008h	Section 6.4.5.8.3
Ch	PRU_CYCLE	Cycle count	20AA 200Ch	20AA 400Ch	20AE 200Ch	20AE 400Ch	Section 6.4.5.8.4
10h	PRU_STALL	Stall count	20AA 2010h	20AA 4010h	20AE 2010h	20AE 4010h	Section 6.4.5.8.5
20h	PRU_CTBIR0	Constant Table Block Index Register 0	20AA 2020h	20AA 4020h	20AE 2020h	20AE 4020h	Section 6.4.5.8.6
24h	PRU_CTBIR1	Constant Table Block Index Register 1	20AA 2024h	20AA 4024h	20AE 2024h	20AE 4024h	Section 6.4.5.8.7
28h	PRU_CTPPR0	Constant Table Programmable Pointer Register 0	20AA 2028h	20AA 4028h	20AE 2028h	20AE 4028h	Section 6.4.5.8.8
2Ch	PRU_CTPPR1	Constant Table Programmable Pointer Register 1	20AA 202Ch	20AA 402Ch	20AE 202Ch	20AE 402Ch	Section 6.4.5.8.9

6.4.5.8.1 PRU_CONTROL Register (Offset = 0h) [reset = 1h]

PRU_CONTROL is shown in Figure 6-67 and described in Table 6-146.

CONTROL REGISTER

Table 6-145. PRU_CONTROL Instances

Instance	Physical Address
PRU_ICSS_0_PRU0_CTRL	20AA 2000h
PRU_ICSS_0_PRU1_CTRL	20AA 4000h
PRU_ICSS_1_PRU0_CTRL	20AE 2000h
PRU_ICSS_1_PRU1_CTRL	20AE 4000h

Figure 6-67. PRU_CONTROL Register

31	30	29	28	27	26	25	24
PCOUNTER_RST_VAL							
R/W-0h							
23	22	21	20	19	18	17	16
PCOUNTER_RST_VAL							
R/W-0h							
15	14	13	12	11	10	9	8
RUNSTATE	BIG_ENDIAN	RESERVED					SINGLE_STEP
R-0h	R-0h	R-0h					R/W-0h
7	6	5	4	3	2	1	0
RESERVED				COUNTER_ENABLE	SLEEPING	ENABLE	SOFT_RST_N
R-0h				R/W-0h	R/W-0h	R/W-0h	R-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-146. PRU_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	PCOUNTER_RST_VAL	R/W	0h	Program Counter Reset Value: This field controls the address where the PRU will start executing code from after it is taken out of reset.
15	RUNSTATE	R	0h	Run State: This bit indicates whether the PRU is currently executing an instruction or is halted. 0 = PRU is halted and host has access to the instruction RAM and debug registers regions. 1 = PRU is currently running and the host is locked out of the instruction RAM and debug registers regions. This bit is used by an external debug agent to know when the PRU has actually halted when waiting for a HALT instruction to execute, a single step to finish, or any other time when the pru_enable has been cleared.
14	BIG_ENDIAN	R	0h	
13-9	RESERVED	R	0h	Reserved
8	SINGLE_STEP	R/W	0h	Single Step Enable: This bit controls whether or not the PRU will only execute a single instruction when enabled. 0 = PRU will free run when enabled. 1 = PRU will execute a single instruction and then the pru_enable bit will be cleared. Note that this bit does not actually enable the PRU, it only sets the policy for how much code will be run after the PRU is enabled. The pru_enable bit must be explicitly asserted. It is legal to initialize both the single_step and pru_enable bits simultaneously. (Two independent writes are not required to cause the stated functionality.)
7-4	RESERVED	R	0h	Reserved
3	COUNTER_ENABLE	R/W	0h	PRU Cycle Counter Enable: Enables PRU cycle counters. 0 = Counters not enabled 1 = Counters enabled

Table 6-146. PRU_CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	SLEEPING	R/W	0h	PRU Sleep Indicator: This bit indicates whether or not the PRU is currently asleep. 0 = PRU is not asleep 1 = PRU is asleep If this bit is written to a 0, the PRU will be forced to power up from sleep mode.
1	ENABLE	R/W	0h	Processor Enable: This bit controls whether or not the PRU is allowed to fetch new instructions. 0 = PRU is disabled. 1 = PRU is enabled. If this bit is de-asserted while the PRU is currently running and has completed the initial cycle of a multi-cycle instruction (LBxO,SBxO,SCAN, etc.), the current instruction will be allowed to complete before the PRU pauses execution. Otherwise, the PRU will halt immediately. Because of the unpredictability timing sensitivity of the instruction execution loop, this bit is not a reliable indication of whether or not the PRU is currently running. The pru_state bit should be consulted for an absolute indication of the run state of the core. When the PRU is halted, its internal state remains coherent therefore this bit can be reasserted without issuing a software reset and the PRU will resume processing exactly where it left off in the instruction stream.
0	SOFT_RST_N	R/W	1h	Soft Reset: When this bit is cleared, the PRU will be reset. This bit is set back to 1 on the next cycle after it has been cleared.

Table 6-147. Register Call Summary for PRU_CONTROL

PRU-ICSS PRU Cores

- PRU-ICSS_PRU_CTRL Registers: [0]
- PRU_CONTROL Register (Offset = 0h) [reset = 1h]: [0]

6.4.5.8.2 PRU_STATUS Register (Offset = 4h) [reset = 0h]

PRU_STATUS is shown in Figure 6-68 and described in Table 6-149.

STATUS REGISTER

Table 6-148. PRU_STATUS Instances

Instance	Physical Address
PRU_ICSS_0_PRU0_CTRL	20AA 2004h
PRU_ICSS_0_PRU1_CTRL	20AA 4004h
PRU_ICSS_1_PRU0_CTRL	20AE 2004h
PRU_ICSS_1_PRU1_CTRL	20AE 4004h

Figure 6-68. PRU_STATUS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PCOUNTER															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-149. PRU_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	PCOUNTER	R	0h	Program Counter: This field is a registered (1 cycle delayed) reflection of the PRU program counter. Note that the PC is an instruction address where each instruction is a 32 bit word. This is not a byte address and to compute the byte address just multiply the PC by 4 (PC of 2 = byte address of 8h, or PC of 8 = byte address of 20h).

Table 6-150. Register Call Summary for PRU_STATUS

PRU-ICSS PRU Cores
- PRU-ICSS_PRU_CTRL Registers: [0]
- PRU_STATUS Register (Offset = 4h) [reset = 0h]: [0]

6.4.5.8.3 PRU_WAKEUP_EN Register (Offset = 8h) [reset = 0h]

PRU_WAKEUP_EN is shown in Figure 6-69 and described in Table 6-152.

WAKEUP ENABLE REGISTER

Table 6-151. PRU_WAKEUP_EN Instances

Instance	Physical Address
PRU_ICSS_0_PRU0_CTRL	20AA 2008h
PRU_ICSS_0_PRU1_CTRL	20AA 4008h
PRU_ICSS_1_PRU0_CTRL	20AE 2008h
PRU_ICSS_1_PRU1_CTRL	20AE 4008h

Figure 6-69. PRU_WAKEUP_EN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITWISE_ENABLES																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-152. PRU_WAKEUP_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITWISE_ENABLES	R/W	0h	Wakeup Enables: This field is ANDed with the incoming R31 status inputs (whose bit positions were specified in the stmap parameter) to produce a vector which is unary ORed to produce the status_wakeup source for the core. Setting any bit in this vector will allow the corresponding status input to wake up the core when it is asserted high. The PRU should set this enable vector prior to executing a SLP (sleep) instruction to ensure that the desired sources can wake up the core.

Table 6-153. Register Call Summary for PRU_WAKEUP_EN

PRU-ICSS PRU Cores

- PRU-ICSS_PRU_CTRL Registers: [0]
- PRU_WAKEUP_EN Register (Offset = 8h) [reset = 0h]: [0]

6.4.5.8.4 PRU_CYCLE Register (Offset = Ch) [reset = 0h]

PRU_CYCLE is shown in Figure 6-70 and described in Table 6-155.

CYCLE COUNT. This register counts the number of cycles for which the PRU has been enabled.

Table 6-154. PRU_CYCLE Instances

Instance	Physical Address
PRU_ICSS_0_PRU0_CTRL	20AA 200Ch
PRU_ICSS_0_PRU1_CTRL	20AA 400Ch
PRU_ICSS_1_PRU0_CTRL	20AE 200Ch
PRU_ICSS_1_PRU1_CTRL	20AE 400Ch

Figure 6-70. PRU_CYCLE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CYCLECOUNT																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-155. PRU_CYCLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CYCLECOUNT	R/W	0h	This value is incremented by 1 for every cycle during which the PRU is enabled and the counter is enabled (both bits ENABLE and COUNTENABLE set in the PRU control register). Counting halts while the PRU is disabled or counter is disabled, and resumes when re-eneabled. Counter clears the COUNTENABLE bit in the PRU control register when the count reaches FFFFFFFFh. (Count does does not wrap). The register can be read at any time. The register can be cleared when the counter or PRU is disabled. Clearing this register also clears the PRU Stall Count Register.

Table 6-156. Register Call Summary for PRU_CYCLE

PRU-ICSS PRU Cores
- PRU-ICSS_PRU_CTRL Registers: [0]
- PRU_CYCLE Register (Offset = Ch) [reset = 0h]: [0]

6.4.5.8.5 PRU_STALL Register (Offset = 10h) [reset = 0h]

PRU_STALL is shown in Figure 6-71 and described in Table 6-158.

STALL COUNT. This register counts the number of cycles for which the PRU has been enabled, but unable to fetch a new instruction. It is linked to the Cycle Count Register (0Ch) such that this register reflects the stall cycles measured over the same cycles as counted by the cycle count register. Thus the value of this register is always less than or equal to cycle count.

Table 6-157. PRU_STALL Instances

Instance	Physical Address
PRU_ICSS_0_PRU0_CTRL	20AA 2010h
PRU_ICSS_0_PRU1_CTRL	20AA 4010h
PRU_ICSS_1_PRU0_CTRL	20AE 2010h
PRU_ICSS_1_PRU1_CTRL	20AE 4010h

Figure 6-71. PRU_STALL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STALLCOUNT																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-158. PRU_STALL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STALLCOUNT	R/W	0h	This value is incremented by 1 for every cycle during which the PRU is enabled and the counter is enabled (both bits ENABLE and COUNTENABLE set in the PRU control register), and the PRU was unable to fetch a new instruction for any reason.

Table 6-159. Register Call Summary for PRU_STALL

PRU-ICSS PRU Cores
- PRU-ICSS_PRU_CTRL Registers: [0]
- PRU_STALL Register (Offset = 10h) [reset = 0h]: [0]

6.4.5.8.6 PRU_CTBIR0 Register (Offset = 20h) [reset = 0h]

PRU_CTBIR0 is shown in Figure 6-72 and described in Table 6-161.

CONSTANT TABLE BLOCK INDEX REGISTER 0. This register is used to set the block indices which are used to modify entries 24 and 25 in the PRU Constant Table. This register can be written by the PRU whenever it needs to change to a new base pointer for a block in the State Scratchpad RAM. This function is useful since the PRU is often processing multiple processing threads which require it to change contexts. The PRU can use this register to avoid requiring excessive amounts of code for repetitive context switching.

Table 6-160. PRU_CTBIR0 Instances

Instance	Physical Address
PRU_ICSS_0_PRU0_CTRL	20AA 2020h
PRU_ICSS_0_PRU1_CTRL	20AA 4020h
PRU_ICSS_1_PRU0_CTRL	20AE 2020h
PRU_ICSS_1_PRU1_CTRL	20AE 4020h

Figure 6-72. PRU_CTBIR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								C25_BLK_INDEX							
R-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								C24_BLK_INDEX							
R-0h								R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-161. PRU_CTBIR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	C25_BLK_INDEX	R/W	0h	PRU Constant Entry 25 Block Index: This field sets the value that will appear in bits 11:8 of entry 25 in the PRU Constant Table.
15-8	RESERVED	R	0h	Reserved
7-0	C24_BLK_INDEX	R/W	0h	PRU Constant Entry 24 Block Index: This field sets the value that will appear in bits 11:8 of entry 24 in the PRU Constant Table.

Table 6-162. Register Call Summary for PRU_CTBIR0

PRU-ICSS PRU Cores
- PRU-ICSS_PRU_CTRL Registers: [0]
- PRU_CTBIR0 Register (Offset = 20h) [reset = 0h]: [0]

6.4.5.8.7 PRU_CTBIR1 Register (Offset = 24h) [reset = 0h]

PRU_CTBIR1 is shown in Figure 6-73 and described in Table 6-164.

CONSTANT TABLE BLOCK INDEX REGISTER 1. This register is used to set the block indices which are used to modify entries 26 and 27 in the PRU Constant Table. This register can be written by the PRU whenever it needs to change to a new base pointer for a block in the State Scratchpad RAM. This function is useful since the PRU is often processing multiple processing threads which require it to change contexts. The PRU can use this register to avoid requiring excessive amounts of code for repetitive context switching.

Table 6-163. PRU_CTBIR1 Instances

Instance	Physical Address
PRU_ICSS_0_PRU0_CTRL	20AA 2024h
PRU_ICSS_0_PRU1_CTRL	20AA 4024h
PRU_ICSS_1_PRU0_CTRL	20AE 2024h
PRU_ICSS_1_PRU1_CTRL	20AE 4024h

Figure 6-73. PRU_CTBIR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								C27_BLK_INDEX							
R-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								C26_BLK_INDEX							
R-0h								R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-164. PRU_CTBIR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	C27_BLK_INDEX	R/W	0h	PRU Constant Entry 27 Block Index: This field sets the value that will appear in bits 11:8 of entry 27 in the PRU Constant Table.
15-8	RESERVED	R	0h	Reserved
7-0	C26_BLK_INDEX	R/W	0h	PRU Constant Entry 26 Block Index: This field sets the value that will appear in bits 11:8 of entry 26 in the PRU Constant Table.

Table 6-165. Register Call Summary for PRU_CTBIR1

PRU-ICSS PRU Cores
- PRU-ICSS_PRU_CTRL Registers: [0]
- PRU_CTBIR1 Register (Offset = 24h) [reset = 0h]: [0]

6.4.5.8.8 PRU_CTPPR0 Register (Offset = 28h) [reset = 0h]

PRU_CTPPR0 is shown in Figure 6-74 and described in Table 6-167.

CONSTANT TABLE PROGRAMMABLE POINTER REGISTER 0. This register allows the PRU to set up the 256-byte page index for entries 28 and 29 in the PRU Constant Table which serve as general purpose pointers which can be configured to point to any locations inside the session router address map. This register is useful when the PRU needs to frequently access certain structures inside the session router address space whose locations are not hard coded such as tables in scratchpad memory.

Table 6-166. PRU_CTPPR0 Instances

Instance	Physical Address
PRU_ICSS_0_PRU0_CTRL	20AA 2028h
PRU_ICSS_0_PRU1_CTRL	20AA 4028h
PRU_ICSS_1_PRU0_CTRL	20AE 2028h
PRU_ICSS_1_PRU1_CTRL	20AE 4028h

Figure 6-74. PRU_CTPPR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
C29_POINTER																C28_POINTER															
R/W-0h																R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-167. PRU_CTPPR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	C29_POINTER	R/W	0h	PRU Constant Entry 29 Pointer: This field sets the value that will appear in bits 23:8 of entry 29 in the PRU Constant Table.
15-0	C28_POINTER	R/W	0h	PRU Constant Entry 28 Pointer: This field sets the value that will appear in bits 23:8 of entry 28 in the PRU Constant Table.

Table 6-168. Register Call Summary for PRU_CTPPR0

PRU-ICSS PRU Cores
- PRU-ICSS_PRU_CTRL Registers: [0]
- PRU_CTPPR0 Register (Offset = 28h) [reset = 0h]: [0]

6.4.5.8.9 PRU_CTPPR1 Register (Offset = 2Ch) [reset = 0h]

PRU_CTPPR1 is shown in Figure 6-75 and described in Table 6-170.

CONSTANT TABLE PROGRAMMABLE POINTER REGISTER 1. This register functions the same as the PRU Constant Table Programmable Pointer Register 0 but allows the PRU to control entries 30 and 31 in the PRU Constant Table.

Table 6-169. PRU_CTPPR1 Instances

Instance	Physical Address
PRU_ICSS_0_PRU0_CTRL	20AA 202Ch
PRU_ICSS_0_PRU1_CTRL	20AA 402Ch
PRU_ICSS_1_PRU0_CTRL	20AE 202Ch
PRU_ICSS_1_PRU1_CTRL	20AE 402Ch

Figure 6-75. PRU_CTPPR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
C31_POINTER																C30_POINTER															
R/W-0h																R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-170. PRU_CTPPR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	C31_POINTER	R/W	0h	PRU Constant Entry 31 Pointer: This field sets the value that will appear in bits 23:8 of entry 31 in the PRU Constant Table.
15-0	C30_POINTER	R/W	0h	PRU Constant Entry 30 Pointer: This field sets the value that will appear in bits 23:8 of entry 30 in the PRU Constant Table.

Table 6-171. Register Call Summary for PRU_CTPPR1

PRU-ICSS PRU Cores
- PRU-ICSS_PRU_CTRL Registers: [0]
- PRU_CTPPR1 Register (Offset = 2Ch) [reset = 0h]: [0]

6.4.5.9 PRU_ICSS_PRU_DEBUG Registers

Table 6-173 lists the memory-mapped registers for the PRU_ICSS_PRU_DEBUG. All register offset addresses not listed in Table 6-173 should be considered as reserved locations and the register contents should not be modified.

Table 6-172. PRU-ICSS_PRU_DEBUG Instances

Instance	Base Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2400h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4400h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2400h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4400h

Table 6-173. PRU-ICSS_PRU_DEBUG Registers

Offset	Acronym	Register Name	PRU_ICSS_0_PRU_DEBUG_0 Physical Address	PRU_ICSS_0_PRU_DEBUG_1 Physical Address	PRU_ICSS_1_PRU_DEBUG_0 Physical Address	PRU_ICSS_1_PRU_DEBUG_1 Physical Address	Section
0h	PRU_ICSS_DBG_GPREG0	DEBUG PRU GENERAL PURPOSE REGISTER 0	20AA 2400h	20AA 4400h	20AE 2400h	20AE 4400h	Section 6.4.5.9.1
4h	PRU_ICSS_DBG_GPREG1	DEBUG PRU GENERAL PURPOSE REGISTER 1	20AA 2404h	20AA 4404h	20AE 2404h	20AE 4404h	Section 6.4.5.9.2
8h	PRU_ICSS_DBG_GPREG2	DEBUG PRU GENERAL PURPOSE REGISTER 2	20AA 2408h	20AA 4408h	20AE 2408h	20AE 4408h	Section 6.4.5.9.3
Ch	PRU_ICSS_DBG_GPREG3	DEBUG PRU GENERAL PURPOSE REGISTER 3	20AA 240Ch	20AA 440Ch	20AE 240Ch	20AE 440Ch	Section 6.4.5.9.4
10h	PRU_ICSS_DBG_GPREG4	DEBUG PRU GENERAL PURPOSE REGISTER 4	20AA 2410h	20AA 4410h	20AE 2410h	20AE 4410h	Section 6.4.5.9.5
14h	PRU_ICSS_DBG_GPREG5	DEBUG PRU GENERAL PURPOSE REGISTER 5	20AA 2414h	20AA 4414h	20AE 2414h	20AE 4414h	Section 6.4.5.9.6
18h	PRU_ICSS_DBG_GPREG6	DEBUG PRU GENERAL PURPOSE REGISTER 6	20AA 2418h	20AA 4418h	20AE 2418h	20AE 4418h	Section 6.4.5.9.7
1Ch	PRU_ICSS_DBG_GPREG7	DEBUG PRU GENERAL PURPOSE REGISTER 7	20AA 241Ch	20AA 441Ch	20AE 241Ch	20AE 441Ch	Section 6.4.5.9.8
20h	PRU_ICSS_DBG_GPREG8	DEBUG PRU GENERAL PURPOSE REGISTER 8	20AA 2420h	20AA 4420h	20AE 2420h	20AE 4420h	Section 6.4.5.9.9
24h	PRU_ICSS_DBG_GPREG9	DEBUG PRU GENERAL PURPOSE REGISTER 9	20AA 2424h	20AA 4424h	20AE 2424h	20AE 4424h	Section 6.4.5.9.10
28h	PRU_ICSS_DBG_GPREG10	DEBUG PRU GENERAL PURPOSE REGISTER 10	20AA 2428h	20AA 4428h	20AE 2428h	20AE 4428h	Section 6.4.5.9.11
2Ch	PRU_ICSS_DBG_GPREG11	DEBUG PRU GENERAL PURPOSE REGISTER 11	20AA 242Ch	20AA 442Ch	20AE 242Ch	20AE 442Ch	Section 6.4.5.9.12
30h	PRU_ICSS_DBG_GPREG12	DEBUG PRU GENERAL PURPOSE REGISTER 12	20AA 2430h	20AA 4430h	20AE 2430h	20AE 4430h	Section 6.4.5.9.13
34h	PRU_ICSS_DBG_GPREG13	DEBUG PRU GENERAL PURPOSE REGISTER 13	20AA 2434h	20AA 4434h	20AE 2434h	20AE 4434h	Section 6.4.5.9.14
38h	PRU_ICSS_DBG_GPREG14	DEBUG PRU GENERAL PURPOSE REGISTER 14	20AA 2438h	20AA 4438h	20AE 2438h	20AE 4438h	Section 6.4.5.9.15
3Ch	PRU_ICSS_DBG_GPREG15	DEBUG PRU GENERAL PURPOSE REGISTER 15	20AA 243Ch	20AA 443Ch	20AE 243Ch	20AE 443Ch	Section 6.4.5.9.16
40h	PRU_ICSS_DBG_GPREG16	DEBUG PRU GENERAL PURPOSE REGISTER 16	20AA 2440h	20AA 4440h	20AE 2440h	20AE 4440h	Section 6.4.5.9.17
44h	PRU_ICSS_DBG_GPREG17	DEBUG PRU GENERAL PURPOSE REGISTER 17	20AA 2444h	20AA 4444h	20AE 2444h	20AE 4444h	Section 6.4.5.9.18
48h	PRU_ICSS_DBG_GPREG18	DEBUG PRU GENERAL PURPOSE REGISTER 18	20AA 2448h	20AA 4448h	20AE 2448h	20AE 4448h	Section 6.4.5.9.19

Table 6-173. PRU-ICSS_PRU_DEBUG Registers (continued)

Offset	Acronym	Register Name	PRU_ICSS_0_PRU_DEBUG_0 Physical Address	PRU_ICSS_0_PRU_DEBUG_1 Physical Address	PRU_ICSS_1_PRU_DEBUG_0 Physical Address	PRU_ICSS_1_PRU_DEBUG_1 Physical Address	Section
4Ch	PRU_ICSS_DBG_GPREG19	DEBUG PRU GENERAL PURPOSE REGISTER 19	20AA 244Ch	20AA 444Ch	20AE 244Ch	20AE 444Ch	Section 6.4.5.9.20
50h	PRU_ICSS_DBG_GPREG20	DEBUG PRU GENERAL PURPOSE REGISTER 20	20AA 2450h	20AA 4450h	20AE 2450h	20AE 4450h	Section 6.4.5.9.21
54h	PRU_ICSS_DBG_GPREG21	DEBUG PRU GENERAL PURPOSE REGISTER 21	20AA 2454h	20AA 4454h	20AE 2454h	20AE 4454h	Section 6.4.5.9.22
58h	PRU_ICSS_DBG_GPREG22	DEBUG PRU GENERAL PURPOSE REGISTER 22	20AA 2458h	20AA 4458h	20AE 2458h	20AE 4458h	Section 6.4.5.9.23
5Ch	PRU_ICSS_DBG_GPREG23	DEBUG PRU GENERAL PURPOSE REGISTER 23	20AA 245Ch	20AA 445Ch	20AE 245Ch	20AE 445Ch	Section 6.4.5.9.24
60h	PRU_ICSS_DBG_GPREG24	DEBUG PRU GENERAL PURPOSE REGISTER 24	20AA 2460h	20AA 4460h	20AE 2460h	20AE 4460h	Section 6.4.5.9.25
64h	PRU_ICSS_DBG_GPREG25	DEBUG PRU GENERAL PURPOSE REGISTER 25	20AA 2464h	20AA 4464h	20AE 2464h	20AE 4464h	Section 6.4.5.9.26
68h	PRU_ICSS_DBG_GPREG26	DEBUG PRU GENERAL PURPOSE REGISTER 26	20AA 2468h	20AA 4468h	20AE 2468h	20AE 4468h	Section 6.4.5.9.27
6Ch	PRU_ICSS_DBG_GPREG27	DEBUG PRU GENERAL PURPOSE REGISTER 27	20AA 246Ch	20AA 446Ch	20AE 246Ch	20AE 446Ch	Section 6.4.5.9.28
70h	PRU_ICSS_DBG_GPREG28	DEBUG PRU GENERAL PURPOSE REGISTER 28	20AA 2470h	20AA 4470h	20AE 2470h	20AE 4470h	Section 6.4.5.9.29
74h	PRU_ICSS_DBG_GPREG29	DEBUG PRU GENERAL PURPOSE REGISTER 29	20AA 2474h	20AA 4474h	20AE 2474h	20AE 4474h	Section 6.4.5.9.30
78h	PRU_ICSS_DBG_GPREG30	DEBUG PRU GENERAL PURPOSE REGISTER 30	20AA 2478h	20AA 4478h	20AE 2478h	20AE 4478h	Section 6.4.5.9.31
7Ch	PRU_ICSS_DBG_GPREG31	DEBUG PRU GENERAL PURPOSE REGISTER 31	20AA 247Ch	20AA 447Ch	20AE 247Ch	20AE 447Ch	Section 6.4.5.9.32
80h	PRU_ICSS_DBG_CT_REG0	DEBUG PRU CONSTANTS TABLE ENTRY 0	20AA 2480h	20AA 4480h	20AE 2480h	20AE 4480h	Section 6.4.5.9.33
84h	PRU_ICSS_DBG_CT_REG1	DEBUG PRU CONSTANTS TABLE ENTRY 1	20AA 2484h	20AA 4484h	20AE 2484h	20AE 4484h	Section 6.4.5.9.34
88h	PRU_ICSS_DBG_CT_REG2	DEBUG PRU CONSTANTS TABLE ENTRY 2	20AA 2488h	20AA 4488h	20AE 2488h	20AE 4488h	Section 6.4.5.9.35
8Ch	PRU_ICSS_DBG_CT_REG3	DEBUG PRU CONSTANTS TABLE ENTRY 3	20AA 248Ch	20AA 448Ch	20AE 248Ch	20AE 448Ch	Section 6.4.5.9.36
90h	PRU_ICSS_DBG_CT_REG4	DEBUG PRU CONSTANTS TABLE ENTRY 4	20AA 2490h	20AA 4490h	20AE 2490h	20AE 4490h	Section 6.4.5.9.37
94h	PRU_ICSS_DBG_CT_REG5	DEBUG PRU CONSTANTS TABLE ENTRY 5	20AA 2494h	20AA 4494h	20AE 2494h	20AE 4494h	Section 6.4.5.9.38
98h	PRU_ICSS_DBG_CT_REG6	DEBUG PRU CONSTANTS TABLE ENTRY 6	20AA 2498h	20AA 4498h	20AE 2498h	20AE 4498h	Section 6.4.5.9.39
9Ch	PRU_ICSS_DBG_CT_REG7	DEBUG PRU CONSTANTS TABLE ENTRY 7	20AA 249Ch	20AA 449Ch	20AE 249Ch	20AE 449Ch	Section 6.4.5.9.40
A0h	PRU_ICSS_DBG_CT_REG8	DEBUG PRU CONSTANTS TABLE ENTRY 8	20AA 24A0h	20AA 44A0h	20AE 24A0h	20AE 44A0h	Section 6.4.5.9.41

Table 6-173. PRU-ICSS_PRU_DEBUG Registers (continued)

Offset	Acronym	Register Name	PRU_ICSS_0_PRU_DEBUG_0 Physical Address	PRU_ICSS_0_PRU_DEBUG_1 Physical Address	PRU_ICSS_1_PRU_DEBUG_0 Physical Address	PRU_ICSS_1_PRU_DEBUG_1 Physical Address	Section
A4h	PRU_ICSS_DBG_CT_REG9	DEBUG PRU CONSTANTS TABLE ENTRY 9	20AA 24A4h	20AA 44A4h	20AE 24A4h	20AE 44A4h	Section 6.4.5.9.42
A8h	PRU_ICSS_DBG_CT_REG10	DEBUG PRU CONSTANTS TABLE ENTRY 10	20AA 24A8h	20AA 44A8h	20AE 24A8h	20AE 44A8h	Section 6.4.5.9.43
ACh	PRU_ICSS_DBG_CT_REG11	DEBUG PRU CONSTANTS TABLE ENTRY 11	20AA 24ACh	20AA 44ACh	20AE 24ACh	20AE 44ACh	Section 6.4.5.9.44
B0h	PRU_ICSS_DBG_CT_REG12	DEBUG PRU CONSTANTS TABLE ENTRY 12	20AA 24B0h	20AA 44B0h	20AE 24B0h	20AE 44B0h	Section 6.4.5.9.45
B4h	PRU_ICSS_DBG_CT_REG13	DEBUG PRU CONSTANTS TABLE ENTRY 13	20AA 24B4h	20AA 44B4h	20AE 24B4h	20AE 44B4h	Section 6.4.5.9.46
B8h	PRU_ICSS_DBG_CT_REG14	DEBUG PRU CONSTANTS TABLE ENTRY 14	20AA 24B8h	20AA 44B8h	20AE 24B8h	20AE 44B8h	Section 6.4.5.9.47
BCh	PRU_ICSS_DBG_CT_REG15	DEBUG PRU CONSTANTS TABLE ENTRY 15	20AA 24BCh	20AA 44BCh	20AE 24BCh	20AE 44BCh	Section 6.4.5.9.48
C0h	PRU_ICSS_DBG_CT_REG16	DEBUG PRU CONSTANTS TABLE ENTRY 16	20AA 24C0h	20AA 44C0h	20AE 24C0h	20AE 44C0h	Section 6.4.5.9.49
C4h	PRU_ICSS_DBG_CT_REG17	DEBUG PRU CONSTANTS TABLE ENTRY 17	20AA 24C4h	20AA 44C4h	20AE 24C4h	20AE 44C4h	Section 6.4.5.9.50
C8h	PRU_ICSS_DBG_CT_REG18	DEBUG PRU CONSTANTS TABLE ENTRY 18	20AA 24C8h	20AA 44C8h	20AE 24C8h	20AE 44C8h	Section 6.4.5.9.51
CCh	PRU_ICSS_DBG_CT_REG19	DEBUG PRU CONSTANTS TABLE ENTRY 19	20AA 24CCh	20AA 44CCh	20AE 24CCh	20AE 44CCh	Section 6.4.5.9.52
D0h	PRU_ICSS_DBG_CT_REG20	DEBUG PRU CONSTANTS TABLE ENTRY 20	20AA 24D0h	20AA 44D0h	20AE 24D0h	20AE 44D0h	Section 6.4.5.9.53
D4h	PRU_ICSS_DBG_CT_REG21	DEBUG PRU CONSTANTS TABLE ENTRY 21	20AA 24D4h	20AA 44D4h	20AE 24D4h	20AE 44D4h	Section 6.4.5.9.54
D8h	PRU_ICSS_DBG_CT_REG22	DEBUG PRU CONSTANTS TABLE ENTRY 22	20AA 24D8h	20AA 44D8h	20AE 24D8h	20AE 44D8h	Section 6.4.5.9.55
DCh	PRU_ICSS_DBG_CT_REG23	DEBUG PRU CONSTANTS TABLE ENTRY 23	20AA 24DCh	20AA 44DCh	20AE 24DCh	20AE 44DCh	Section 6.4.5.9.56
E0h	PRU_ICSS_DBG_CT_REG24	DEBUG PRU CONSTANTS TABLE ENTRY 24	20AA 24E0h	20AA 44E0h	20AE 24E0h	20AE 44E0h	Section 6.4.5.9.57
E4h	PRU_ICSS_DBG_CT_REG25	DEBUG PRU CONSTANTS TABLE ENTRY 25	20AA 24E4h	20AA 44E4h	20AE 24E4h	20AE 44E4h	Section 6.4.5.9.58
E8h	PRU_ICSS_DBG_CT_REG26	DEBUG PRU CONSTANTS TABLE ENTRY 26	20AA 24E8h	20AA 44E8h	20AE 24E8h	20AE 44E8h	Section 6.4.5.9.59
ECh	PRU_ICSS_DBG_CT_REG27	DEBUG PRU CONSTANTS TABLE ENTRY 27	20AA 24ECh	20AA 44ECh	20AE 24ECh	20AE 44ECh	Section 6.4.5.9.60

Table 6-173. PRU-ICSS_PRU_DEBUG Registers (continued)

Offset	Acronym	Register Name	PRU_ICSS_0_PRU_DEBUG_0 Physical Address	PRU_ICSS_0_PRU_DEBUG_1 Physical Address	PRU_ICSS_1_PRU_DEBUG_0 Physical Address	PRU_ICSS_1_PRU_DEBUG_1 Physical Address	Section
F0h	PRU_ICSS_DBG_CT_REG28	DEBUG PRU CONSTANTS TABLE ENTRY 28	20AA 24F0h	20AA 44F0h	20AE 24F0h	20AE 44F0h	Section 6.4.5.9.61
F4h	PRU_ICSS_DBG_CT_REG29	DEBUG PRU CONSTANTS TABLE ENTRY 29	20AA 24F4h	20AA 44F4h	20AE 24F4h	20AE 44F4h	Section 6.4.5.9.62
F8h	PRU_ICSS_DBG_CT_REG30	DEBUG PRU CONSTANTS TABLE ENTRY 30	20AA 24F8h	20AA 44F8h	20AE 24F8h	20AE 44F8h	Section 6.4.5.9.63
FCh	PRU_ICSS_DBG_CT_REG31	DEBUG PRU CONSTANTS TABLE ENTRY 31	20AA 24FCh	20AA 44FCh	20AE 24FCh	20AE 44FCh	Section 6.4.5.9.64

6.4.5.9.1 PRU_ICSS_DBG_GPREG0 Register (Offset = 0h) [reset = 0h]

PRU_ICSS_DBG_GPREG0 is shown in Figure 6-76 and described in Table 6-175.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 0. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-174. PRU_ICSS_DBG_GPREG0 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2400h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4400h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2400h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4400h

Figure 6-76. PRU_ICSS_DBG_GPREG0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG0																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-175. PRU_ICSS_DBG_GPREG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG0	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-176. Register Call Summary for PRU_ICSS_DBG_GPREG0

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_GPREG0 Register (Offset = 0h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.2 PRU_ICSS_DBG_GPREG1 Register (Offset = 4h) [reset = 0h]

PRU_ICSS_DBG_GPREG1 is shown in Figure 6-77 and described in Table 6-178.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 1. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-177. PRU_ICSS_DBG_GPREG1 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2404h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4404h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2404h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4404h

Figure 6-77. PRU_ICSS_DBG_GPREG1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG1																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-178. PRU_ICSS_DBG_GPREG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG1	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-179. Register Call Summary for PRU_ICSS_DBG_GPREG1

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_GPREG1 Register (Offset = 4h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.3 PRU_ICSS_DBG_GPREG2 Register (Offset = 8h) [reset = 0h]

PRU_ICSS_DBG_GPREG2 is shown in Figure 6-78 and described in Table 6-181.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 2. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-180. PRU_ICSS_DBG_GPREG2 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2408h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4408h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2408h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4408h

Figure 6-78. PRU_ICSS_DBG_GPREG2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG2																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-181. PRU_ICSS_DBG_GPREG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG2	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-182. Register Call Summary for PRU_ICSS_DBG_GPREG2

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_GPREG2 Register (Offset = 8h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.4 PRU_ICSS_DBG_GPREG3 Register (Offset = Ch) [reset = 0h]

PRU_ICSS_DBG_GPREG3 is shown in Figure 6-79 and described in Table 6-184.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 3. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-183. PRU_ICSS_DBG_GPREG3 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 240Ch
PRU_ICSS_0_PRU_DEBUG_1	20AA 440Ch
PRU_ICSS_1_PRU_DEBUG_0	20AE 240Ch
PRU_ICSS_1_PRU_DEBUG_1	20AE 440Ch

Figure 6-79. PRU_ICSS_DBG_GPREG3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG3																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-184. PRU_ICSS_DBG_GPREG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG3	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-185. Register Call Summary for PRU_ICSS_DBG_GPREG3

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_GPREG3 Register (Offset = Ch) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.5 PRU_ICSS_DBG_GPREG4 Register (Offset = 10h) [reset = 0h]

PRU_ICSS_DBG_GPREG4 is shown in Figure 6-80 and described in Table 6-187.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 4. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-186. PRU_ICSS_DBG_GPREG4 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2410h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4410h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2410h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4410h

Figure 6-80. PRU_ICSS_DBG_GPREG4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG4																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-187. PRU_ICSS_DBG_GPREG4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG4	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-188. Register Call Summary for PRU_ICSS_DBG_GPREG4

PRU-ICSS PRU Cores
- PRU_ICSS_PRU_DEBUG Registers: [0]
- PRU_ICSS_DBG_GPREG4 Register (Offset = 10h) [reset = 0h]: [0]

6.4.5.9.6 PRU_ICSS_DBG_GPREG5 Register (Offset = 14h) [reset = 0h]

PRU_ICSS_DBG_GPREG5 is shown in Figure 6-81 and described in Table 6-190.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 5. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-189. PRU_ICSS_DBG_GPREG5 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2414h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4414h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2414h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4414h

Figure 6-81. PRU_ICSS_DBG_GPREG5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG5																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-190. PRU_ICSS_DBG_GPREG5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG5	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-191. Register Call Summary for PRU_ICSS_DBG_GPREG5

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_GPREG5 Register (Offset = 14h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.7 PRU_ICSS_DBG_GPREG6 Register (Offset = 18h) [reset = 0h]

PRU_ICSS_DBG_GPREG6 is shown in Figure 6-82 and described in Table 6-193.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 6. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-192. PRU_ICSS_DBG_GPREG6 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2418h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4418h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2418h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4418h

Figure 6-82. PRU_ICSS_DBG_GPREG6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG6																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-193. PRU_ICSS_DBG_GPREG6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG6	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-194. Register Call Summary for PRU_ICSS_DBG_GPREG6

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_GPREG6 Register (Offset = 18h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.8 PRU_ICSS_DBG_GPREG7 Register (Offset = 1Ch) [reset = 0h]

PRU_ICSS_DBG_GPREG7 is shown in Figure 6-83 and described in Table 6-196.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 7. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-195. PRU_ICSS_DBG_GPREG7 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 241Ch
PRU_ICSS_0_PRU_DEBUG_1	20AA 441Ch
PRU_ICSS_1_PRU_DEBUG_0	20AE 241Ch
PRU_ICSS_1_PRU_DEBUG_1	20AE 441Ch

Figure 6-83. PRU_ICSS_DBG_GPREG7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG7																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-196. PRU_ICSS_DBG_GPREG7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG7	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-197. Register Call Summary for PRU_ICSS_DBG_GPREG7

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_GPREG7 Register (Offset = 1Ch) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.9 PRU_ICSS_DBG_GPREG8 Register (Offset = 20h) [reset = 0h]

PRU_ICSS_DBG_GPREG8 is shown in Figure 6-84 and described in Table 6-199.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 8. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-198. PRU_ICSS_DBG_GPREG8 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2420h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4420h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2420h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4420h

Figure 6-84. PRU_ICSS_DBG_GPREG8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG8																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-199. PRU_ICSS_DBG_GPREG8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG8	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-200. Register Call Summary for PRU_ICSS_DBG_GPREG8

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_GPREG8 Register (Offset = 20h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.10 PRU_ICSS_DBG_GPREG9 Register (Offset = 24h) [reset = 0h]

PRU_ICSS_DBG_GPREG9 is shown in Figure 6-85 and described in Table 6-202.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 9. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-201. PRU_ICSS_DBG_GPREG9 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2424h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4424h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2424h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4424h

Figure 6-85. PRU_ICSS_DBG_GPREG9 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG9																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-202. PRU_ICSS_DBG_GPREG9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG9	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-203. Register Call Summary for PRU_ICSS_DBG_GPREG9

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_GPREG9 Register (Offset = 24h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.11 PRU_ICSS_DBG_GPREG10 Register (Offset = 28h) [reset = 0h]

PRU_ICSS_DBG_GPREG10 is shown in Figure 6-86 and described in Table 6-205.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 10. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-204. PRU_ICSS_DBG_GPREG10 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2428h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4428h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2428h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4428h

Figure 6-86. PRU_ICSS_DBG_GPREG10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG10																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-205. PRU_ICSS_DBG_GPREG10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG10	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-206. Register Call Summary for PRU_ICSS_DBG_GPREG10

PRU-ICSS PRU Cores
- PRU_ICSS_PRU_DEBUG Registers: [0]
- PRU_ICSS_DBG_GPREG10 Register (Offset = 28h) [reset = 0h]: [0]

6.4.5.9.12 PRU_ICSS_DBG_GPREG11 Register (Offset = 2Ch) [reset = 0h]

PRU_ICSS_DBG_GPREG11 is shown in Figure 6-87 and described in Table 6-208.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 11. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-207. PRU_ICSS_DBG_GPREG11 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 242Ch
PRU_ICSS_0_PRU_DEBUG_1	20AA 442Ch
PRU_ICSS_1_PRU_DEBUG_0	20AE 242Ch
PRU_ICSS_1_PRU_DEBUG_1	20AE 442Ch

Figure 6-87. PRU_ICSS_DBG_GPREG11 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG11																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-208. PRU_ICSS_DBG_GPREG11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG11	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-209. Register Call Summary for PRU_ICSS_DBG_GPREG11

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_GPREG11 Register (Offset = 2Ch) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.13 PRU_ICSS_DBG_GPREG12 Register (Offset = 30h) [reset = 0h]

PRU_ICSS_DBG_GPREG12 is shown in Figure 6-88 and described in Table 6-211.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 12. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-210. PRU_ICSS_DBG_GPREG12 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2430h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4430h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2430h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4430h

Figure 6-88. PRU_ICSS_DBG_GPREG12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG12																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-211. PRU_ICSS_DBG_GPREG12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG12	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-212. Register Call Summary for PRU_ICSS_DBG_GPREG12

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_GPREG12 Register (Offset = 30h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.14 PRU_ICSS_DBG_GPREG13 Register (Offset = 34h) [reset = 0h]

PRU_ICSS_DBG_GPREG13 is shown in Figure 6-89 and described in Table 6-214.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 13. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-213. PRU_ICSS_DBG_GPREG13 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2434h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4434h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2434h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4434h

Figure 6-89. PRU_ICSS_DBG_GPREG13 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG13																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-214. PRU_ICSS_DBG_GPREG13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG13	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-215. Register Call Summary for PRU_ICSS_DBG_GPREG13

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_GPREG13 Register (Offset = 34h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.15 PRU_ICSS_DBG_GPREG14 Register (Offset = 38h) [reset = 0h]

PRU_ICSS_DBG_GPREG14 is shown in Figure 6-90 and described in Table 6-217.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 14. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-216. PRU_ICSS_DBG_GPREG14 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2438h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4438h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2438h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4438h

Figure 6-90. PRU_ICSS_DBG_GPREG14 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG14																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-217. PRU_ICSS_DBG_GPREG14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG14	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-218. Register Call Summary for PRU_ICSS_DBG_GPREG14

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_GPREG14 Register (Offset = 38h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.16 PRU_ICSS_DBG_GPREG15 Register (Offset = 3Ch) [reset = 0h]

PRU_ICSS_DBG_GPREG15 is shown in Figure 6-91 and described in Table 6-220.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 15. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-219. PRU_ICSS_DBG_GPREG15 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 243Ch
PRU_ICSS_0_PRU_DEBUG_1	20AA 443Ch
PRU_ICSS_1_PRU_DEBUG_0	20AE 243Ch
PRU_ICSS_1_PRU_DEBUG_1	20AE 443Ch

Figure 6-91. PRU_ICSS_DBG_GPREG15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG15																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-220. PRU_ICSS_DBG_GPREG15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG15	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-221. Register Call Summary for PRU_ICSS_DBG_GPREG15

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_GPREG15 Register (Offset = 3Ch) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.17 PRU_ICSS_DBG_GPREG16 Register (Offset = 40h) [reset = 0h]

PRU_ICSS_DBG_GPREG16 is shown in Figure 6-92 and described in Table 6-223.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 16. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-222. PRU_ICSS_DBG_GPREG16 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2440h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4440h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2440h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4440h

Figure 6-92. PRU_ICSS_DBG_GPREG16 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG16																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-223. PRU_ICSS_DBG_GPREG16 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG16	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-224. Register Call Summary for PRU_ICSS_DBG_GPREG16

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_GPREG16 Register (Offset = 40h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.18 PRU_ICSS_DBG_GPREG17 Register (Offset = 44h) [reset = 0h]

PRU_ICSS_DBG_GPREG17 is shown in Figure 6-93 and described in Table 6-226.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 17. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-225. PRU_ICSS_DBG_GPREG17 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2444h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4444h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2444h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4444h

Figure 6-93. PRU_ICSS_DBG_GPREG17 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG17																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-226. PRU_ICSS_DBG_GPREG17 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG17	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-227. Register Call Summary for PRU_ICSS_DBG_GPREG17

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_GPREG17 Register (Offset = 44h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.19 PRU_ICSS_DBG_GPREG18 Register (Offset = 48h) [reset = 0h]

PRU_ICSS_DBG_GPREG18 is shown in Figure 6-94 and described in Table 6-229.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 18. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-228. PRU_ICSS_DBG_GPREG18 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2448h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4448h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2448h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4448h

Figure 6-94. PRU_ICSS_DBG_GPREG18 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG18																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-229. PRU_ICSS_DBG_GPREG18 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG18	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-230. Register Call Summary for PRU_ICSS_DBG_GPREG18

PRU-ICSS PRU Cores
- PRU_ICSS_PRU_DEBUG Registers: [0]
- PRU_ICSS_DBG_GPREG18 Register (Offset = 48h) [reset = 0h]: [0]

6.4.5.9.20 PRU_ICSS_DBG_GPREG19 Register (Offset = 4Ch) [reset = 0h]

PRU_ICSS_DBG_GPREG19 is shown in Figure 6-95 and described in Table 6-232.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 19. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-231. PRU_ICSS_DBG_GPREG19 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 244Ch
PRU_ICSS_0_PRU_DEBUG_1	20AA 444Ch
PRU_ICSS_1_PRU_DEBUG_0	20AE 244Ch
PRU_ICSS_1_PRU_DEBUG_1	20AE 444Ch

Figure 6-95. PRU_ICSS_DBG_GPREG19 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG19																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-232. PRU_ICSS_DBG_GPREG19 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG19	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-233. Register Call Summary for PRU_ICSS_DBG_GPREG19

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_GPREG19 Register (Offset = 4Ch) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.21 PRU_ICSS_DBG_GPREG20 Register (Offset = 50h) [reset = 0h]

PRU_ICSS_DBG_GPREG20 is shown in Figure 6-96 and described in Table 6-235.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 20. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-234. PRU_ICSS_DBG_GPREG20 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2450h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4450h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2450h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4450h

Figure 6-96. PRU_ICSS_DBG_GPREG20 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG20																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-235. PRU_ICSS_DBG_GPREG20 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG20	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-236. Register Call Summary for PRU_ICSS_DBG_GPREG20

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_GPREG20 Register (Offset = 50h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.22 PRU_ICSS_DBG_GPREG21 Register (Offset = 54h) [reset = 0h]

PRU_ICSS_DBG_GPREG21 is shown in Figure 6-97 and described in Table 6-238.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 21. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-237. PRU_ICSS_DBG_GPREG21 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2454h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4454h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2454h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4454h

Figure 6-97. PRU_ICSS_DBG_GPREG21 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG21																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-238. PRU_ICSS_DBG_GPREG21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG21	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-239. Register Call Summary for PRU_ICSS_DBG_GPREG21

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_GPREG21 Register (Offset = 54h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.23 PRU_ICSS_DBG_GPREG22 Register (Offset = 58h) [reset = 0h]

PRU_ICSS_DBG_GPREG22 is shown in Figure 6-98 and described in Table 6-241.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 22. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-240. PRU_ICSS_DBG_GPREG22 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2458h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4458h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2458h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4458h

Figure 6-98. PRU_ICSS_DBG_GPREG22 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG22																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-241. PRU_ICSS_DBG_GPREG22 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG22	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-242. Register Call Summary for PRU_ICSS_DBG_GPREG22

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_GPREG22 Register (Offset = 58h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.24 PRU_ICSS_DBG_GPREG23 Register (Offset = 5Ch) [reset = 0h]

PRU_ICSS_DBG_GPREG23 is shown in Figure 6-99 and described in Table 6-244.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 23. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-243. PRU_ICSS_DBG_GPREG23 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 245Ch
PRU_ICSS_0_PRU_DEBUG_1	20AA 445Ch
PRU_ICSS_1_PRU_DEBUG_0	20AE 245Ch
PRU_ICSS_1_PRU_DEBUG_1	20AE 445Ch

Figure 6-99. PRU_ICSS_DBG_GPREG23 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG23																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-244. PRU_ICSS_DBG_GPREG23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG23	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-245. Register Call Summary for PRU_ICSS_DBG_GPREG23

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_GPREG23 Register (Offset = 5Ch) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.25 PRU_ICSS_DBG_GPREG24 Register (Offset = 60h) [reset = 0h]

PRU_ICSS_DBG_GPREG24 is shown in Figure 6-100 and described in Table 6-247.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 24. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-246. PRU_ICSS_DBG_GPREG24 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2460h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4460h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2460h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4460h

Figure 6-100. PRU_ICSS_DBG_GPREG24 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG24																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-247. PRU_ICSS_DBG_GPREG24 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG24	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-248. Register Call Summary for PRU_ICSS_DBG_GPREG24

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_GPREG24 Register (Offset = 60h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.26 PRU_ICSS_DBG_GPREG25 Register (Offset = 64h) [reset = 0h]

PRU_ICSS_DBG_GPREG25 is shown in Figure 6-101 and described in Table 6-250.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 25. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-249. PRU_ICSS_DBG_GPREG25 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2464h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4464h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2464h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4464h

Figure 6-101. PRU_ICSS_DBG_GPREG25 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG25																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-250. PRU_ICSS_DBG_GPREG25 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG25	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-251. Register Call Summary for PRU_ICSS_DBG_GPREG25

PRU-ICSS PRU Cores
- PRU_ICSS_PRU_DEBUG Registers: [0]
- PRU_ICSS_DBG_GPREG25 Register (Offset = 64h) [reset = 0h]: [0]

6.4.5.9.27 PRU_ICSS_DBG_GPREG26 Register (Offset = 68h) [reset = 0h]

PRU_ICSS_DBG_GPREG26 is shown in Figure 6-102 and described in Table 6-253.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 26. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-252. PRU_ICSS_DBG_GPREG26 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2468h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4468h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2468h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4468h

Figure 6-102. PRU_ICSS_DBG_GPREG26 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG26																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-253. PRU_ICSS_DBG_GPREG26 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG26	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-254. Register Call Summary for PRU_ICSS_DBG_GPREG26

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_GPREG26 Register (Offset = 68h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.28 PRU_ICSS_DBG_GPREG27 Register (Offset = 6Ch) [reset = 0h]

PRU_ICSS_DBG_GPREG27 is shown in Figure 6-103 and described in Table 6-256.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 27. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-255. PRU_ICSS_DBG_GPREG27 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 246Ch
PRU_ICSS_0_PRU_DEBUG_1	20AA 446Ch
PRU_ICSS_1_PRU_DEBUG_0	20AE 246Ch
PRU_ICSS_1_PRU_DEBUG_1	20AE 446Ch

Figure 6-103. PRU_ICSS_DBG_GPREG27 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG27																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-256. PRU_ICSS_DBG_GPREG27 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG27	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-257. Register Call Summary for PRU_ICSS_DBG_GPREG27

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_GPREG27 Register (Offset = 6Ch) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.29 PRU_ICSS_DBG_GPREG28 Register (Offset = 70h) [reset = 0h]

PRU_ICSS_DBG_GPREG28 is shown in Figure 6-104 and described in Table 6-259.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 28. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-258. PRU_ICSS_DBG_GPREG28 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2470h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4470h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2470h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4470h

Figure 6-104. PRU_ICSS_DBG_GPREG28 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG28																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-259. PRU_ICSS_DBG_GPREG28 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG28	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-260. Register Call Summary for PRU_ICSS_DBG_GPREG28

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_GPREG28 Register (Offset = 70h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.30 PRU_ICSS_DBG_GPREG29 Register (Offset = 74h) [reset = 0h]

PRU_ICSS_DBG_GPREG29 is shown in Figure 6-105 and described in Table 6-262.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 29. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-261. PRU_ICSS_DBG_GPREG29 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2474h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4474h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2474h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4474h

Figure 6-105. PRU_ICSS_DBG_GPREG29 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG29																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-262. PRU_ICSS_DBG_GPREG29 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG29	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-263. Register Call Summary for PRU_ICSS_DBG_GPREG29

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_GPREG29 Register (Offset = 74h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.31 PRU_ICSS_DBG_GPREG30 Register (Offset = 78h) [reset = 0h]

PRU_ICSS_DBG_GPREG30 is shown in Figure 6-106 and described in Table 6-265.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 30. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-264. PRU_ICSS_DBG_GPREG30 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2478h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4478h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2478h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4478h

Figure 6-106. PRU_ICSS_DBG_GPREG30 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GP_REG30																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-265. PRU_ICSS_DBG_GPREG30 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG30	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

Table 6-266. Register Call Summary for PRU_ICSS_DBG_GPREG30

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_GPREG30 Register (Offset = 78h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.32 PRU_ICSS_DBG_GPREG31 Register (Offset = 7Ch) [reset = 0h]

PRU_ICSS_DBG_GPREG31 is shown in Figure 6-107 and described in Table 6-268.

Return to Summary Table.

DEBUG PRU GENERAL PURPOSE REGISTER 31. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Table 6-267. PRU_ICSS_DBG_GPREG31 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 247Ch
PRU_ICSS_0_PRU_DEBUG_1	20AA 447Ch
PRU_ICSS_1_PRU_DEBUG_0	20AE 247Ch
PRU_ICSS_1_PRU_DEBUG_1	20AE 447Ch

Figure 6-107. PRU_ICSS_DBG_GPREG31 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG31																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-268. PRU_ICSS_DBG_GPREG31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG31	R/W	0h	

Table 6-269. Register Call Summary for PRU_ICSS_DBG_GPREG31

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_GPREG31 Register (Offset = 7Ch) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.33 PRU_ICSS_DBG_CT_REG0 Register (Offset = 80h) [reset = 00020000h]

PRU_ICSS_DBG_CT_REG0 is shown in Figure 6-108 and described in Table 6-271.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 0. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-270. PRU_ICSS_DBG_CT_REG0 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2480h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4480h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2480h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4480h

Figure 6-108. PRU_ICSS_DBG_CT_REG0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG0																															
R-00020000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-271. PRU_ICSS_DBG_CT_REG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG0	R	00020000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-272. Register Call Summary for PRU_ICSS_DBG_CT_REG0

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_CT_REG0 Register (Offset = 80h) [reset = 00020000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.34 PRU_ICSS_DBG_CT_REG1 Register (Offset = 84h) [reset = 48040000h]

PRU_ICSS_DBG_CT_REG1 is shown in Figure 6-109 and described in Table 6-274.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 1. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-273. PRU_ICSS_DBG_CT_REG1 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2484h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4484h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2484h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4484h

Figure 6-109. PRU_ICSS_DBG_CT_REG1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG1																															
R-48040000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-274. PRU_ICSS_DBG_CT_REG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG1	R	48040000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-275. Register Call Summary for PRU_ICSS_DBG_CT_REG1

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_CT_REG1 Register (Offset = 84h) [reset = 48040000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.35 PRU_ICSS_DBG_CT_REG2 Register (Offset = 88h) [reset = 4802A000h]

PRU_ICSS_DBG_CT_REG2 is shown in Figure 6-110 and described in Table 6-277.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 2. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-276. PRU_ICSS_DBG_CT_REG2 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2488h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4488h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2488h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4488h

Figure 6-110. PRU_ICSS_DBG_CT_REG2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG2																															
R-4802A000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-277. PRU_ICSS_DBG_CT_REG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG2	R	4802A000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-278. Register Call Summary for PRU_ICSS_DBG_CT_REG2

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_CT_REG2 Register (Offset = 88h) [reset = 4802A000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.36 PRU_ICSS_DBG_CT_REG3 Register (Offset = 8Ch) [reset = 00030000h]

PRU_ICSS_DBG_CT_REG3 is shown in Figure 6-111 and described in Table 6-280.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 3. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-279. PRU_ICSS_DBG_CT_REG3 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 248Ch
PRU_ICSS_0_PRU_DEBUG_1	20AA 448Ch
PRU_ICSS_1_PRU_DEBUG_0	20AE 248Ch
PRU_ICSS_1_PRU_DEBUG_1	20AE 448Ch

Figure 6-111. PRU_ICSS_DBG_CT_REG3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG3																															
R-00030000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-280. PRU_ICSS_DBG_CT_REG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG3	R	00030000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-281. Register Call Summary for PRU_ICSS_DBG_CT_REG3

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_CT_REG3 Register (Offset = 8Ch) [reset = 00030000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.37 PRU_ICSS_DBG_CT_REG4 Register (Offset = 90h) [reset = 00026000h]

PRU_ICSS_DBG_CT_REG4 is shown in Figure 6-112 and described in Table 6-283.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 4. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-282. PRU_ICSS_DBG_CT_REG4 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2490h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4490h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2490h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4490h

Figure 6-112. PRU_ICSS_DBG_CT_REG4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG4																															
R-00026000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-283. PRU_ICSS_DBG_CT_REG4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG4	R	00026000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-284. Register Call Summary for PRU_ICSS_DBG_CT_REG4

PRU-ICSS PRU Cores
- PRU_ICSS_PRU_DEBUG Registers: [0]
- PRU_ICSS_DBG_CT_REG4 Register (Offset = 90h) [reset = 00026000h]: [0]

6.4.5.9.38 PRU_ICSS_DBG_CT_REG5 Register (Offset = 94h) [reset = 48060000h]

PRU_ICSS_DBG_CT_REG5 is shown in Figure 6-113 and described in Table 6-286.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 5. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-285. PRU_ICSS_DBG_CT_REG5 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2494h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4494h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2494h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4494h

Figure 6-113. PRU_ICSS_DBG_CT_REG5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG5																															
R-48060000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-286. PRU_ICSS_DBG_CT_REG5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG5	R	48060000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-287. Register Call Summary for PRU_ICSS_DBG_CT_REG5

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_CT_REG5 Register (Offset = 94h) [reset = 48060000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.39 PRU_ICSS_DBG_CT_REG6 Register (Offset = 98h) [reset = 48030000h]

PRU_ICSS_DBG_CT_REG6 is shown in Figure 6-114 and described in Table 6-289.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 6. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-288. PRU_ICSS_DBG_CT_REG6 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 2498h
PRU_ICSS_0_PRU_DEBUG_1	20AA 4498h
PRU_ICSS_1_PRU_DEBUG_0	20AE 2498h
PRU_ICSS_1_PRU_DEBUG_1	20AE 4498h

Figure 6-114. PRU_ICSS_DBG_CT_REG6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG6																															
R-48030000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-289. PRU_ICSS_DBG_CT_REG6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG6	R	48030000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-290. Register Call Summary for PRU_ICSS_DBG_CT_REG6

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_CT_REG6 Register (Offset = 98h) [reset = 48030000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.40 PRU_ICSS_DBG_CT_REG7 Register (Offset = 9Ch) [reset = 00028000h]

PRU_ICSS_DBG_CT_REG7 is shown in Figure 6-115 and described in Table 6-292.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 7. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-291. PRU_ICSS_DBG_CT_REG7 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 249Ch
PRU_ICSS_0_PRU_DEBUG_1	20AA 449Ch
PRU_ICSS_1_PRU_DEBUG_0	20AE 249Ch
PRU_ICSS_1_PRU_DEBUG_1	20AE 449Ch

Figure 6-115. PRU_ICSS_DBG_CT_REG7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG7																															
R-00028000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-292. PRU_ICSS_DBG_CT_REG7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG7	R	00028000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-293. Register Call Summary for PRU_ICSS_DBG_CT_REG7

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_CT_REG7 Register (Offset = 9Ch) [reset = 00028000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.41 PRU_ICSS_DBG_CT_REG8 Register (Offset = A0h) [reset = 46000000h]

PRU_ICSS_DBG_CT_REG8 is shown in Figure 6-116 and described in Table 6-295.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 8. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-294. PRU_ICSS_DBG_CT_REG8 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24A0h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44A0h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24A0h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44A0h

Figure 6-116. PRU_ICSS_DBG_CT_REG8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG8																															
R-46000000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-295. PRU_ICSS_DBG_CT_REG8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG8	R	46000000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-296. Register Call Summary for PRU_ICSS_DBG_CT_REG8

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_CT_REG8 Register (Offset = A0h) [reset = 46000000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.42 PRU_ICSS_DBG_CT_REG9 Register (Offset = A4h) [reset = 4A100000h]

PRU_ICSS_DBG_CT_REG9 is shown in Figure 6-117 and described in Table 6-298.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 9. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-297. PRU_ICSS_DBG_CT_REG9 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24A4h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44A4h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24A4h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44A4h

Figure 6-117. PRU_ICSS_DBG_CT_REG9 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG9																															
R-4A100000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-298. PRU_ICSS_DBG_CT_REG9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG9	R	4A100000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-299. Register Call Summary for PRU_ICSS_DBG_CT_REG9

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_CT_REG9 Register (Offset = A4h) [reset = 4A100000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.43 PRU_ICSS_DBG_CT_REG10 Register (Offset = A8h) [reset = 48318000h]

PRU_ICSS_DBG_CT_REG10 is shown in Figure 6-118 and described in Table 6-301.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 10. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-300. PRU_ICSS_DBG_CT_REG10 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24A8h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44A8h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24A8h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44A8h

Figure 6-118. PRU_ICSS_DBG_CT_REG10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG10																															
R-48318000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-301. PRU_ICSS_DBG_CT_REG10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG10	R	48318000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-302. Register Call Summary for PRU_ICSS_DBG_CT_REG10

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_CT_REG10 Register (Offset = A8h) [reset = 48318000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.44 PRU_ICSS_DBG_CT_REG11 Register (Offset = ACh) [reset = 48022000h]

PRU_ICSS_DBG_CT_REG11 is shown in Figure 6-119 and described in Table 6-304.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 11. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-303. PRU_ICSS_DBG_CT_REG11 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24ACh
PRU_ICSS_0_PRU_DEBUG_1	20AA 44ACh
PRU_ICSS_1_PRU_DEBUG_0	20AE 24ACh
PRU_ICSS_1_PRU_DEBUG_1	20AE 44ACh

Figure 6-119. PRU_ICSS_DBG_CT_REG11 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG11																															
R-48022000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-304. PRU_ICSS_DBG_CT_REG11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG11	R	48022000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-305. Register Call Summary for PRU_ICSS_DBG_CT_REG11

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_CT_REG11 Register (Offset = ACh) [reset = 48022000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.45 PRU_ICSS_DBG_CT_REG12 Register (Offset = B0h) [reset = 48024000h]

PRU_ICSS_DBG_CT_REG12 is shown in Figure 6-120 and described in Table 6-307.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 12. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-306. PRU_ICSS_DBG_CT_REG12 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24B0h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44B0h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24B0h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44B0h

Figure 6-120. PRU_ICSS_DBG_CT_REG12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG12																															
R-48024000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-307. PRU_ICSS_DBG_CT_REG12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG12	R	48024000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-308. Register Call Summary for PRU_ICSS_DBG_CT_REG12

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_CT_REG12 Register (Offset = B0h) [reset = 48024000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.46 PRU_ICSS_DBG_CT_REG13 Register (Offset = B4h) [reset = 48310000h]

PRU_ICSS_DBG_CT_REG13 is shown in Figure 6-121 and described in Table 6-310.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 13. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-309. PRU_ICSS_DBG_CT_REG13 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24B4h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44B4h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24B4h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44B4h

Figure 6-121. PRU_ICSS_DBG_CT_REG13 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG13																															
R-48310000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-310. PRU_ICSS_DBG_CT_REG13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG13	R	48310000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-311. Register Call Summary for PRU_ICSS_DBG_CT_REG13

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_CT_REG13 Register (Offset = B4h) [reset = 48310000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.47 PRU_ICSS_DBG_CT_REG14 Register (Offset = B8h) [reset = 481CC000h]

PRU_ICSS_DBG_CT_REG14 is shown in Figure 6-122 and described in Table 6-313.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 14. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-312. PRU_ICSS_DBG_CT_REG14 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24B8h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44B8h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24B8h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44B8h

Figure 6-122. PRU_ICSS_DBG_CT_REG14 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG14																															
R-481CC000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-313. PRU_ICSS_DBG_CT_REG14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG14	R	481CC000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-314. Register Call Summary for PRU_ICSS_DBG_CT_REG14

PRU-ICSS PRU Cores
- PRU_ICSS_PRU_DEBUG Registers: [0]
- PRU_ICSS_DBG_CT_REG14 Register (Offset = B8h) [reset = 481CC000h]: [0]

6.4.5.9.48 PRU_ICSS_DBG_CT_REG15 Register (Offset = BCh) [reset = 481D0000h]

PRU_ICSS_DBG_CT_REG15 is shown in Figure 6-123 and described in Table 6-316.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 15. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-315. PRU_ICSS_DBG_CT_REG15 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24BCh
PRU_ICSS_0_PRU_DEBUG_1	20AA 44BCh
PRU_ICSS_1_PRU_DEBUG_0	20AE 24BCh
PRU_ICSS_1_PRU_DEBUG_1	20AE 44BCh

Figure 6-123. PRU_ICSS_DBG_CT_REG15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG15																															
R-481D0000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-316. PRU_ICSS_DBG_CT_REG15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG15	R	481D0000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-317. Register Call Summary for PRU_ICSS_DBG_CT_REG15

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_CT_REG15 Register (Offset = BCh) [reset = 481D0000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.49 PRU_ICSS_DBG_CT_REG16 Register (Offset = C0h) [reset = 481A0000h]

PRU_ICSS_DBG_CT_REG16 is shown in Figure 6-124 and described in Table 6-319.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 16. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-318. PRU_ICSS_DBG_CT_REG16 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24C0h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44C0h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24C0h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44C0h

Figure 6-124. PRU_ICSS_DBG_CT_REG16 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG16																															
R-481A0000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-319. PRU_ICSS_DBG_CT_REG16 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG16	R	481A0000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-320. Register Call Summary for PRU_ICSS_DBG_CT_REG16

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_CT_REG16 Register (Offset = C0h) [reset = 481A0000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.50 PRU_ICSS_DBG_CT_REG17 Register (Offset = C4h) [reset = 4819C000h]

PRU_ICSS_DBG_CT_REG17 is shown in Figure 6-125 and described in Table 6-322.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 17. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-321. PRU_ICSS_DBG_CT_REG17 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24C4h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44C4h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24C4h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44C4h

Figure 6-125. PRU_ICSS_DBG_CT_REG17 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG17																															
R-4819C000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-322. PRU_ICSS_DBG_CT_REG17 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG17	R	4819C000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-323. Register Call Summary for PRU_ICSS_DBG_CT_REG17

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_CT_REG17 Register (Offset = C4h) [reset = 4819C000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.51 PRU_ICSS_DBG_CT_REG18 Register (Offset = C8h) [reset = 48300000h]

PRU_ICSS_DBG_CT_REG18 is shown in Figure 6-126 and described in Table 6-325.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 18. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-324. PRU_ICSS_DBG_CT_REG18 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24C8h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44C8h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24C8h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44C8h

Figure 6-126. PRU_ICSS_DBG_CT_REG18 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG18																															
R-48300000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-325. PRU_ICSS_DBG_CT_REG18 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG18	R	48300000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-326. Register Call Summary for PRU_ICSS_DBG_CT_REG18

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_CT_REG18 Register (Offset = C8h) [reset = 48300000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.52 PRU_ICSS_DBG_CT_REG19 Register (Offset = CCh) [reset = 48302000h]

PRU_ICSS_DBG_CT_REG19 is shown in Figure 6-127 and described in Table 6-328.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 19. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-327. PRU_ICSS_DBG_CT_REG19 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24CCh
PRU_ICSS_0_PRU_DEBUG_1	20AA 44CCh
PRU_ICSS_1_PRU_DEBUG_0	20AE 24CCh
PRU_ICSS_1_PRU_DEBUG_1	20AE 44CCh

Figure 6-127. PRU_ICSS_DBG_CT_REG19 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG19																															
R-48302000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-328. PRU_ICSS_DBG_CT_REG19 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG19	R	48302000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-329. Register Call Summary for PRU_ICSS_DBG_CT_REG19

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_CT_REG19 Register (Offset = CCh) [reset = 48302000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.53 PRU_ICSS_DBG_CT_REG20 Register (Offset = D0h) [reset = 48304000h]

PRU_ICSS_DBG_CT_REG20 is shown in Figure 6-128 and described in Table 6-331.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 20. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-330. PRU_ICSS_DBG_CT_REG20 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24D0h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44D0h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24D0h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44D0h

Figure 6-128. PRU_ICSS_DBG_CT_REG20 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG20																															
R-48304000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-331. PRU_ICSS_DBG_CT_REG20 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG20	R	48304000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-332. Register Call Summary for PRU_ICSS_DBG_CT_REG20

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_CT_REG20 Register (Offset = D0h) [reset = 48304000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.54 PRU_ICSS_DBG_CT_REG21 Register (Offset = D4h) [reset = 00032400h]

PRU_ICSS_DBG_CT_REG21 is shown in Figure 6-129 and described in Table 6-334.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 21. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-333. PRU_ICSS_DBG_CT_REG21 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24D4h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44D4h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24D4h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44D4h

Figure 6-129. PRU_ICSS_DBG_CT_REG21 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG21																															
R-00032400h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-334. PRU_ICSS_DBG_CT_REG21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG21	R	00032400h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-335. Register Call Summary for PRU_ICSS_DBG_CT_REG21

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_CT_REG21 Register (Offset = D4h) [reset = 00032400h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.55 PRU_ICSS_DBG_CT_REG22 Register (Offset = D8h) [reset = 480C8000h]

PRU_ICSS_DBG_CT_REG22 is shown in Figure 6-130 and described in Table 6-337.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 22. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-336. PRU_ICSS_DBG_CT_REG22 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24D8h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44D8h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24D8h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44D8h

Figure 6-130. PRU_ICSS_DBG_CT_REG22 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG22																															
R-480C8000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-337. PRU_ICSS_DBG_CT_REG22 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG22	R	480C8000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-338. Register Call Summary for PRU_ICSS_DBG_CT_REG22

PRU-ICSS PRU Cores
- PRU_ICSS_DBG_CT_REG22 Register (Offset = D8h) [reset = 480C8000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.56 PRU_ICSS_DBG_CT_REG23 Register (Offset = DCh) [reset = 480CA000h]

PRU_ICSS_DBG_CT_REG23 is shown in Figure 6-131 and described in Table 6-340.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 23. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-339. PRU_ICSS_DBG_CT_REG23 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24DCh
PRU_ICSS_0_PRU_DEBUG_1	20AA 44DCh
PRU_ICSS_1_PRU_DEBUG_0	20AE 24DCh
PRU_ICSS_1_PRU_DEBUG_1	20AE 44DCh

Figure 6-131. PRU_ICSS_DBG_CT_REG23 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG23																															
R-480CA000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-340. PRU_ICSS_DBG_CT_REG23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG23	R	480CA000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

Table 6-341. Register Call Summary for PRU_ICSS_DBG_CT_REG23

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_CT_REG23 Register (Offset = DCh) [reset = 480CA000h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.57 PRU_ICSS_DBG_CT_REG24 Register (Offset = E0h) [reset = 0h]

PRU_ICSS_DBG_CT_REG24 is shown in Figure 6-132 and described in Table 6-343.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 24. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-342. PRU_ICSS_DBG_CT_REG24 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24E0h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44E0h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24E0h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44E0h

Figure 6-132. PRU_ICSS_DBG_CT_REG24 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG24																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-343. PRU_ICSS_DBG_CT_REG24 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG24	R	0h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table. This entry is partially programmable through the C24_BLK_INDEX in the PRU Control register. The reset value for this Constant Table Entry is 0x00000n00, n=C24_BLK_INDEX[3:0].

Table 6-344. Register Call Summary for PRU_ICSS_DBG_CT_REG24

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_CT_REG24 Register (Offset = E0h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.58 PRU_ICSS_DBG_CT_REG25 Register (Offset = E4h) [reset = 0h]

PRU_ICSS_DBG_CT_REG25 is shown in Figure 6-133 and described in Table 6-346.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 25. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-345. PRU_ICSS_DBG_CT_REG25 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24E4h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44E4h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24E4h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44E4h

Figure 6-133. PRU_ICSS_DBG_CT_REG25 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG25																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-346. PRU_ICSS_DBG_CT_REG25 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG25	R	0h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table. This entry is partially programmable through the C25_BLK_INDEX in the PRU Control register. The reset value for this Constant Table Entry is 0x00002n00, n=C25_BLK_INDEX[3:0].

Table 6-347. Register Call Summary for PRU_ICSS_DBG_CT_REG25

PRU-ICSS PRU Cores

- PRU_ICSS_PRU_DEBUG Registers: [0]
- PRU_ICSS_DBG_CT_REG25 Register (Offset = E4h) [reset = 0h]: [0]

6.4.5.9.59 PRU_ICSS_DBG_CT_REG26 Register (Offset = E8h) [reset = 0h]

PRU_ICSS_DBG_CT_REG26 is shown in Figure 6-134 and described in Table 6-349.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 26. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-348. PRU_ICSS_DBG_CT_REG26 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24E8h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44E8h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24E8h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44E8h

Figure 6-134. PRU_ICSS_DBG_CT_REG26 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG26																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-349. PRU_ICSS_DBG_CT_REG26 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG26	R	0h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table. This entry is partially programmable through the C26_BLK_INDEX in the PRU Control register. The reset value for this Constant Table Entry is 0x0002En00, n=C26_BLK_INDEX[3:0].

Table 6-350. Register Call Summary for PRU_ICSS_DBG_CT_REG26

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_CT_REG26 Register (Offset = E8h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.60 PRU_ICSS_DBG_CT_REG27 Register (Offset = ECh) [reset = 0h]

PRU_ICSS_DBG_CT_REG27 is shown in Figure 6-135 and described in Table 6-352.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 27. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-351. PRU_ICSS_DBG_CT_REG27 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24ECh
PRU_ICSS_0_PRU_DEBUG_1	20AA 44ECh
PRU_ICSS_1_PRU_DEBUG_0	20AE 24ECh
PRU_ICSS_1_PRU_DEBUG_1	20AE 44ECh

Figure 6-135. PRU_ICSS_DBG_CT_REG27 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG27																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-352. PRU_ICSS_DBG_CT_REG27 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG27	R	0h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table. This entry is partially programmable through the C27_BLK_INDEX in the PRU Control register. The reset value for this Constant Table Entry is 0x00032n00, n=C27_BLK_INDEX[3:0].

Table 6-353. Register Call Summary for PRU_ICSS_DBG_CT_REG27

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_CT_REG27 Register (Offset = ECh) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.61 PRU_ICSS_DBG_CT_REG28 Register (Offset = F0h) [reset = 0h]

PRU_ICSS_DBG_CT_REG28 is shown in Figure 6-136 and described in Table 6-355.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 28. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-354. PRU_ICSS_DBG_CT_REG28 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24F0h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44F0h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24F0h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44F0h

Figure 6-136. PRU_ICSS_DBG_CT_REG28 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG28																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-355. PRU_ICSS_DBG_CT_REG28 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG28	R	0h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table. This entry is partially programmable through the C28_POINTER in the PRU Control register. The reset value for this Constant Table Entry is 0x00nnnn00, nnnn=C28_POINTER[15:0].

Table 6-356. Register Call Summary for PRU_ICSS_DBG_CT_REG28

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_CT_REG28 Register (Offset = F0h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.62 PRU_ICSS_DBG_CT_REG29 Register (Offset = F4h) [reset = 0h]

PRU_ICSS_DBG_CT_REG29 is shown in Figure 6-137 and described in Table 6-358.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 29. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-357. PRU_ICSS_DBG_CT_REG29 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24F4h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44F4h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24F4h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44F4h

Figure 6-137. PRU_ICSS_DBG_CT_REG29 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG29																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-358. PRU_ICSS_DBG_CT_REG29 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG29	R	0h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table. This entry is partially programmable through the C29_POINTER in the PRU Control register. The reset value for this Constant Table Entry is 0x49nnnn00, nnnn=C29_POINTER[15:0].

Table 6-359. Register Call Summary for PRU_ICSS_DBG_CT_REG29

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_CT_REG29 Register (Offset = F4h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.63 PRU_ICSS_DBG_CT_REG30 Register (Offset = F8h) [reset = 0h]

PRU_ICSS_DBG_CT_REG30 is shown in Figure 6-138 and described in Table 6-361.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 30. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-360. PRU_ICSS_DBG_CT_REG30 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24F8h
PRU_ICSS_0_PRU_DEBUG_1	20AA 44F8h
PRU_ICSS_1_PRU_DEBUG_0	20AE 24F8h
PRU_ICSS_1_PRU_DEBUG_1	20AE 44F8h

Figure 6-138. PRU_ICSS_DBG_CT_REG30 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG30																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-361. PRU_ICSS_DBG_CT_REG30 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG30	R	0h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table. This entry is partially programmable through the C30_POINTER in the PRU Control register. The reset value for this Constant Table Entry is 0x40nnnn00, nnnn=C30_POINTER[15:0].

Table 6-362. Register Call Summary for PRU_ICSS_DBG_CT_REG30

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_CT_REG30 Register (Offset = F8h) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.5.9.64 PRU_ICSS_DBG_CT_REG31 Register (Offset = FCh) [reset = 0h]

PRU_ICSS_DBG_CT_REG31 is shown in Figure 6-139 and described in Table 6-364.

Return to Summary Table.

DEBUG PRU CONSTANTS TABLE ENTRY 31. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Table 6-363. PRU_ICSS_DBG_CT_REG31 Instances

Instance	Physical Address
PRU_ICSS_0_PRU_DEBUG_0	20AA 24FCh
PRU_ICSS_0_PRU_DEBUG_1	20AA 44FCh
PRU_ICSS_1_PRU_DEBUG_0	20AE 24FCh
PRU_ICSS_1_PRU_DEBUG_1	20AE 44FCh

Figure 6-139. PRU_ICSS_DBG_CT_REG31 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CT_REG31																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-364. PRU_ICSS_DBG_CT_REG31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG31	R	0h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table. This entry is partially programmable through the C31_POINTER in the PRU Control register. The reset value for this Constant Table Entry is 0x80nnnn00, nnnn=C31_POINTER[15:0].

Table 6-365. Register Call Summary for PRU_ICSS_DBG_CT_REG31

PRU-ICSS PRU Cores

- PRU_ICSS_DBG_CT_REG31 Register (Offset = FCh) [reset = 0h]: [0]
- PRU_ICSS_PRU_DEBUG Registers: [0]

6.4.6 PRU-ICSS Local Interrupt Controller

This section describes functionality of the PRU-ICSS integrated Interrupt Controller - ICSS_INTC.

6.4.6.1 PRU-ICSS Interrupt Controller Overview

The PRU-ICSS interrupt controller (ICSS_INTC) maps interrupts coming from different parts of the device (mapped to PRU-ICSS_0/PRU-ICSS_1) to a reduced set of PRU-ICSS interrupt channels.

The ICSS_INTC has the following features:

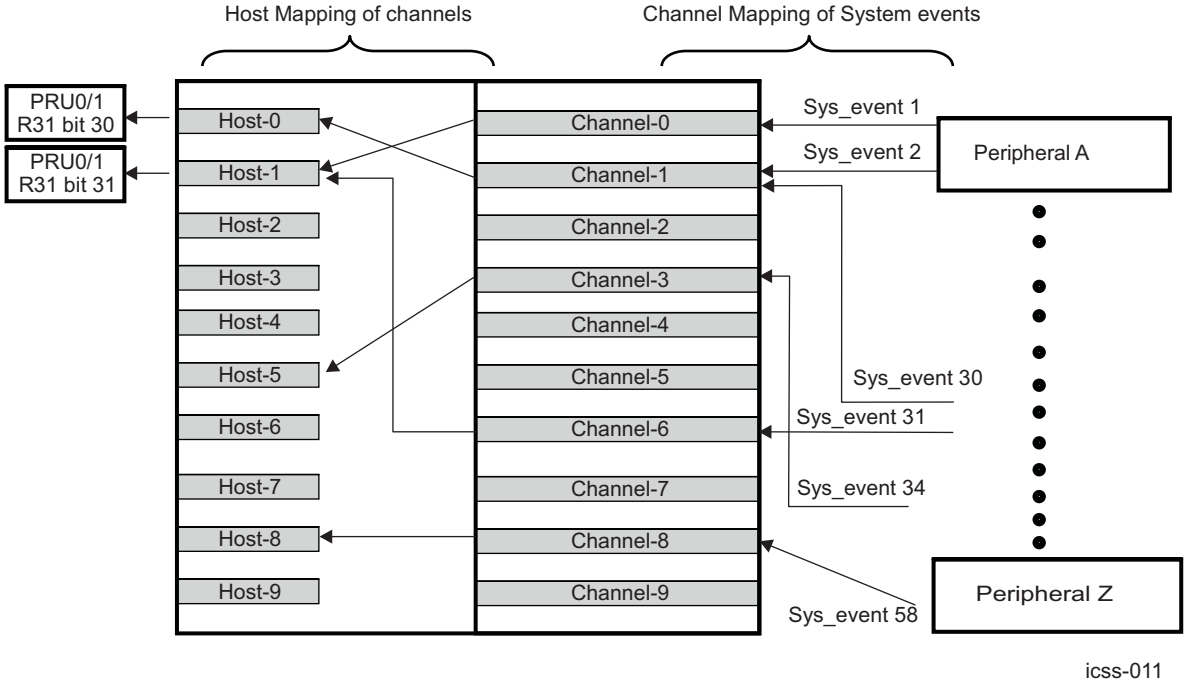
- Capturing up to 64 System Events (inputs)
- Supports up to 10 output interrupt channels.
- Generation of 10 Host Interrupts
  - 2 Host Interrupts for the PRUs.
  - 7 Host Interrupts exported from the PRU-ICSS for signaling the Arm interrupt controllers (pulse and level provided).
  - 1 Host Interrupt for the other PRU-ICSS.
- Each system event can be enabled and disabled.
- Each host event can be enabled and disabled.
- Hardware prioritization of events.

6.4.6.2 PRU-ICSS Interrupt Controller Functional Description

The PRU-ICSS incorporates an interrupt controller - ICSS_INTC that supports up to 64 system interrupts from different peripherals (including 32 interrupts from PRU-ICSS located interrupt sources). The ICSS_INTC maps these system events to 10 channels inside the ICSS_INTC (see Figure 6-140). Interrupts from these 10 channels are further mapped to 10 Host Interrupts.

- Any of the 64 system interrupts can be mapped to any of the 10 channels.
- Multiple interrupts can be mapped to a single channel.
- An interrupt should not be mapped to more than one channel.
- Any of the 10 channels can be mapped to any of the 10 host interrupts. It is recommended to map channel “x” to host interrupt “x”, where x is from 0 to 9
- A channel should not be mapped to more than one host interrupt
- For channels mapping to the same host interrupt, lower number channels have higher priority.
- For interrupts on same channel, priority is determined by the hardware interrupt number. The lower the interrupt number, the higher the priority.
- Host Interrupt 0 is connected to bit 30 in register 31 (R31) of PRU0 and PRU1.
- Host Interrupt 1 is connected to bit 31 in register 31 (R31) for PRU0 and PRU1.
- Host Interrupts 2 through 9 exported from PRU-ICSS and mapped to interrupt controllers in the device.
- Host Interrupt 7 is exported from PRU-ICSS for signaling the other PRU-ICSS

Figure 6-140. PRU-ICSS Interrupt Controller Block Diagram



6.4.6.2.1 PRU-ICSS Interrupt Controller System Events

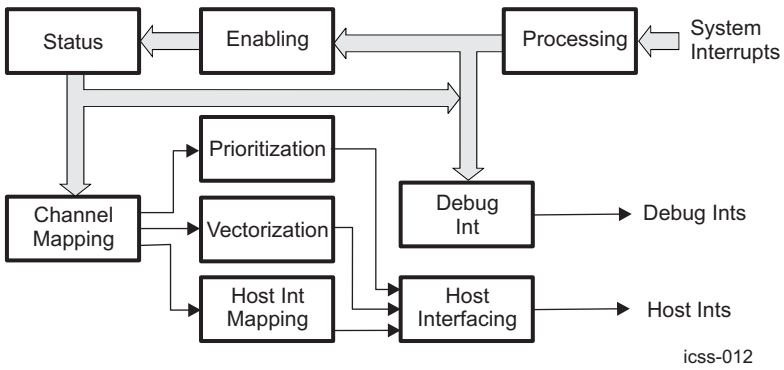
The PRU-ICSS system events - interrupt inputs. The device includes a internal mux that selects the "Standard" (default) or "MII_RT" mode system events. The mux control signal is controlled by PRUSS_MII_RT[0] MII_RT_EVENT_EN, which can be modified by software in PRU-ICSS CFG register space.

6.4.6.2.2 PRU-ICSS Interrupt Controller System Events Flow

The ICSS_INTC module controls the system event mapping to the host interrupt interface. System events are generated by the device peripherals or PRUs. The ICSS_INTC receives the system interrupts and maps them to internal channels. The channels are used to group interrupts together and to prioritize them. These channels are then mapped onto the host interrupts. Interrupts from the system side are active high in polarity. They are also pulse type of interrupts.

The ICSS_INTC encompasses many functions to process the system interrupts and prepare them for the host interface. These functions are: processing, enabling, status, channel mapping, host interrupt mapping, prioritization, and host interfacing. Figure 6-141 illustrates the flow of system interrupts through the functions to the host. The following subsections describe each part of the flow.

Figure 6-141. Flow of System Interrupts to Host



6.4.6.2.2.1 PRU-ICSS Interrupt Processing

This block does following tasks:

- Synchronization of slower and asynchronous interrupts
- Conversion of polarity to active high
- Conversion of interrupt type to pulse interrupts

After the processing block, all interrupts will be active high pulses.

6.4.6.2.2.1.1 PRU-ICSS Interrupt Enabling

The next stage of ICSS_INTC is to enable system interrupts based on programmed settings. The following sequence is to be followed to enable interrupts:

- Enable required system interrupts: System interrupts that are required to get propagated to host are to be enabled individually by writing to INDEX field in the system interrupt enable indexed set register (PRUSS_INTC_EISR). The interrupt to enable is the index value written. This sets the Enable Register bit of the given index.
- Enable required host interrupts: By writing 1 to the appropriate bit of the INDEX field in the host interrupt enable indexed set register (PRUSS_INTC_HIEISR), enable the required host interrupts. The host interrupt to enable is the index value written. This enables the host interrupt output or triggers the output again if that host interrupt is already enabled.
- Enable all host interrupts: By setting the ENABLE bit in the global enable register (PRUSS_INTC_GER) to 1, all host interrupts will be enabled. Individual host interrupts are still enabled or disabled from their individual enables and are not overridden by the global enable.

6.4.6.2.2.2 PRU-ICSS Interrupt Status Checking

The next stage is to capture which system interrupts are pending. There are two kinds of pending status: raw status and enabled status. Raw status is the pending status of the system interrupt without regards to the enable bit for the system interrupt. Enabled status is the pending status of the system interrupts with the enable bits active. When the enable bit is inactive, the enabled status will always be inactive. The enabled status of system interrupts is captured in system interrupt status enabled/clear registers (PRUSS_INTC_SECR1 and PRUSS_INTC_SECR0).

Status of system interrupt 'N' is indicated by the N-th bit of PRUSS_INTC_SECR1 and PRUSS_INTC_SECR0. Since there are 64 system interrupts, two 32-bit registers are used to capture the enabled status of interrupts. The pending status reflects whether the system interrupt occurred since the last time the status register bit was cleared. Each bit in the status register can be individually cleared.

6.4.6.2.2.3 PRU-ICSS Interrupt Channel Mapping

The ICSS_INTC has 10 internal channels to which enabled system interrupts can be mapped. Channel 0 has highest priority and channel 9 has the lowest priority. Channels are used to group the system interrupts into a smaller number of priorities that can be given to a host interface with a very small number of interrupt inputs.

When multiple system interrupts are mapped to the same channel their interrupts are ORed together so that when either is active the output is active. The channel map registers (PRUSS_INTC_CMRi, where i=0 to 15) define the channel for each system interrupt. There is one register per 4 system interrupts; therefore, there are 16 channel map registers for a system of 64 interrupts. The channel for each system interrupt can be set using these registers.

6.4.6.2.2.3.1 PRU-ICSS Host Interrupt Mapping

The hosts can be the local PRU processors (PRU0 and PRU1) as well as device processors located outside PRU-ICSS such as MPU Cortex-A15, DSP, etc. The 10 channels from the ICSS_INTC can be mapped to any of the 10 Host interrupts. The Host map registers (PRUSS_INTC_HMR0 - PRUSS_INTC_HMR2) define the channel for each system interrupt. There is one register per 4 channels; therefore, there are 3 host map registers for 10 channels. When multiple channels are mapped to the same host interrupt, then prioritization is done to select which interrupt is in the highest-priority channel and which should be sent first to the host.

6.4.6.2.2.3.2 PRU-ICSS Interrupt Prioritization

The next stage of the ICSS_INTC is prioritization. Since multiple interrupts can feed into a single channel and multiple channels can feed into a single host interrupt, it is to read the status of all system interrupts to determine the highest priority interrupt that is pending. The ICSS_INTC provides hardware to perform this prioritization with a given scheme so that software does not have to do this. There are two levels of prioritizations:

- The first level of prioritization is between the active channels for a host interrupt. Channel 0 has the highest priority and channel 9 has the lowest. So the first level of prioritization picks the lowest numbered active channel.
- The second level of prioritization is between the active system interrupts for the prioritized channel. The system interrupt in position 0 has the highest priority and system interrupt 63 has the lowest priority. So the second level of prioritization picks the lowest position active system interrupt.

This is the final prioritized system interrupt for the host interrupt and is stored in the global prioritized index register (PRUSS_INTC_GPIR). The highest priority pending interrupt with respect to each host interrupts can be obtained using the host interrupt prioritized index registers (PRUSS_INTC_HIPIRj where j=0 to 9).

6.4.6.2.2.4 PRU-ICSS Interrupt Nesting

The ICSS_INTC can also perform a nesting function in its prioritization. Nesting is a method of disabling certain interrupts (usually lower-priority interrupts) when an interrupt is taken so that only those desired interrupts can trigger to the host while it is servicing the current interrupt. The typical usage is to nest on the current interrupt and disable all interrupts of the same or lower priority (or channel). Then the host will only be interrupted from a higher priority interrupt.

The nesting is done in one of three methods:

1. Nesting for all host interrupts, based on channel priority: When an interrupt is taken, the nesting level is set to its channel priority. From then, that channel priority and all lower priority channels will be disabled from generating host interrupts and only higher priority channels are allowed. When the interrupt is completely serviced, the nesting level is returned to its original value. When there is no interrupt being serviced, there are no channels disabled due to nesting. The global nesting level register (PRUSS_INTC_GNLR) allows the checking and setting of the global nesting level across all host interrupts. The nesting level is the channel (and all of lower priority channels) that are nested out because of a current interrupt.
2. Nesting for individual host interrupts, based on channel priority: Always nest based on channel priority for each host interrupt individually. When an interrupt is taken on a host interrupt, then, the nesting level is set to its channel priority for just that host interrupt, and other host interrupts do not have their nesting affected. Then for that host interrupt, equal or lower priority channels will not interrupt the host but may on other host interrupts if programmed. When the interrupt is completely serviced the nesting level for the host interrupt is returned to its original value. The host interrupt nesting level registers (PRUSS_INTC_HINLRj where j=0 to 9) display and control the nesting level for each host interrupt. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.
3. Software manually performs the nesting of interrupts. When an interrupt is taken, the software will disable all the host interrupts, manually update the enables for any or all the system interrupts, and then re-enables all the host interrupts. This now allows only the system interrupts that are still enabled to trigger to the host. When the interrupt is completely serviced the software must reverse the changes to re-enable the nested out system interrupts. This method requires the most software interaction but gives the most flexibility if simple channel based nesting mechanisms are not adequate.

6.4.6.2.2.5 PRU-ICSS Interrupt Status Clearing

After servicing the interrupt (after execution of the ISR), interrupt status is to be cleared. If a system interrupt status is not cleared, then another host interrupt may not be triggered or another host interrupt may be triggered incorrectly. It is also essential to clear all system interrupts before the PRU is halted as the PRU does not power down unless all the interrupt status are cleared. For clearing the status of an interrupt, whose interrupt number is N, write a 1 to the Nth bit position in the system interrupt status enabled/clear registers (PRUSS_INTC_SECR0 and PRUSS_INTC_SECR1). System interrupt N can also be cleared by writing the value N into the system interrupt status indexed clear register (PRUSS_INTC_SICR).

6.4.6.2.3 PRU-ICSS Interrupt Disabling

At any time, if any interrupt is not to be propagated to the host, then that interrupt should be disabled. For disabling an interrupt whose interrupt number is N, write a 1 to the Nth bit in the system interrupt enable clear registers (PRUSS_INTC_ECR0 and PRUSS_INTC_ECR1). System interrupt N can also be disabled by writing the value N in the system interrupt enable indexed clear register (PRUSS_INTC_EICR).

6.4.6.3 PRU-ICSS Interrupt Controller Basic Programming Model

Follow these steps to configure the interrupt controller.

1. Set polarity and type of system event through the System Interrupt Polarity Registers (PRUSS_INTC_SIPR1 and PRUSS_INTC_SIPR0) and the System Interrupt Type Registers (PRUSS_INTC_SITR1 and PRUSS_INTC_SITR0). Polarity of all system interrupts is always high. Type of all system interrupts is always pulse.
2. Map system event to ICSS_INTC channel through PRUSS_INTC_CMRi (i=0 to 15) channel mapping registers.
3. Map channel to host interrupt through ICSS_INTC_HMR0/1/2 registers. Recommended channel “x” to be mapped to host interrupt “x”.
4. Clear system interrupt by writing 1 to ICSS_INTC_SECR0/1 registers.
5. Enable host interrupt by writing index value to PRUSS_INTC_HIEISR register.
6. Enable interrupt nesting if desired.
7. Globally enable all interrupts through register PRUSS_INTC_GER[0] ENABLE_HINT_ANY bit.

6.4.6.4 PRU-ICSS Interrupt Requests Mapping

The PRU-ICSS_0_INTC/ PRU-ICSS_1_INTC lines 0 through 31 are mapped to events which are generated by PRU-ICSS integrated modules. Table 6-366 shows mapping of the different PRU-ICSS internally sourced IRQ events to PRU-ICSS_0_INTC/ PRU-ICSS_1_INTC interrupt lines 0 through 31.

Table 6-366. PRU-ICSS_0/ PRU-ICSS_1 Internal Interrupts

Event Number	PRU-ICSS Internal Interrupt Signal Name	Source
PRU-ICSS_0_INTC		
31	pr0_pru_mst_intr[15]_intr_req	PRU-ICSS_0_PRU0, PRU-ICSS_0_PRU1
30	pr0_pru_mst_intr[14]_intr_req	PRU-ICSS_0_PRU0, PRU-ICSS_0_PRU1
29	pr0_pru_mst_intr[13]_intr_req	PRU-ICSS_0_PRU0, PRU-ICSS_0_PRU1
28	pr0_pru_mst_intr[12]_intr_req	PRU-ICSS_0_PRU0, PRU-ICSS_0_PRU1
27	pr0_pru_mst_intr[11]_intr_req	PRU-ICSS_0_PRU0, PRU-ICSS_0_PRU1
26	pr0_pru_mst_intr[10]_intr_req	PRU-ICSS_0_PRU0, PRU-ICSS_0_PRU1
25	pr0_pru_mst_intr[9]_intr_req	PRU-ICSS_0_PRU0, PRU-ICSS_0_PRU1
24	pr0_pru_mst_intr[8]_intr_req	PRU-ICSS_0_PRU0, PRU-ICSS_0_PRU1
23	pr0_pru_mst_intr[7]_intr_req	PRU-ICSS_0_PRU0, PRU-ICSS_0_PRU1
22	pr0_pru_mst_intr[6]_intr_req	PRU-ICSS_0_PRU0, PRU-ICSS_0_PRU1
21	pr0_pru_mst_intr[5]_intr_req	PRU-ICSS_0_PRU0, PRU-ICSS_0_PRU1
20	pr0_pru_mst_intr[4]_intr_req	PRU-ICSS_0_PRU0, PRU-ICSS_0_PRU1
19	pr0_pru_mst_intr[3]_intr_req	PRU-ICSS_0_PRU0, PRU-ICSS_0_PRU1
18	pr0_pru_mst_intr[2]_intr_req	PRU-ICSS_0_PRU0, PRU-ICSS_0_PRU1
17	pr0_pru_mst_intr[1]_intr_req	PRU-ICSS_0_PRU0, PRU-ICSS_0_PRU1
16	pr0_pru_mst_intr[0]_intr_req	PRU-ICSS_0_PRU0, PRU-ICSS_0_PRU1
15	pr0_ecap_intr_req	PRU-ICSS_0_eCAP0
14	pr0_sync0_out_pend	PRU-ICSS_0_IEP
13	pr0_sync1_out_pend	PRU-ICSS_0_IEP
12	pr0_latch0_in (input to PRU-ICSS_0)	PRU-ICSS_0_IEP
11	pr0_latch1_in (input to PRU-ICSS_0)	PRU-ICSS_0_IEP
10	pr0_pdi_wd_exp_pend	PRU-ICSS_0_IEP
9	pr0_pd_wd_exp_pend	PRU-ICSS_0_IEP
8	pr0_digio_event_req	PRU-ICSS_0_IEP
7	pr0_iep_tim_cap_cmp_pend	PRU-ICSS_0_IEP
6	pr0_uart0_uint_intr_req	PRU-ICSS_0_UART0
5	pr0_uart0_utxevt_intr_req	PRU-ICSS_0_UART0
4	pr0_uart0_urxevt_intr_req	PRU-ICSS_0_UART0
3	pr0_xfr_timeout	PRU-ICSS_0 Scratch Pad
2	pr0_pru1_r31_status_cnt16	PRU-ICSS_0_PRU1 (Shift Capture)
1	pr0_pru0_r31_status_cnt16	PRU-ICSS_0_PRU0 (Shift Capture)
0	pr0_ecc_err_intr	PRU-ICSS_0 ECC Logic
PRU-ICSS_1_INTC		
31	pr1_pru_mst_intr[15]_intr_req	PRU-ICSS_1_PRU0, PRU-ICSS_1_PRU1
30	pr1_pru_mst_intr[14]_intr_req	PRU-ICSS_1_PRU0, PRU-ICSS_1_PRU1
29	pr1_pru_mst_intr[13]_intr_req	PRU-ICSS_1_PRU0, PRU-ICSS_1_PRU1
28	pr1_pru_mst_intr[12]_intr_req	PRU-ICSS_1_PRU0, PRU-ICSS_1_PRU1
27	pr1_pru_mst_intr[11]_intr_req	PRU-ICSS_1_PRU0, PRU-ICSS_1_PRU1
26	pr1_pru_mst_intr[10]_intr_req	PRU-ICSS_1_PRU0, PRU-ICSS_1_PRU1
25	pr1_pru_mst_intr[9]_intr_req	PRU-ICSS_1_PRU0, PRU-ICSS_1_PRU1
24	pr1_pru_mst_intr[8]_intr_req	PRU-ICSS_1_PRU0, PRU-ICSS_1_PRU1

Table 6-366. PRU-ICSS_0/ PRU-ICSS_1 Internal Interrupts (continued)

Event Number	PRU-ICSS Internal Interrupt Signal Name	Source
23	pr1_pru_mst_intr[7]_intr_req	PRU-ICSS_1_PRU0, PRU-ICSS_1_PRU1
22	pr1_pru_mst_intr[6]_intr_req	PRU-ICSS_1_PRU0, PRU-ICSS_1_PRU1
21	pr1_pru_mst_intr[5]_intr_req	PRU-ICSS_1_PRU0, PRU-ICSS_1_PRU1
20	pr1_pru_mst_intr[4]_intr_req	PRU-ICSS_1_PRU0, PRU-ICSS_1_PRU1
19	pr1_pru_mst_intr[3]_intr_req	PRU-ICSS_1_PRU0, PRU-ICSS_1_PRU1
18	pr1_pru_mst_intr[2]_intr_req	PRU-ICSS_1_PRU0, PRU-ICSS_1_PRU1
17	pr1_pru_mst_intr[1]_intr_req	PRU-ICSS_1_PRU0, PRU-ICSS_1_PRU1
16	pr1_pru_mst_intr[0]_intr_req	PRU-ICSS_1_PRU0, PRU-ICSS_1_PRU1
15	pr1_ecap_intr_req	PRU-ICSS_1_eCAP0
14	pr1_sync0_out_pend	PRU-ICSS_1_IEP
13	pr1_sync1_out_pend	PRU-ICSS_1_IEP
12	pr1_latch0_in (input to PRU-ICSS_1)	PRU-ICSS_1_IEP
11	pr1_latch1_in (input to PRU-ICSS_1)	PRU-ICSS_1_IEP
10	pr1_pdi_wd_exp_pend	PRU-ICSS_1_IEP
9	pr1_pd_wd_exp_pend	PRU-ICSS_1_IEP
8	pr1_digio_event_req	PRU-ICSS_1_IEP
7	pr1_iep_tim_cap_cmp_pend	PRU-ICSS_1_IEP
6	pr1_uart_uint_intr_req	PRU-ICSS_1_UART0
5	pr1_uart0_utxevt_intr_req	PRU-ICSS_1_UART0
4	pr1_uart0_urxevt_intr_req	PRU-ICSS_1_UART0
3	pr1_xfr_timeout	PRU-ICSS_1 Scratch Pad
2	pr1_pru1_r31_status_cnt16	PRU-ICSS_1_PRU1 (Shift Capture)
1	pr1_pru0_r31_status_cnt16	PRU-ICSS_1_PRU0 (Shift Capture)
0	pr1_ecc_err_intr	PRU-ICSS_1 ECC Logic

The IRQ input lines 32 through 63 receive interrupts which come from various device peripherals located outside PRU-ICSS_0 and PRU-ICSS_1. They are delivered on the PRU-ICSS_0_INTC/ PRU-ICSS_1_INTC inputs (32 through 63).

Note that for the PRU-ICSS_INTC input lines **32 through 55**, there is an additional multiplexing option programmable in the PRU-ICSS_CFG located register bit - PRUSS_MII_RT[0] MII_RT_EVENT_EN. By default the MII_RT_EVENT_EN is set to 0b0 which selects the IRQ sources to be the PRU-ICSS dedicated device outputs ("**Standard**" mode). By setting MII_RT_EVENT_EN to 0b1, a set of PRU-ICSS MII_RT module associated events, are mapped to the same lines.

The Table 6-367 and the Table 6-368 shows PRU-ICSS_0/PRU-ICSS_1 MII_RT events mapping on the PRU-ICSS_0_INTC / PRU-ICSS_1_INTC inputs 32 through 55 (PRU-ICSS_0) / 32 through 55 (PRU-ICSS_1) valid for the "MII_RT" mode (with PRUSS_MII_RT[0] MII_RT_EVENT_EN register bit set to "0b1").

Table 6-367. PRU-ICSS_0 MII_RT Mode Interrupts

PRU-ICSS_0 IRQ [(1)]	Signal Name (MII_RT Mode enabled) - PRUSS_MII_RT[0] MII_RT_EVENT_EN=0b1
55	(pr1_mii1_col & pr1_mii1_txen) (external signals from device pins)
54	PRU1_RX_EOF
53	MDIO_MII_LINK[1]
52	PORT1_TX_OVERFLOW
51	PORT1_TX_UNDERFLOW
50	PRU1_RX_OVERFLOW

(1) Signals 63–56 and 31–0 for "MII_RT" Mode are the same as for "Standard" Mode.

Table 6-367. PRU-ICSS_0 MII_RT Mode Interrupts (continued)

PRU-ICSS_0 IRQ [1]	Signal Name (MII_RT Mode enabled) - PRUSS_MII_RT[0] MII_RT_EVENT_EN=0b1
49	PRU1_RX_NIBBLE_ODD
48	PRU1_RX_CRC
47	PRU1_RX_SOF
46	PRU1_RX_SFD
45	PRU1_RX_ERR32
44	PRU1_RX_ERR
43	(pr1_mii0_col & pr1_mii0_txen) (external signals from device pins)
42	PRU0_RX_EOF
41	MDIO_MII_LINK[0]
40	PORT0_TX_OVERFLOW
39	PORT0_TX_UNDERFLOW
38	PRU0_RX_OVERFLOW
37	PRU0_RX_NIBBLE_ODD
36	PRU0_RX_CRC
35	PRU0_RX_SOF
34	PRU0_RX_SFD
33	PRU0_RX_ERR32
32	PRU0_RX_ERR

Table 6-368. PRU-ICSS_1 MII_RT Mode Interrupts

PRU-ICSS_1 IRQ [1]	Signal Name (MII_RT Mode enabled) - PRUSS_MII_RT[0] MII_RT_EVENT_EN=0b1
55	(pr1_mii1_col & pr1_mii1_txen) (external signals from device pins)
54	PRU1_RX_EOF
53	MDIO_MII_LINK[1]
52	PORT1_TX_OVERFLOW
51	PORT1_TX_UNDERFLOW
50	PRU1_RX_OVERFLOW
49	PRU1_RX_NIBBLE_ODD
48	PRU1_RX_CRC
47	PRU1_RX_SOF
46	PRU1_RX_SFD
45	PRU1_RX_ERR32
44	PRU1_RX_ERR
43	(pr1_mii0_col & pr1_mii0_txen) (external signals from device pins)
42	PRU0_RX_EOF
41	MDIO_MII_LINK[0]
40	PORT0_TX_OVERFLOW
39	PORT0_TX_UNDERFLOW
38	PRU0_RX_OVERFLOW
37	PRU0_RX_NIBBLE_ODD
36	PRU0_RX_CRC
35	PRU0_RX_SOF
34	PRU0_RX_SFD
33	PRU0_RX_ERR32

(1) Signals 63–56 and 31–0 for "MII_RT" Mode are the same as for "Standard" Mode.

Table 6-368. PRU-ICSS_1 MII_RT Mode Interrupts (continued)

PRU-ICSS_1 IRQ [1]	Signal Name (MII_RT Mode enabled) - PRUSS_MII_RT[0] MII_RT_EVENT_EN=0b1
32	PRU0_RX_ERR

Table 6-369. PRU-ICSS_0 System Events

Event Number	Signal Name (MII_RT Mode disabled) - PRUSS_MII_RTPRUSS_MII_RT[0] MII_RT_EVENT_EN=0b0	Source
63	EDMACC_1_REGION_6_INT	EDMACC_1 Region 6 DMA completion interrupt
62	EDMACC_0_REGION_6_INT	EDMACC_0 Region 6 DMA completion interrupt
61	MSGMGR_QUE_PEND_59	Message Manager (MSGMGR) que 59 pending interrupt
60	MSGMGR_QUE_PEND_58	Message Manager (MSGMGR) que 58 pending interrup
59	MSGMGR_QUE_PEND_7	Message Manager (MSGMGR) que 7 pending interrup
58	MSGMGR_QUE_PEND_6	Message Manager (MSGMGR) que 6 pending interrup
57	IPC_GR12	BOOT_CFG Interprocessor communication. Interrupt generated by writing 1h to IPCGR12[0] IPCGR12_REG
56	ICSS1_HOST_INT	PRU-ICSS_1 Host Interrupt 7 (exported as pr1_host_intr5_intr_pend_req)
55	IPC_GR11	BOOT_CFG Interprocessor communication. Interrupt generated by writing 1h to IPCGR11[0] IPCGR11_REG
54	CIC_0_OUT102	Chip-level Interrupt Controller (CIC) output 102 host interrupt
53	CIC_0_OUT101	Chip-level Interrupt Controller (CIC) output 101 host interrupt
52	CIC_0_OUT100	Chip-level Interrupt Controller (CIC) output 100 host interrupt
51	CIC_0_OUT99	Chip-level Interrupt Controller (CIC) output 99 host interrupt
50	CIC_0_OUT98	Chip-level Interrupt Controller (CIC) output 98 host interrupt
49	CIC_0_OUT97	Chip-level Interrupt Controller (CIC) output 97 host interrupt
48	CIC_0_OUT96	Chip-level Interrupt Controller (CIC) output 96 host interrupt
47	CIC_0_OUT95	Chip-level Interrupt Controller (CIC) output 95 host interrupt
46	CIC_0_OUT94	Chip-level Interrupt Controller (CIC) output 94 host interrupt
45	CIC_0_OUT93	Chip-level Interrupt Controller (CIC) output 93 host interrupt
44	CIC_0_OUT92	Chip-level Interrupt Controller (CIC) output 92 host interrupt
43	CIC_0_OUT91	Chip-level Interrupt Controller (CIC) output 91 host interrupt
42	CIC_0_OUT90	Chip-level Interrupt Controller (CIC) output 90 host interrupt
41	CIC_0_OUT89	Chip-level Interrupt Controller (CIC) output 89 host interrupt
40	CIC_0_OUT88	Chip-level Interrupt Controller (CIC) output 88 host interrupt
39	CIC_0_OUT83	Chip-level Interrupt Controller (CIC) output 83 host interrupt
38	CIC_0_OUT82	Chip-level Interrupt Controller (CIC) output 82 host interrupt

Table 6-369. PRU-ICSS_0 System Events (continued)

Event Number	Signal Name (MII_RT Mode disabled) - PRUSS_MII_RTPRUSS_MII_RT[0] MII_RT_EVENT_EN=0b0	Source
37	CIC_0_OUT81	Chip-level Interrupt Controller (CIC) output 81 host interrupt
36	CIC_0_OUT80	Chip-level Interrupt Controller (CIC) output 80 host interrupt
35	I2C_0_INT	I2C_0 interrupt
34	SPI_1_INT0	SPI_1 Level 0 interrupt
33	SPI_0_INT0	SPI_0 Level 0 interrupt
32	QSPI_INT	QSPI interrupt

Table 6-370. PRU-ICSS_1 System Events

Event Number	Signal Name (MII_RT Mode disabled) - PRUSS_MII_RT[0] MII_RT_EVENT_EN=0b0	Source
63	EDMACC_1_REGION_7_INT	EDMACC_1 Region 7 DMA completion interrupt
62	EDMACC_0_REGION_7_INT	EDMACC_0 Region 7 DMA completion interrupt
61	MSGMGR_QUE_PEND_61	Message Manager (MSGMGR) que 61 pending interrupt
60	MSGMGR_QUE_PEND_60	Message Manager (MSGMGR) que 60 pending interrupt
59	MSGMGR_QUE_PEND_9	Message Manager (MSGMGR) que 9 pending interrupt
58	MSGMGR_QUE_PEND_8	Message Manager (MSGMGR) que 8 pending interrupt
57	IPC_GR14	BOOT_CFG Interprocessor communication. Interrupt generated by writing 1h to IPCGR14[0] IPCGR14_REG
56	ICSS0_HOST_INT	PRU-ICSS_0 Host Interrupt 7 (exported as pr0_host_intr5_intr_pend_req)
55	IPC_GR13	BOOT_CFG Interprocessor communication. Interrupt generated by writing 1h to IPCGR13[0] IPCGR13_REG
54	PWM_SOCB	Start of conversion B event. ePWM_x (where x = 0 to 5) or PRU-ICSS1
53	PWM_SOCA	Start of conversion A event. ePWM_x (where x = 0 to 5) or PRU-ICSS1
52	CIC_0_OUT87	Chip-level Interrupt Controller (CIC) output 87 host interrupt
51	CIC_0_OUT86	Chip-level Interrupt Controller (CIC) output 86 host interrupt
50	CIC_0_OUT85	Chip-level Interrupt Controller (CIC) output 85 host interrupt
49	CIC_0_OUT84	Chip-level Interrupt Controller (CIC) output 84 host interrupt
48	CIC_0_OUT83	Chip-level Interrupt Controller (CIC) output 83 host interrupt
47	CIC_0_OUT82	Chip-level Interrupt Controller (CIC) output 82 host interrupt
46	CIC_0_OUT81	Chip-level Interrupt Controller (CIC) output 81 host interrupt
45	CIC_0_OUT80	Chip-level Interrupt Controller (CIC) output 80 host interrupt
44	EQEP_2_INT	eQEP_2 interrupt
43	EQEP_1_INT	eQEP_1 interrupt
42	EQEP_0_INT	eQEP_0 interrupt
41	EPWM_3_TRIP_ZONE	ePWM_3 interrupt
40	EPWM_0_TRIP_ZONE	ePWM_0 interrupt
39	EPWM_5_INT	ePWM_5 interrupt

Table 6-370. PRU-ICSS_1 System Events (continued)

Event Number	Signal Name (MII_RT Mode disabled) - PRUSS_MII_RT[0] MII_RT_EVENT_EN=0b0	Source
38	EPWM_4_INT	ePWM_4 interrupt
37	EPWM_3_INT	ePWM_3 interrupt
36	EPWM_2_INT	ePWM_2 interrupt
35	EPWM_1_INT	ePWM_1 interrupt
34	EPWM_0_INT	ePWM_0 interrupt
33	ECAP_1_INT	eCAP_1 interrupt
32	ECAP_0_INT	eCAP_0 interrupt

6.4.6.5 PRU-ICSS Interrupt Controller Registers

Table 6-372 lists the memory-mapped registers for the PRU-ICSS interrupt controller. All register offset addresses not listed in Table 6-372 should be considered as reserved locations and the register contents should not be modified.

Table 6-371. PRU-ICSS_INTC Instances

Instance	Base Address
PRU_ICSS_0_INTC	20AA 0000h
PRU_ICSS_1_INTC	20AE 0000h

Table 6-372. PRU-ICSS_INTC Registers

Offset	Acronym	Register Name	PRU_ICSS_0_ INTC Physical Address	PRU_ICSS_1_ INTC Physical Address	Section
0h	PRUSS_INTC_REVID	Revision ID Register	20AA 0000h	20AE 0000h	Section 6.4.6.5.1
4h	PRUSS_INTC_CR	Control Register	20AA 0004h	20AE 0004h	Section 6.4.6.5.2
10h	PRUSS_INTC_GER	Global Host Interrupt Enable Register	20AA 0010h	20AE 0010h	Section 6.4.6.5.3
1Ch	PRUSS_INTC_GNLR	Global Nesting Level Register	20AA 001Ch	20AE 001Ch	Section 6.4.6.5.4
20h	PRUSS_INTC_SISR	System Interrupt Status Indexed Set Register	20AA 0020h	20AE 0020h	Section 6.4.6.5.5
24h	PRUSS_INTC_SICR	System Interrupt Status Indexed Clear Register	20AA 0024h	20AE 0024h	Section 6.4.6.5.6
28h	PRUSS_INTC_EISR	System Interrupt Enable Indexed Set Register	20AA 0028h	20AE 0028h	Section 6.4.6.5.7
2Ch	PRUSS_INTC_EICR	System Interrupt Enable Indexed Clear Register	20AA 002Ch	20AE 002Ch	Section 6.4.6.5.8
34h	PRUSS_INTC_HIEISR	Host Interrupt Enable Indexed Set Register	20AA 0034h	20AE 0034h	Section 6.4.6.5.9
38h	PRUSS_INTC_HIDISR	Host Interrupt Enable Indexed Clear Register	20AA 0038h	20AE 0038h	Section 6.4.6.5.10
80h	PRUSS_INTC_GPIR	Global Prioritized Index Register	20AA 0080h	20AE 0080h	Section 6.4.6.5.11
200h	PRUSS_INTC_SRSR0	System Interrupt Status Raw Set Register0	20AA 0200h	20AE 0200h	Section 6.4.6.5.12
204h	PRUSS_INTC_SRSR1	System Interrupt Status Raw Set Register1	20AA 0204h	20AE 0204h	Section 6.4.6.5.13
280h	PRUSS_INTC_SECR0	System Interrupt Status Enabled Clear Register0	20AA 0280h	20AE 0280h	Section 6.4.6.5.14
284h	PRUSS_INTC_SECR1	System Interrupt Status Enabled Clear Register1	20AA 0284h	20AE 0284h	Section 6.4.6.5.15
300h	PRUSS_INTC_ESR0	System Interrupt Enable Set Register0	20AA 0300h	20AE 0300h	Section 6.4.6.5.16
304h	PRUSS_INTC_ERS1	System Interrupt Enable Set Register1	20AA 0304h	20AE 0304h	Section 6.4.6.5.17
380h	PRUSS_INTC_ECR0	System Interrupt Enable Clear Register0	20AA 0380h	20AE 0380h	Section 6.4.6.5.18
384h	PRUSS_INTC_ECR1	System Interrupt Enable Clear Register1	20AA 0384h	20AE 0384h	Section 6.4.6.5.19
400h	PRUSS_INTC_CMR_0	Channel Map Register_0	20AA 0400h	20AE 0400h	Section 6.4.6.5.20
404h	PRUSS_INTC_CMR_1	Channel Map Register_1	20AA 0404h	20AE 0404h	Section 6.4.6.5.21

Table 6-372. PRU-ICSS_INTC Registers (continued)

Offset	Acronym	Register Name	PRU_ICSS_0_ INTC Physical Address	PRU_ICSS_1_ INTC Physical Address	Section
408h	PRUSS_INTC_CMR_2	Channel Map Register_2	20AA 0408h	20AE 0408h	Section 6.4.6.5.22
40Ch	PRUSS_INTC_CMR_3	Channel Map Register_3	20AA 040Ch	20AE 040Ch	Section 6.4.6.5.23
410h	PRUSS_INTC_CMR_4	Channel Map Register_4	20AA 0410h	20AE 0410h	Section 6.4.6.5.24
414h	PRUSS_INTC_CMR_5	Channel Map Register_5	20AA 0414h	20AE 0414h	Section 6.4.6.5.25
418h	PRUSS_INTC_CMR_6	Channel Map Register_6	20AA 0418h	20AE 0418h	Section 6.4.6.5.26
41Ch	PRUSS_INTC_CMR_7	Channel Map Register_7	20AA 041Ch	20AE 041Ch	Section 6.4.6.5.27
420h	PRUSS_INTC_CMR_8	Channel Map Register_8	20AA 0420h	20AE 0420h	Section 6.4.6.5.28
424h	PRUSS_INTC_CMR_9	Channel Map Register_9	20AA 0424h	20AE 0424h	Section 6.4.6.5.29
428h	PRUSS_INTC_CMR_10	Channel Map Register_10	20AA 0428h	20AE 0428h	Section 6.4.6.5.30
42Ch	PRUSS_INTC_CMR_11	Channel Map Register_11	20AA 042Ch	20AE 042Ch	Section 6.4.6.5.31
430h	PRUSS_INTC_CMR_12	Channel Map Register_12	20AA 0430h	20AE 0430h	Section 6.4.6.5.32
434h	PRUSS_INTC_CMR_13	Channel Map Register_13	20AA 0434h	20AE 0434h	Section 6.4.6.5.33
438h	PRUSS_INTC_CMR_14	Channel Map Register_14	20AA 0438h	20AE 0438h	Section 6.4.6.5.34
43Ch	PRUSS_INTC_CMR_15	Channel Map Register_15	20AA 043Ch	20AE 043Ch	Section 6.4.6.5.35
800h	PRUSS_INTC_HMR0	Host Interrupt Map Register0	20AA 0800h	20AE 0800h	Section 6.4.6.5.36
804h	PRUSS_INTC_HMR1	Host Interrupt Map Register1	20AA 0804h	20AE 0804h	Section 6.4.6.5.37
808h	PRUSS_INTC_HMR2	Host Interrupt Map Register2	20AA 0808h	20AE 0808h	Section 6.4.6.5.38
900h	PRUSS_INTC_HIPIR_0	Host Interrupt Prioritized Index Register_0	20AA 0900h	20AE 0900h	Section 6.4.6.5.39
904h	PRUSS_INTC_HIPIR_1	Host Interrupt Prioritized Index Register_1	20AA 0904h	20AE 0904h	Section 6.4.6.5.40
908h	PRUSS_INTC_HIPIR_2	Host Interrupt Prioritized Index Register_2	20AA 0908h	20AE 0908h	Section 6.4.6.5.41
90Ch	PRUSS_INTC_HIPIR_3	Host Interrupt Prioritized Index Register_3	20AA 090Ch	20AE 090Ch	Section 6.4.6.5.42
910h	PRUSS_INTC_HIPIR_4	Host Interrupt Prioritized Index Register_4	20AA 0910h	20AE 0910h	Section 6.4.6.5.43
914h	PRUSS_INTC_HIPIR_5	Host Interrupt Prioritized Index Register_5	20AA 0914h	20AE 0914h	Section 6.4.6.5.44
918h	PRUSS_INTC_HIPIR_6	Host Interrupt Prioritized Index Register_6	20AA 0918h	20AE 0918h	Section 6.4.6.5.45
91Ch	PRUSS_INTC_HIPIR_7	Host Interrupt Prioritized Index Register_7	20AA 091Ch	20AE 091Ch	Section 6.4.6.5.46
920h	PRUSS_INTC_HIPIR_8	Host Interrupt Prioritized Index Register_8	20AA 0920h	20AE 0920h	Section 6.4.6.5.47
924h	PRUSS_INTC_HIPIR_9	Host Interrupt Prioritized Index Register_9	20AA 0924h	20AE 0924h	Section 6.4.6.5.48

Table 6-372. PRU-ICSS_INTC Registers (continued)

Offset	Acronym	Register Name	PRU_ICSS_0_ INTC Physical Address	PRU_ICSS_1_ INTC Physical Address	Section
D00h	PRUSS_INTC_SIPR0	System Interrupt Polarity Register0	20AA 0D00h	20AE 0D00h	Section 6.4.6.5.49
D04h	PRUSS_INTC_SIPR1	System Interrupt Polarity Register1	20AA 0D04h	20AE 0D04h	Section 6.4.6.5.50
D80h	PRUSS_INTC_SITR0	System Interrupt Type Register0	20AA 0D80h	20AE 0D80h	Section 6.4.6.5.51
D84h	PRUSS_INTC_SITR1	System Interrupt Type Register1	20AA 0D84h	20AE 0D84h	Section 6.4.6.5.52
1100h	PRUSS_INTC_HINLR_0	Host Interrupt Nesting Level Register_0	20AA 1100h	20AE 1100h	Section 6.4.6.5.53
1104h	PRUSS_INTC_HINLR_1	Host Interrupt Nesting Level Register_1	20AA 1104h	20AE 1104h	Section 6.4.6.5.54
1108h	PRUSS_INTC_HINLR_2	Host Interrupt Nesting Level Register_2	20AA 1108h	20AE 1108h	Section 6.4.6.5.55
110Ch	PRUSS_INTC_HINLR_3	Host Interrupt Nesting Level Register_3	20AA 110Ch	20AE 110Ch	Section 6.4.6.5.56
1110h	PRUSS_INTC_HINLR_4	Host Interrupt Nesting Level Register_4	20AA 1110h	20AE 1110h	Section 6.4.6.5.57
1114h	PRUSS_INTC_HINLR_5	Host Interrupt Nesting Level Register_5	20AA 1114h	20AE 1114h	Section 6.4.6.5.58
1118h	PRUSS_INTC_HINLR_6	Host Interrupt Nesting Level Register_6	20AA 1118h	20AE 1118h	Section 6.4.6.5.59
111Ch	PRUSS_INTC_HINLR_7	Host Interrupt Nesting Level Register_7	20AA 111Ch	20AE 111Ch	Section 6.4.6.5.60
1120h	PRUSS_INTC_HINLR_8	Host Interrupt Nesting Level Register_8	20AA 1120h	20AE 1120h	Section 6.4.6.5.61
1124h	PRUSS_INTC_HINLR_9	Host Interrupt Nesting Level Register_9	20AA 1124h	20AE 1124h	Section 6.4.6.5.62
1500h	PRUSS_INTC_HIER	Host Interrupt Enable Registers	20AA 1500h	20AE 1500h	Section 6.4.6.5.63

6.4.6.5.1 PRUSS_INTC_REVID Register (Offset = 0h) [reset = 0h]

PRUSS_INTC_REVID is shown in Figure 6-142 and described in Table 6-374.

Revision ID Register

Table 6-373. PRUSS_INTC_REVID Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0000h
PRU_ICSS_1_INTC	20AE 0000h

Figure 6-142. PRUSS_INTC_REVID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REVISION																															
R--h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-374. PRUSS_INTC_REVID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REVISION	R	-h	IP Revision

Table 6-375. Register Call Summary for PRUSS_INTC_REVID

PRU-ICSS Local Interrupt Controller

- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_REVID Register (Offset = 0h) [reset = 0h]: [0]

6.4.6.5.2 PRUSS_INTC_CR Register (Offset = 4h) [reset = 0h]

PRUSS_INTC_CR is shown in Figure 6-143 and described in Table 6-377.

The Control Register holds global control parameters and can forces a soft reset on the module.

Table 6-376. PRUSS_INTC_CR Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0004h
PRU_ICSS_1_INTC	20AE 0004h

Figure 6-143. PRUSS_INTC_CR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			PRIORITY_HOLD_MODE	NEST_MODE		WAKEUP_MODE	RESERVED
R-0h			R/W-0h	R/W-0h		R/W-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-377. PRUSS_INTC_CR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4	PRIORITY_HOLD_MODE	R/W	0h	Reserved
3-2	NEST_MODE	R/W	0h	The nesting mode. 0h: No nesting 1h: Automatic individual nesting (per host interrupt) 2h: Automatic global nesting (over all host interrupts) 3h: Manual nesting
1	WAKEUP_MODE	R/W	0h	Reserved
0	RESERVED	R	0h	Reserved

Table 6-378. Register Call Summary for PRUSS_INTC_CR

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CR Register (Offset = 4h) [reset = 0h]: [0]

6.4.6.5.3 PRUSS_INTC_GER Register (Offset = 10h) [reset = 0h]

PRUSS_INTC_GER is shown in Figure 6-144 and described in Table 6-380.

The Global Host Interrupt Enable Register enables all the host interrupts. Individual host interrupts are still enabled or disabled from their individual enables and are not overridden by the global enable.

Table 6-379. PRUSS_INTC_GER Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0010h
PRU_ICSS_1_INTC	20AE 0010h

Figure 6-144. PRUSS_INTC_GER Register

31	30	29	28	27	26	25	24
RESERVED							
R-0000 000h							
23	22	21	20	19	18	17	16
RESERVED							
R-0000 000h							
15	14	13	12	11	10	9	8
RESERVED							
R-0000 000h							
7	6	5	4	3	2	1	0
RESERVED							ENABLE_HINT_ANY
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-380. PRUSS_INTC_GER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0000 000h	Reserved
0	ENABLE_HINT_ANY	R/W	0h	The current global enable value when read. Writes set the global enable.

Table 6-381. Register Call Summary for PRUSS_INTC_GER

PRU-ICSS Local Interrupt Controller

- PRU-ICSS Interrupt Controller Basic Programming Model: [0]
- PRU-ICSS Interrupt Controller Registers: [0]
- PRU-ICSS Interrupt Processing: [0]
- PRUSS_INTC_GER Register (Offset = 10h) [reset = 0h]: [0]

6.4.6.5.4 PRUSS_INTC_GNLR Register (Offset = 1Ch) [reset = 100h]

PRUSS_INTC_GNLR is shown in Figure 6-145 and described in Table 6-383.

The Global Nesting Level Register allows the checking and setting of the global nesting level across all host interrupts when automatic global nesting mode is set. The nesting level is the channel (and all of lower priority) that are nested out because of a current interrupt. This register is only available when nesting is configured.

Table 6-382. PRUSS_INTC_GNLR Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 001Ch
PRU_ICSS_1_INTC	20AE 001Ch

Figure 6-145. PRUSS_INTC_GNLR Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							GLB_NEST_LEVEL
R-0h							R/W-100h
7	6	5	4	3	2	1	0
GLB_NEST_LEVEL							
R/W-100h							

LEGEND: R = Read Only; R/W = Read/Write; W = Write Only; -*n* = value after reset

Table 6-383. PRUSS_INTC_GNLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Always read as 0. Writes of 1 override the automatic nesting and set the nesting_level to the written data.
30-9	RESERVED	R	0h	Reserved
8-0	GLB_NEST_LEVEL	R/W	100h	The current global nesting level (highest channel that is nested). Writes set the nesting level. In auto nesting mode this value is updated internally unless the auto_override bit is set.

Table 6-384. Register Call Summary for PRUSS_INTC_GNLR

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRU-ICSS Interrupt Nesting: [0]
- PRUSS_INTC_GNLR Register (Offset = 1Ch) [reset = 100h]: [0]

6.4.6.5.5 PRUSS_INTC_SISR Register (Offset = 20h) [reset = 0h]

PRUSS_INTC_SISR is shown in Figure 6-146 and described in Table 6-386.

The System Interrupt Status Indexed Set Register allows setting the status of an interrupt. The interrupt to set is the index value written. This sets the Raw Status Register bit of the given index.

Table 6-385. PRUSS_INTC_SISR Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0020h
PRU_ICSS_1_INTC	20AE 0020h

Figure 6-146. PRUSS_INTC_SISR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0000 00h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						STATUS_SET_INDEX									
R-0000 00h						W-0h									

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 6-386. PRUSS_INTC_SISR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0000 00h	Reserved
9-0	STATUS_SET_INDEX	W	0h	Writes set the status of the interrupt given in the index value. Reads return 0.

Table 6-387. Register Call Summary for PRUSS_INTC_SISR

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_SISR Register (Offset = 20h) [reset = 0h]: [0]

6.4.6.5.6 PRUSS_INTC_SICR Register (Offset = 24h) [reset = 0h]

PRUSS_INTC_SICR is shown in Figure 6-147 and described in Table 6-389.

The System Interrupt Status Indexed Clear Register allows clearing the status of an interrupt. The interrupt to clear is the index value written. This clears the Raw Status Register bit of the given index.

Table 6-388. PRUSS_INTC_SICR Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0024h
PRU_ICSS_1_INTC	20AE 0024h

Figure 6-147. PRUSS_INTC_SICR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0000 00h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						STATUS_CLR_INDEX									
R-0000 00h						W-0h									

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 6-389. PRUSS_INTC_SICR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0000 00h	Reserved
9-0	STATUS_CLR_INDEX	W	0h	Writes clear the status of the interrupt given in the index value. Reads return 0.

Table 6-390. Register Call Summary for PRUSS_INTC_SICR

PRU-ICSS Local Interrupt Controller

- PRU-ICSS Interrupt Controller Registers: [0]
- PRU-ICSS Interrupt Status Clearing: [0]
- PRUSS_INTC_SICR Register (Offset = 24h) [reset = 0h]: [0]

6.4.6.5.7 PRUSS_INTC_EISR Register (Offset = 28h) [reset = 0h]

PRUSS_INTC_EISR is shown in Figure 6-148 and described in Table 6-392.

The System Interrupt Enable Indexed Set Register allows enabling an interrupt. The interrupt to enable is the index value written. This sets the Enable Register bit of the given index.

Table 6-391. PRUSS_INTC_EISR Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0028h
PRU_ICSS_1_INTC	20AE 0028h

Figure 6-148. PRUSS_INTC_EISR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0000 00h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						ENABLE_SET_INDEX									
R-0000 00h						W-0h									

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 6-392. PRUSS_INTC_EISR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0000 00h	Reserved
9-0	ENABLE_SET_INDEX	W	0h	Writes set the enable of the interrupt given in the index value. Reads return 0.

Table 6-393. Register Call Summary for PRUSS_INTC_EISR

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_EISR Register (Offset = 28h) [reset = 0h]: [0]
- PRU-ICSS Interrupt Processing: [0]

6.4.6.5.8 PRUSS_INTC_EICR Register (Offset = 2Ch) [reset = 0h]

PRUSS_INTC_EICR is shown in Figure 6-149 and described in Table 6-395.

The System Interrupt Enable Indexed Clear Register allows disabling an interrupt. The interrupt to disable is the index value written. This clears the Enable Register bit of the given index.

Table 6-394. PRUSS_INTC_EICR Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 002Ch
PRU_ICSS_1_INTC	20AE 002Ch

Figure 6-149. PRUSS_INTC_EICR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0000 00h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						ENABLE_CLR_INDEX									
R-0000 00h						W-0h									

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 6-395. PRUSS_INTC_EICR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0000 00h	Reserved
9-0	ENABLE_CLR_INDEX	W	0h	Writes clear the enable of the interrupt given in the index value. Reads return 0.

Table 6-396. Register Call Summary for PRUSS_INTC_EICR

PRU-ICSS Local Interrupt Controller

- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_EICR Register (Offset = 2Ch) [reset = 0h]: [0]
- PRU-ICSS Interrupt Disabling: [0]

6.4.6.5.9 PRUSS_INTC_HIEISR Register (Offset = 34h) [reset = 0h]

PRUSS_INTC_HIEISR is shown in Figure 6-150 and described in Table 6-398.

The Host Interrupt Enable Indexed Set Register allows enabling a host interrupt output. The host interrupt to enable is the index value written. This enables the host interrupt output or triggers the output again if already enabled.

Table 6-397. PRUSS_INTC_HIEISR Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0034h
PRU_ICSS_1_INTC	20AE 0034h

Figure 6-150. PRUSS_INTC_HIEISR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0000 00h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						HINT_ENABLE_SET_INDEX									
R-0000 00h						R/W-0h									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-398. PRUSS_INTC_HIEISR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0000 00h	Reserved
9-0	HINT_ENABLE_SET_INDEX	R/W	0h	Writes set the enable of the host interrupt given in the index value. Reads return 0.

Table 6-399. Register Call Summary for PRUSS_INTC_HIEISR

PRU-ICSS Local Interrupt Controller

- PRU-ICSS Interrupt Controller Basic Programming Model: [0]
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HIEISR Register (Offset = 34h) [reset = 0h]: [0]
- PRU-ICSS Interrupt Processing: [0]

6.4.6.5.10 PRUSS_INTC_HIDISR Register (Offset = 38h) [reset = 0h]

PRUSS_INTC_HIDISR is shown in Figure 6-151 and described in Table 6-401.

The Host Interrupt Enable Indexed Clear Register allows disabling a host interrupt output. The host interrupt to disable is the index value written. This disables the host interrupt output.

Table 6-400. PRUSS_INTC_HIDISR Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0038h
PRU_ICSS_1_INTC	20AE 0038h

Figure 6-151. PRUSS_INTC_HIDISR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0000 00h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						HINT_ENABLE_CLR_INDEX									
R-0000 00h						R/W-0h									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-401. PRUSS_INTC_HIDISR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0000 00h	Reserved
9-0	HINT_ENABLE_CLR_INDEX	R/W	0h	Writes clear the enable of the host interrupt given in the index value. Reads return 0.

Table 6-402. Register Call Summary for PRUSS_INTC_HIDISR

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HIDISR Register (Offset = 38h) [reset = 0h]: [0]

6.4.6.5.11 PRUSS_INTC_GPIR Register (Offset = 80h) [reset = 80000000h]

PRUSS_INTC_GPIR is shown in Figure 6-152 and described in Table 6-404.

The Global Prioritized Index Register shows the interrupt number of the highest priority interrupt pending across all the host interrupts.

Table 6-403. PRUSS_INTC_GPIR Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0080h
PRU_ICSS_1_INTC	20AE 0080h

Figure 6-152. PRUSS_INTC_GPIR Register

31	30	29	28	27	26	25	24
GLB_NONE	RESERVED						
R-1h	R-0000 00h						
23	22	21	20	19	18	17	16
RESERVED							
R-0000 00h							
15	14	13	12	11	10	9	8
RESERVED						GLB_PRI_INTR	
R-0000 00h						R-0h	
7	6	5	4	3	2	1	0
GLB_PRI_INTR							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 6-404. PRUSS_INTC_GPIR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GLB_NONE	R	1h	No Interrupt is pending. Can be used by host to test for a negative value to see if no interrupts are pending.
30-10	RESERVED	R	0000 00h	Reserved
9-0	GLB_PRI_INTR	R	0h	The currently highest priority interrupt index pending across all the host interrupts.

Table 6-405. Register Call Summary for PRUSS_INTC_GPIR

PRU-ICSS Local Interrupt Controller

- PRU-ICSS Interrupt Controller Registers: [0]
- PRU-ICSS Interrupt Channel Mapping: [0]
- PRUSS_INTC_GPIR Register (Offset = 80h) [reset = 80000000h]: [0]

6.4.6.5.12 PRUSS_INTC_SRSR0 Register (Offset = 200h) [reset = 0h]

PRUSS_INTC_SRSR0 is shown in Figure 6-153 and described in Table 6-407.

The System Interrupt Status Raw Set Register0 show the pending enabled status of the system interrupts 0 to 31. Software can write to the Status Set Registers to set a system interrupt without a hardware trigger. There is one bit per system interrupt.

Table 6-406. PRUSS_INTC_SRSR0 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0200h
PRU_ICSS_1_INTC	20AE 0200h

Figure 6-153. PRUSS_INTC_SRSR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAW_STATUS_31_0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-407. PRUSS_INTC_SRSR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RAW_STATUS_31_0	R/W	0h	System interrupt raw status and setting of the system interrupts 0 to 31. Reads return the raw status. Write a 1 in a bit position to set the status of the system interrupt. Writing a 0 has no effect.

Table 6-408. Register Call Summary for PRUSS_INTC_SRSR0

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_SRSR0 Register (Offset = 200h) [reset = 0h]: [0]

6.4.6.5.13 PRUSS_INTC_SRSR1 Register (Offset = 204h) [reset = 0h]

PRUSS_INTC_SRSR1 is shown in Figure 6-154 and described in Table 6-410.

The System Interrupt Status Raw Set Register1 show the pending enabled status of the system interrupts 32 to 63. Software can write to the Status Set Registers to set a system interrupt without a hardware trigger. There is one bit per system interrupt.

Table 6-409. PRUSS_INTC_SRSR1 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0204h
PRU_ICSS_1_INTC	20AE 0204h

Figure 6-154. PRUSS_INTC_SRSR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAW_STATUS_63_32																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-410. PRUSS_INTC_SRSR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RAW_STATUS_63_32	R/W	0h	System interrupt raw status and setting of the system interrupts 32 to 63. Reads return the raw status. Write a 1 in a bit position to set the status of the system interrupt. Writing a 0 has no effect.

Table 6-411. Register Call Summary for PRUSS_INTC_SRSR1

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_SRSR1 Register (Offset = 204h) [reset = 0h]: [0]

6.4.6.5.14 PRUSS_INTC_SECR0 Register (Offset = 280h) [reset = 0h]

PRUSS_INTC_SECR0 is shown in Figure 6-155 and described in Table 6-413.

The System Interrupt Status Enabled Clear Register0 show the pending enabled status of the system interrupts 0 to 31. Software can write to the Status Clear Registers to clear a system interrupt after it has been serviced. If a system interrupt status is not cleared then another host interrupt may not be triggered or another host interrupt may be triggered incorrectly. There is one bit per system interrupt.

Table 6-412. PRUSS_INTC_SECR0 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0280h
PRU_ICSS_1_INTC	20AE 0280h

Figure 6-155. PRUSS_INTC_SECR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENA_STATUS_31_0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-413. PRUSS_INTC_SECR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ENA_STATUS_31_0	R/W	0h	System interrupt enabled status and clearing of the system interrupts 0 to 31. Reads return the enabled status (before enabling with the Enable Registers). Write a 1 in a bit position to clear the status of the system interrupt. Writing a 0 has no effect.

Table 6-414. Register Call Summary for PRUSS_INTC_SECR0

PRU-ICSS Local Interrupt Controller

- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_SECR0 Register (Offset = 280h) [reset = 0h]: [0]
- PRU-ICSS Interrupt Status Clearing: [0]
- PRU-ICSS Interrupt Status Checking: [0][1]

6.4.6.5.15 PRUSS_INTC_SECR1 Register (Offset = 284h) [reset = 0h]

PRUSS_INTC_SECR1 is shown in Figure 6-156 and described in Table 6-416.

The System Interrupt Status Enabled Clear Register1 show the pending enabled status of the system interrupts 32 to 63. Software can write to the Status Clear Registers to clear a system interrupt after it has been serviced. If a system interrupt status is not cleared then another host interrupt may not be triggered or another host interrupt may be triggered incorrectly. There is one bit per system interrupt.

Table 6-415. PRUSS_INTC_SECR1 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0284h
PRU_ICSS_1_INTC	20AE 0284h

Figure 6-156. PRUSS_INTC_SECR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENA_STATUS_63_32																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-416. PRUSS_INTC_SECR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ENA_STATUS_63_32	R/W	0h	System interrupt enabled status and clearing of the system interrupts 32 to 63. Reads return the enabled status (before enabling with the Enable Registers). Write a 1 in a bit position to clear the status of the system interrupt. Writing a 0 has no effect.

Table 6-417. Register Call Summary for PRUSS_INTC_SECR1

PRU-ICSS Local Interrupt Controller

- PRU-ICSS Interrupt Controller Registers: [0]
- PRU-ICSS Interrupt Status Clearing: [0]
- PRU-ICSS Interrupt Status Checking: [0][1]
- PRUSS_INTC_SECR1 Register (Offset = 284h) [reset = 0h]: [0]

6.4.6.5.16 PRUSS_INTC_ESR0 Register (Offset = 300h) [reset = 0h]

PRUSS_INTC_ESR0 is shown in Figure 6-157 and described in Table 6-419.

The System Interrupt Enable Set Register0 enables system interrupts 0 to 31 to trigger outputs. System interrupts that are not enabled do not interrupt the host. There is a bit per system interrupt.

Table 6-418. PRUSS_INTC_ESR0 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0300h
PRU_ICSS_1_INTC	20AE 0300h

Figure 6-157. PRUSS_INTC_ESR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENABLE_SET_31_0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-419. PRUSS_INTC_ESR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ENABLE_SET_31_0	R/W	0h	System interrupt enables system interrupts 0 to 31. Read returns the enable value (0 = disabled, 1 = enabled). Write a 1 in a bit position to set that enable. Writing a 0 has no effect.

Table 6-420. Register Call Summary for PRUSS_INTC_ESR0

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_ESR0 Register (Offset = 300h) [reset = 0h]: [0]

6.4.6.5.17 PRUSS_INTC_ERS1 Register (Offset = 304h) [reset = 0h]

PRUSS_INTC_ERS1 is shown in Figure 6-158 and described in Table 6-422.

The System Interrupt Enable Set Register1 enables system interrupts 32 to 63 to trigger outputs. System interrupts that are not enabled do not interrupt the host. There is a bit per system interrupt.

Table 6-421. PRUSS_INTC_ERS1 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0304h
PRU_ICSS_1_INTC	20AE 0304h

Figure 6-158. PRUSS_INTC_ERS1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENABLE_SET_63_32																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-422. PRUSS_INTC_ERS1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ENABLE_SET_63_32	R/W	0h	System interrupt enables system interrupts 32 to 63. Read returns the enable value (0 = disabled, 1 = enabled). Write a 1 in a bit position to set that enable. Writing a 0 has no effect.

Table 6-423. Register Call Summary for PRUSS_INTC_ERS1

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_ERS1 Register (Offset = 304h) [reset = 0h]: [0]

6.4.6.5.18 PRUSS_INTC_ECR0 Register (Offset = 380h) [reset = 0h]

PRUSS_INTC_ECR0 is shown in Figure 6-159 and described in Table 6-425.

The System Interrupt Enable Clear Register0 disables system interrupts 0 to 31 to map to channels. System interrupts that are not enabled do not interrupt the host. There is a bit per system interrupt.

Table 6-424. PRUSS_INTC_ECR0 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0380h
PRU_ICSS_1_INTC	20AE 0380h

Figure 6-159. PRUSS_INTC_ECR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENABLE_CLR_31_0																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 6-425. PRUSS_INTC_ECR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ENABLE_CLR_31_0	W	0h	System interrupt enables system interrupts 0 to 31. Write a 1 in a bit position to clear that enable. Writing a 0 has no effect.

Table 6-426. Register Call Summary for PRUSS_INTC_ECR0

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_ECR0 Register (Offset = 380h) [reset = 0h]: [0]
- PRU-ICSS Interrupt Disabling: [0]

6.4.6.5.19 PRUSS_INTC_ECR1 Register (Offset = 384h) [reset = 0h]

PRUSS_INTC_ECR1 is shown in Figure 6-160 and described in Table 6-428.

The System Interrupt Enable Clear Register1 disables system interrupts 32 to 63 to map to channels. System interrupts that are not enabled do not interrupt the host. There is a bit per system interrupt.

Table 6-427. PRUSS_INTC_ECR1 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0384h
PRU_ICSS_1_INTC	20AE 0384h

Figure 6-160. PRUSS_INTC_ECR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENABLE_CLR_63_32																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 6-428. PRUSS_INTC_ECR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ENABLE_CLR_63_32	W	0h	System interrupt enables system interrupts 32 to 63. Write a 1 in a bit position to clear that enable. Writing a 0 has no effect.

Table 6-429. Register Call Summary for PRUSS_INTC_ECR1

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_ECR1 Register (Offset = 384h) [reset = 0h]: [0]
- PRU-ICSS Interrupt Disabling: [0]

6.4.6.5.20 PRUSS_INTC_CMR_0 Register (Offset = 400h) [reset = 0h]

PRUSS_INTC_CMR_0 is shown in Figure 6-161 and described in Table 6-431.

The Channel Map Register0 specify the channel for the system interrupts 0 to 3. There is one register per 4 system interrupts.

Table 6-430. PRUSS_INTC_CMR_0 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0400h
PRU_ICSS_1_INTC	20AE 0400h

Figure 6-161. PRUSS_INTC_CMR_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_3				RESERVED				CH_MAP_2			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_1				RESERVED				CH_MAP_0			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-431. PRUSS_INTC_CMR_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	CH_MAP_3	R/W	0h	Sets the channel for the system interrupt 3
23-20	RESERVED	R	0h	Reserved
19-16	CH_MAP_2	R/W	0h	Sets the channel for the system interrupt 2
15-12	RESERVED	R	0h	Reserved
11-8	CH_MAP_1	R/W	0h	Sets the channel for the system interrupt 1
7-4	RESERVED	R	0h	Reserved
3-0	CH_MAP_0	R/W	0h	Sets the channel for the system interrupt 0

Table 6-432. Register Call Summary for PRUSS_INTC_CMR_0

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CMR_0 Register (Offset = 400h) [reset = 0h]: [0]

6.4.6.5.21 PRUSS_INTC_CMR_1 Register (Offset = 404h) [reset = 0h]

PRUSS_INTC_CMR_1 is shown in Figure 6-162 and described in Table 6-434.

The Channel Map Register1 specify the channel for the system interrupts 4 to 7. There is one register per 4 system interrupts.

Table 6-433. PRUSS_INTC_CMR_1 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0404h
PRU_ICSS_1_INTC	20AE 0404h

Figure 6-162. PRUSS_INTC_CMR_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_7				RESERVED				CH_MAP_6			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_5				RESERVED				CH_MAP_4			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-434. PRUSS_INTC_CMR_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	CH_MAP_7	R/W	0h	Sets the channel for the system interrupt 7
23-20	RESERVED	R	0h	Reserved
19-16	CH_MAP_6	R/W	0h	Sets the channel for the system interrupt 6
15-12	RESERVED	R	0h	Reserved
11-8	CH_MAP_5	R/W	0h	Sets the channel for the system interrupt 5
7-4	RESERVED	R	0h	Reserved
3-0	CH_MAP_4	R/W	0h	Sets the channel for the system interrupt 4

Table 6-435. Register Call Summary for PRUSS_INTC_CMR_1

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CMR_1 Register (Offset = 404h) [reset = 0h]: [0]

6.4.6.5.22 PRUSS_INTC_CMR_2 Register (Offset = 408h) [reset = 0h]

PRUSS_INTC_CMR_2 is shown in Figure 6-163 and described in Table 6-437.

The Channel Map Register2 specify the channel for the system interrupts 8 to 11. There is one register per 4 system interrupts.

Table 6-436. PRUSS_INTC_CMR_2 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0408h
PRU_ICSS_1_INTC	20AE 0408h

Figure 6-163. PRUSS_INTC_CMR_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_11				RESERVED				CH_MAP_10			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_9				RESERVED				CH_MAP_8			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-437. PRUSS_INTC_CMR_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	CH_MAP_11	R/W	0h	Sets the channel for the system interrupt 11
23-20	RESERVED	R	0h	Reserved
19-16	CH_MAP_10	R/W	0h	Sets the channel for the system interrupt 10
15-12	RESERVED	R	0h	Reserved
11-8	CH_MAP_9	R/W	0h	Sets the channel for the system interrupt 9
7-4	RESERVED	R	0h	Reserved
3-0	CH_MAP_8	R/W	0h	Sets the channel for the system interrupt 8

Table 6-438. Register Call Summary for PRUSS_INTC_CMR_2

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CMR_2 Register (Offset = 408h) [reset = 0h]: [0]

6.4.6.5.23 PRUSS_INTC_CMR_3 Register (Offset = 40Ch) [reset = 0h]

PRUSS_INTC_CMR_3 is shown in Figure 6-164 and described in Table 6-440.

The Channel Map Register3 specify the channel for the system interrupts 12 to 15. There is one register per 4 system interrupts.

Table 6-439. PRUSS_INTC_CMR_3 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 040Ch
PRU_ICSS_1_INTC	20AE 040Ch

Figure 6-164. PRUSS_INTC_CMR_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_15				RESERVED				CH_MAP_14			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_13				RESERVED				CH_MAP_12			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-440. PRUSS_INTC_CMR_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	CH_MAP_15	R/W	0h	Sets the channel for the system interrupt 15
23-20	RESERVED	R	0h	Reserved
19-16	CH_MAP_14	R/W	0h	Sets the channel for the system interrupt 14
15-12	RESERVED	R	0h	Reserved
11-8	CH_MAP_13	R/W	0h	Sets the channel for the system interrupt 13
7-4	RESERVED	R	0h	Reserved
3-0	CH_MAP_12	R/W	0h	Sets the channel for the system interrupt 12

Table 6-441. Register Call Summary for PRUSS_INTC_CMR_3

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CMR_3 Register (Offset = 40Ch) [reset = 0h]: [0]

6.4.6.5.24 PRUSS_INTC_CMR_4 Register (Offset = 410h) [reset = 0h]

PRUSS_INTC_CMR_4 is shown in Figure 6-165 and described in Table 6-443.

The Channel Map Register4 specify the channel for the system interrupts 16 to 19. There is one register per 4 system interrupts.

Table 6-442. PRUSS_INTC_CMR_4 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0410h
PRU_ICSS_1_INTC	20AE 0410h

Figure 6-165. PRUSS_INTC_CMR_4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_19				RESERVED				CH_MAP_18			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_17				RESERVED				CH_MAP_16			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-443. PRUSS_INTC_CMR_4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	CH_MAP_19	R/W	0h	Sets the channel for the system interrupt 19
23-20	RESERVED	R	0h	Reserved
19-16	CH_MAP_18	R/W	0h	Sets the channel for the system interrupt 18
15-12	RESERVED	R	0h	Reserved
11-8	CH_MAP_17	R/W	0h	Sets the channel for the system interrupt 17
7-4	RESERVED	R	0h	Reserved
3-0	CH_MAP_16	R/W	0h	Sets the channel for the system interrupt 16

Table 6-444. Register Call Summary for PRUSS_INTC_CMR_4

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CMR_4 Register (Offset = 410h) [reset = 0h]: [0]

6.4.6.5.25 PRUSS_INTC_CMR_5 Register (Offset = 414h) [reset = 0h]

PRUSS_INTC_CMR_5 is shown in Figure 6-166 and described in Table 6-446.

The Channel Map Register5 specify the channel for the system interrupts 20 to 23. There is one register per 4 system interrupts.

Table 6-445. PRUSS_INTC_CMR_5 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0414h
PRU_ICSS_1_INTC	20AE 0414h

Figure 6-166. PRUSS_INTC_CMR_5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_23				RESERVED				CH_MAP_22			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_21				RESERVED				CH_MAP_20			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-446. PRUSS_INTC_CMR_5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	CH_MAP_23	R/W	0h	Sets the channel for the system interrupt 23
23-20	RESERVED	R	0h	Reserved
19-16	CH_MAP_22	R/W	0h	Sets the channel for the system interrupt 22
15-12	RESERVED	R	0h	Reserved
11-8	CH_MAP_21	R/W	0h	Sets the channel for the system interrupt 21
7-4	RESERVED	R	0h	Reserved
3-0	CH_MAP_20	R/W	0h	Sets the channel for the system interrupt 20

Table 6-447. Register Call Summary for PRUSS_INTC_CMR_5

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CMR_5 Register (Offset = 414h) [reset = 0h]: [0]

6.4.6.5.26 PRUSS_INTC_CMR_6 Register (Offset = 418h) [reset = 0h]

PRUSS_INTC_CMR_6 is shown in Figure 6-167 and described in Table 6-449.

The Channel Map Register6 specify the channel for the system interrupts 24 to 27. There is one register per 4 system interrupts.

Table 6-448. PRUSS_INTC_CMR_6 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0418h
PRU_ICSS_1_INTC	20AE 0418h

Figure 6-167. PRUSS_INTC_CMR_6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_27				RESERVED				CH_MAP_26			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_25				RESERVED				CH_MAP_24			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-449. PRUSS_INTC_CMR_6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	CH_MAP_27	R/W	0h	Sets the channel for the system interrupt 27
23-20	RESERVED	R	0h	Reserved
19-16	CH_MAP_26	R/W	0h	Sets the channel for the system interrupt 26
15-12	RESERVED	R	0h	Reserved
11-8	CH_MAP_25	R/W	0h	Sets the channel for the system interrupt 25
7-4	RESERVED	R	0h	Reserved
3-0	CH_MAP_24	R/W	0h	Sets the channel for the system interrupt 24

Table 6-450. Register Call Summary for PRUSS_INTC_CMR_6

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CMR_6 Register (Offset = 418h) [reset = 0h]: [0]

6.4.6.5.27 PRUSS_INTC_CMR_7 Register (Offset = 41Ch) [reset = 0h]

PRUSS_INTC_CMR_7 is shown in Figure 6-168 and described in Table 6-452.

The Channel Map Register7 specify the channel for the system interrupts 28 to 31. There is one register per 4 system interrupts.

Table 6-451. PRUSS_INTC_CMR_7 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 041Ch
PRU_ICSS_1_INTC	20AE 041Ch

Figure 6-168. PRUSS_INTC_CMR_7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_31				RESERVED				CH_MAP_30			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_29				RESERVED				CH_MAP_28			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-452. PRUSS_INTC_CMR_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	CH_MAP_31	R/W	0h	Sets the channel for the system interrupt 31
23-20	RESERVED	R	0h	Reserved
19-16	CH_MAP_30	R/W	0h	Sets the channel for the system interrupt 30
15-12	RESERVED	R	0h	Reserved
11-8	CH_MAP_29	R/W	0h	Sets the channel for the system interrupt 29
7-4	RESERVED	R	0h	Reserved
3-0	CH_MAP_28	R/W	0h	Sets the channel for the system interrupt 28

Table 6-453. Register Call Summary for PRUSS_INTC_CMR_7

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CMR_7 Register (Offset = 41Ch) [reset = 0h]: [0]

6.4.6.5.28 PRUSS_INTC_CMR_8 Register (Offset = 420h) [reset = 0h]

PRUSS_INTC_CMR_8 is shown in Figure 6-169 and described in Table 6-455.

The Channel Map Register8 specify the channel for the system interrupts 32 to 35. There is one register per 4 system interrupts.

Table 6-454. PRUSS_INTC_CMR_8 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0420h
PRU_ICSS_1_INTC	20AE 0420h

Figure 6-169. PRUSS_INTC_CMR_8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_35				RESERVED				CH_MAP_34			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_33				RESERVED				CH_MAP_32			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-455. PRUSS_INTC_CMR_8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	CH_MAP_35	R/W	0h	Sets the channel for the system interrupt 35
23-20	RESERVED	R	0h	Reserved
19-16	CH_MAP_34	R/W	0h	Sets the channel for the system interrupt 34
15-12	RESERVED	R	0h	Reserved
11-8	CH_MAP_33	R/W	0h	Sets the channel for the system interrupt 33
7-4	RESERVED	R	0h	Reserved
3-0	CH_MAP_32	R/W	0h	Sets the channel for the system interrupt 32

Table 6-456. Register Call Summary for PRUSS_INTC_CMR_8

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CMR_8 Register (Offset = 420h) [reset = 0h]: [0]

6.4.6.5.29 PRUSS_INTC_CMR_9 Register (Offset = 424h) [reset = 0h]

PRUSS_INTC_CMR_9 is shown in Figure 6-170 and described in Table 6-458.

The Channel Map Register9 specify the channel for the system interrupts 36 to 39. There is one register per 4 system interrupts.

Table 6-457. PRUSS_INTC_CMR_9 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0424h
PRU_ICSS_1_INTC	20AE 0424h

Figure 6-170. PRUSS_INTC_CMR_9 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_39				RESERVED				CH_MAP_38			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_37				RESERVED				CH_MAP_36			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-458. PRUSS_INTC_CMR_9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	CH_MAP_39	R/W	0h	Sets the channel for the system interrupt 39
23-20	RESERVED	R	0h	Reserved
19-16	CH_MAP_38	R/W	0h	Sets the channel for the system interrupt 38
15-12	RESERVED	R	0h	Reserved
11-8	CH_MAP_37	R/W	0h	Sets the channel for the system interrupt 37
7-4	RESERVED	R	0h	Reserved
3-0	CH_MAP_36	R/W	0h	Sets the channel for the system interrupt 36

Table 6-459. Register Call Summary for PRUSS_INTC_CMR_9

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CMR_9 Register (Offset = 424h) [reset = 0h]: [0]

6.4.6.5.30 PRUSS_INTC_CMR_10 Register (Offset = 428h) [reset = 0h]

PRUSS_INTC_CMR_10 is shown in Figure 6-171 and described in Table 6-461.

The Channel Map Register10 specify the channel for the system interrupts 40 to 43. There is one register per 4 system interrupts.

Table 6-460. PRUSS_INTC_CMR_10 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0428h
PRU_ICSS_1_INTC	20AE 0428h

Figure 6-171. PRUSS_INTC_CMR_10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_43				RESERVED				CH_MAP_42			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_41				RESERVED				CH_MAP_40			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-461. PRUSS_INTC_CMR_10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	CH_MAP_43	R/W	0h	Sets the channel for the system interrupt 43
23-20	RESERVED	R	0h	Reserved
19-16	CH_MAP_42	R/W	0h	Sets the channel for the system interrupt 42
15-12	RESERVED	R	0h	Reserved
11-8	CH_MAP_41	R/W	0h	Sets the channel for the system interrupt 41
7-4	RESERVED	R	0h	Reserved
3-0	CH_MAP_40	R/W	0h	Sets the channel for the system interrupt 40

Table 6-462. Register Call Summary for PRUSS_INTC_CMR_10

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CMR_10 Register (Offset = 428h) [reset = 0h]: [0]

6.4.6.5.31 PRUSS_INTC_CMR_11 Register (Offset = 42Ch) [reset = 0h]

PRUSS_INTC_CMR_11 is shown in Figure 6-172 and described in Table 6-464.

The Channel Map Register11 specify the channel for the system interrupts 44 to 47. There is one register per 4 system interrupts.

Table 6-463. PRUSS_INTC_CMR_11 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 042Ch
PRU_ICSS_1_INTC	20AE 042Ch

Figure 6-172. PRUSS_INTC_CMR_11 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_47				RESERVED				CH_MAP_46			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_45				RESERVED				CH_MAP_44			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-464. PRUSS_INTC_CMR_11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	CH_MAP_47	R/W	0h	Sets the channel for the system interrupt 47
23-20	RESERVED	R	0h	Reserved
19-16	CH_MAP_46	R/W	0h	Sets the channel for the system interrupt 46
15-12	RESERVED	R	0h	Reserved
11-8	CH_MAP_45	R/W	0h	Sets the channel for the system interrupt 45
7-4	RESERVED	R	0h	Reserved
3-0	CH_MAP_44	R/W	0h	Sets the channel for the system interrupt 44

Table 6-465. Register Call Summary for PRUSS_INTC_CMR_11

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CMR_11 Register (Offset = 42Ch) [reset = 0h]: [0]

6.4.6.5.32 PRUSS_INTC_CMR_12 Register (Offset = 430h) [reset = 0h]

PRUSS_INTC_CMR_12 is shown in Figure 6-173 and described in Table 6-467.

The Channel Map Register12 specify the channel for the system interrupts 48 to 51. There is one register per 4 system interrupts.

Table 6-466. PRUSS_INTC_CMR_12 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0430h
PRU_ICSS_1_INTC	20AE 0430h

Figure 6-173. PRUSS_INTC_CMR_12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_51				RESERVED				CH_MAP_50			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_49				RESERVED				CH_MAP_48			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-467. PRUSS_INTC_CMR_12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	CH_MAP_51	R/W	0h	Sets the channel for the system interrupt 51
23-20	RESERVED	R	0h	Reserved
19-16	CH_MAP_50	R/W	0h	Sets the channel for the system interrupt 50
15-12	RESERVED	R	0h	Reserved
11-8	CH_MAP_49	R/W	0h	Sets the channel for the system interrupt 49
7-4	RESERVED	R	0h	Reserved
3-0	CH_MAP_48	R/W	0h	Sets the channel for the system interrupt 48

Table 6-468. Register Call Summary for PRUSS_INTC_CMR_12

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CMR_12 Register (Offset = 430h) [reset = 0h]: [0]

6.4.6.5.33 PRUSS_INTC_CMR_13 Register (Offset = 434h) [reset = 0h]

PRUSS_INTC_CMR_13 is shown in Figure 6-174 and described in Table 6-470.

The Channel Map Register13 specify the channel for the system interrupts 52 to 55. There is one register per 4 system interrupts.

Table 6-469. PRUSS_INTC_CMR_13 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0434h
PRU_ICSS_1_INTC	20AE 0434h

Figure 6-174. PRUSS_INTC_CMR_13 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_55				RESERVED				CH_MAP_54			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_53				RESERVED				CH_MAP_52			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-470. PRUSS_INTC_CMR_13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	CH_MAP_55	R/W	0h	Sets the channel for the system interrupt 55
23-20	RESERVED	R	0h	Reserved
19-16	CH_MAP_54	R/W	0h	Sets the channel for the system interrupt 54
15-12	RESERVED	R	0h	Reserved
11-8	CH_MAP_53	R/W	0h	Sets the channel for the system interrupt 53
7-4	RESERVED	R	0h	Reserved
3-0	CH_MAP_52	R/W	0h	Sets the channel for the system interrupt 52

Table 6-471. Register Call Summary for PRUSS_INTC_CMR_13

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CMR_13 Register (Offset = 434h) [reset = 0h]: [0]

6.4.6.5.34 PRUSS_INTC_CMR_14 Register (Offset = 438h) [reset = 0h]

PRUSS_INTC_CMR_14 is shown in Figure 6-175 and described in Table 6-473.

The Channel Map Register14 specify the channel for the system interrupts 56 to 59. There is one register per 4 system interrupts.

Table 6-472. PRUSS_INTC_CMR_14 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0438h
PRU_ICSS_1_INTC	20AE 0438h

Figure 6-175. PRUSS_INTC_CMR_14 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_59				RESERVED				CH_MAP_58			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_57				RESERVED				CH_MAP_56			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-473. PRUSS_INTC_CMR_14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	CH_MAP_59	R/W	0h	Sets the channel for the system interrupt 59
23-20	RESERVED	R	0h	Reserved
19-16	CH_MAP_58	R/W	0h	Sets the channel for the system interrupt 58
15-12	RESERVED	R	0h	Reserved
11-8	CH_MAP_57	R/W	0h	Sets the channel for the system interrupt 57
7-4	RESERVED	R	0h	Reserved
3-0	CH_MAP_56	R/W	0h	Sets the channel for the system interrupt 56

Table 6-474. Register Call Summary for PRUSS_INTC_CMR_14

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CMR_14 Register (Offset = 438h) [reset = 0h]: [0]

6.4.6.5.35 PRUSS_INTC_CMR_15 Register (Offset = 43Ch) [reset = 0h]

PRUSS_INTC_CMR_15 is shown in Figure 6-176 and described in Table 6-476.

The Channel Map Register15 specify the channel for the system interrupts 60 to 63. There is one register per 4 system interrupts.

Table 6-475. PRUSS_INTC_CMR_15 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 043Ch
PRU_ICSS_1_INTC	20AE 043Ch

Figure 6-176. PRUSS_INTC_CMR_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_63				RESERVED				CH_MAP_62			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_61				RESERVED				CH_MAP_60			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-476. PRUSS_INTC_CMR_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	CH_MAP_63	R/W	0h	Sets the channel for the system interrupt 63
23-20	RESERVED	R	0h	Reserved
19-16	CH_MAP_62	R/W	0h	Sets the channel for the system interrupt 62
15-12	RESERVED	R	0h	Reserved
11-8	CH_MAP_61	R/W	0h	Sets the channel for the system interrupt 61
7-4	RESERVED	R	0h	Reserved
3-0	CH_MAP_60	R/W	0h	Sets the channel for the system interrupt 60

Table 6-477. Register Call Summary for PRUSS_INTC_CMR_15

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_CMR_15 Register (Offset = 43Ch) [reset = 0h]: [0]

6.4.6.5.36 PRUSS_INTC_HMR0 Register (Offset = 800h) [reset = 0h]

PRUSS_INTC_HMR0 is shown in Figure 6-177 and described in Table 6-479.

The Host Interrupt Map Register0 define the host interrupt for channels 0 to 3. There is one register per 4 channels. Channels with forced host interrupt mappings will have their fields read-only.

Table 6-478. PRUSS_INTC_HMR0 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0800h
PRU_ICSS_1_INTC	20AE 0800h

Figure 6-177. PRUSS_INTC_HMR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				HINT_MAP_3				RESERVED				HINT_MAP_2			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				HINT_MAP_1				RESERVED				HINT_MAP_0			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-479. PRUSS_INTC_HMR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	HINT_MAP_3	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 3
23-20	RESERVED	R	0h	Reserved
19-16	HINT_MAP_2	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 2
15-12	RESERVED	R	0h	Reserved
11-8	HINT_MAP_1	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 1
7-4	RESERVED	R	0h	Reserved
3-0	HINT_MAP_0	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 0

Table 6-480. Register Call Summary for PRUSS_INTC_HMR0

PRU-ICSS Local Interrupt Controller

- PRU-ICSS Interrupt Controller Registers: [0]
- PRU-ICSS Interrupt Channel Mapping: [0]
- PRUSS_INTC_HMR0 Register (Offset = 800h) [reset = 0h]: [0]

6.4.6.5.37 PRUSS_INTC_HMR1 Register (Offset = 804h) [reset = 0h]

PRUSS_INTC_HMR1 is shown in Figure 6-178 and described in Table 6-482.

The Host Interrupt Map Register1 define the host interrupt for channels 4 to 7. There is one register per 4 channels. Channels with forced host interrupt mappings will have their fields read-only.

Table 6-481. PRUSS_INTC_HMR1 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0804h
PRU_ICSS_1_INTC	20AE 0804h

Figure 6-178. PRUSS_INTC_HMR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				HINT_MAP_7				RESERVED				HINT_MAP_6			
R-0h				R/W-0h				R-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				HINT_MAP_5				RESERVED				HINT_MAP_4			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-482. PRUSS_INTC_HMR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	HINT_MAP_7	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 7
23-20	RESERVED	R	0h	Reserved
19-16	HINT_MAP_6	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 6
15-12	RESERVED	R	0h	Reserved
11-8	HINT_MAP_5	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 5
7-4	RESERVED	R	0h	Reserved
3-0	HINT_MAP_4	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 4

Table 6-483. Register Call Summary for PRUSS_INTC_HMR1

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HMR1 Register (Offset = 804h) [reset = 0h]: [0]

6.4.6.5.38 PRUSS_INTC_HMR2 Register (Offset = 808h) [reset = 0h]

PRUSS_INTC_HMR2 is shown in Figure 6-179 and described in Table 6-485.

The Host Interrupt Map Register2 define the host interrupt for channels 8 to 9. There is one register per 4 channels. Channels with forced host interrupt mappings will have their fields read-only.

Table 6-484. PRUSS_INTC_HMR2 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0808h
PRU_ICSS_1_INTC	20AE 0808h

Figure 6-179. PRUSS_INTC_HMR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				HINT_MAP_9				RESERVED				HINT_MAP_8			
R-0h				R/W-0h				R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-485. PRUSS_INTC_HMR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11-8	HINT_MAP_9	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 9
7-4	RESERVED	R	0h	Reserved
3-0	HINT_MAP_8	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 8

Table 6-486. Register Call Summary for PRUSS_INTC_HMR2

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRU-ICSS Interrupt Channel Mapping: [0]
- PRUSS_INTC_HMR2 Register (Offset = 808h) [reset = 0h]: [0]

6.4.6.5.39 PRUSS_INTC_HIPIR_0 Register (Offset = 900h) [reset = 80000000h]

PRUSS_INTC_HIPIR_0 is shown in Figure 6-180 and described in Table 6-488.

The Host Interrupt Prioritized Index Register_j (where j=0 to 9) shows the highest priority current pending interrupt for the host interrupt j. There is one register per host interrupt.

Table 6-487. PRUSS_INTC_HIPIR_0 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0900h
PRU_ICSS_1_INTC	20AE 0900h

Figure 6-180. PRUSS_INTC_HIPIR_0 Register

31	30	29	28	27	26	25	24
NONE_HINT	RESERVED						
R-1h	R-0000 00h						
23	22	21	20	19	18	17	16
RESERVED							
R-0000 00h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT	
R-0000 00h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 6-488. PRUSS_INTC_HIPIR_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT	R	1h	No pending interrupt.
30-10	RESERVED	R	0000 00h	Reserved
9-0	PRI_HINT	R	0h	HOST INT j PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

Table 6-489. Register Call Summary for PRUSS_INTC_HIPIR_0

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HIPIR_0 Register (Offset = 900h) [reset = 80000000h]: [0]

6.4.6.5.40 PRUSS_INTC_HIPIR_1 Register (Offset = 904h) [reset = 80000000h]

PRUSS_INTC_HIPIR_1 is shown in Figure 6-181 and described in Table 6-491.

The Host Interrupt Prioritized Index Register_j (where j=0 to 9) shows the highest priority current pending interrupt for the host interrupt j. There is one register per host interrupt.

Table 6-490. PRUSS_INTC_HIPIR_1 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0904h
PRU_ICSS_1_INTC	20AE 0904h

Figure 6-181. PRUSS_INTC_HIPIR_1 Register

31	30	29	28	27	26	25	24
NONE_HINT	RESERVED						
R-1h	R-0000 00h						
23	22	21	20	19	18	17	16
RESERVED							
R-0000 00h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT	
R-0000 00h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 6-491. PRUSS_INTC_HIPIR_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT	R	1h	No pending interrupt.
30-10	RESERVED	R	0000 00h	Reserved
9-0	PRI_HINT	R	0h	HOST INT j PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

Table 6-492. Register Call Summary for PRUSS_INTC_HIPIR_1

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HIPIR_1 Register (Offset = 904h) [reset = 80000000h]: [0]

6.4.6.5.41 PRUSS_INTC_HIPIR_2 Register (Offset = 908h) [reset = 80000000h]

PRUSS_INTC_HIPIR_2 is shown in Figure 6-182 and described in Table 6-494.

The Host Interrupt Prioritized Index Register_j (where j=0 to 9) shows the highest priority current pending interrupt for the host interrupt j. There is one register per host interrupt.

Table 6-493. PRUSS_INTC_HIPIR_2 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0908h
PRU_ICSS_1_INTC	20AE 0908h

Figure 6-182. PRUSS_INTC_HIPIR_2 Register

31	30	29	28	27	26	25	24
NONE_HINT	RESERVED						
R-1h	R-0000 00h						
23	22	21	20	19	18	17	16
RESERVED							
R-0000 00h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT	
R-0000 00h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 6-494. PRUSS_INTC_HIPIR_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT	R	1h	No pending interrupt.
30-10	RESERVED	R	0000 00h	Reserved
9-0	PRI_HINT	R	0h	HOST INT j PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

Table 6-495. Register Call Summary for PRUSS_INTC_HIPIR_2

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HIPIR_2 Register (Offset = 908h) [reset = 80000000h]: [0]

6.4.6.5.42 PRUSS_INTC_HIPIR_3 Register (Offset = 90Ch) [reset = 80000000h]

PRUSS_INTC_HIPIR_3 is shown in Figure 6-183 and described in Table 6-497.

The Host Interrupt Prioritized Index Register_j (where j=0 to 9) shows the highest priority current pending interrupt for the host interrupt j. There is one register per host interrupt.

Table 6-496. PRUSS_INTC_HIPIR_3 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 090Ch
PRU_ICSS_1_INTC	20AE 090Ch

Figure 6-183. PRUSS_INTC_HIPIR_3 Register

31	30	29	28	27	26	25	24
NONE_HINT	RESERVED						
R-1h	R-0000 00h						
23	22	21	20	19	18	17	16
RESERVED							
R-0000 00h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT	
R-0000 00h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 6-497. PRUSS_INTC_HIPIR_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT	R	1h	No pending interrupt.
30-10	RESERVED	R	0000 00h	Reserved
9-0	PRI_HINT	R	0h	HOST INT j PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

Table 6-498. Register Call Summary for PRUSS_INTC_HIPIR_3

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HIPIR_3 Register (Offset = 90Ch) [reset = 80000000h]: [0]

6.4.6.5.43 PRUSS_INTC_HIPIR_4 Register (Offset = 910h) [reset = 80000000h]

PRUSS_INTC_HIPIR_4 is shown in Figure 6-184 and described in Table 6-500.

The Host Interrupt Prioritized Index Register_j (where j=0 to 9) shows the highest priority current pending interrupt for the host interrupt j. There is one register per host interrupt.

Table 6-499. PRUSS_INTC_HIPIR_4 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0910h
PRU_ICSS_1_INTC	20AE 0910h

Figure 6-184. PRUSS_INTC_HIPIR_4 Register

31	30	29	28	27	26	25	24
NONE_HINT	RESERVED						
R-1h	R-0000 00h						
23	22	21	20	19	18	17	16
RESERVED							
R-0000 00h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT	
R-0000 00h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 6-500. PRUSS_INTC_HIPIR_4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT	R	1h	No pending interrupt.
30-10	RESERVED	R	0000 00h	Reserved
9-0	PRI_HINT	R	0h	HOST INT j PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

Table 6-501. Register Call Summary for PRUSS_INTC_HIPIR_4

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HIPIR_4 Register (Offset = 910h) [reset = 80000000h]: [0]

6.4.6.5.44 PRUSS_INTC_HIPIR_5 Register (Offset = 914h) [reset = 80000000h]

PRUSS_INTC_HIPIR_5 is shown in Figure 6-185 and described in Table 6-503.

The Host Interrupt Prioritized Index Register_j (where j=0 to 9) shows the highest priority current pending interrupt for the host interrupt j. There is one register per host interrupt.

Table 6-502. PRUSS_INTC_HIPIR_5 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0914h
PRU_ICSS_1_INTC	20AE 0914h

Figure 6-185. PRUSS_INTC_HIPIR_5 Register

31	30	29	28	27	26	25	24
NONE_HINT	RESERVED						
R-1h	R-0000 00h						
23	22	21	20	19	18	17	16
RESERVED							
R-0000 00h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT	
R-0000 00h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 6-503. PRUSS_INTC_HIPIR_5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT	R	1h	No pending interrupt.
30-10	RESERVED	R	0000 00h	Reserved
9-0	PRI_HINT	R	0h	HOST INT j PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

Table 6-504. Register Call Summary for PRUSS_INTC_HIPIR_5

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HIPIR_5 Register (Offset = 914h) [reset = 80000000h]: [0]

6.4.6.5.45 PRUSS_INTC_HIPIR_6 Register (Offset = 918h) [reset = 80000000h]

PRUSS_INTC_HIPIR_6 is shown in Figure 6-186 and described in Table 6-506.

The Host Interrupt Prioritized Index Register_j (where j=0 to 9) shows the highest priority current pending interrupt for the host interrupt j. There is one register per host interrupt.

Table 6-505. PRUSS_INTC_HIPIR_6 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0918h
PRU_ICSS_1_INTC	20AE 0918h

Figure 6-186. PRUSS_INTC_HIPIR_6 Register

31	30	29	28	27	26	25	24
NONE_HINT	RESERVED						
R-1h	R-0000 00h						
23	22	21	20	19	18	17	16
RESERVED							
R-0000 00h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT	
R-0000 00h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 6-506. PRUSS_INTC_HIPIR_6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT	R	1h	No pending interrupt.
30-10	RESERVED	R	0000 00h	Reserved
9-0	PRI_HINT	R	0h	HOST INT j PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

Table 6-507. Register Call Summary for PRUSS_INTC_HIPIR_6

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HIPIR_6 Register (Offset = 918h) [reset = 80000000h]: [0]

6.4.6.5.46 PRUSS_INTC_HIPIR_7 Register (Offset = 91Ch) [reset = 80000000h]

PRUSS_INTC_HIPIR_7 is shown in Figure 6-187 and described in Table 6-509.

The Host Interrupt Prioritized Index Register_j (where j=0 to 9) shows the highest priority current pending interrupt for the host interrupt j. There is one register per host interrupt.

Table 6-508. PRUSS_INTC_HIPIR_7 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 091Ch
PRU_ICSS_1_INTC	20AE 091Ch

Figure 6-187. PRUSS_INTC_HIPIR_7 Register

31	30	29	28	27	26	25	24
NONE_HINT	RESERVED						
R-1h	R-0000 00h						
23	22	21	20	19	18	17	16
RESERVED							
R-0000 00h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT	
R-0000 00h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 6-509. PRUSS_INTC_HIPIR_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT	R	1h	No pending interrupt.
30-10	RESERVED	R	0000 00h	Reserved
9-0	PRI_HINT	R	0h	HOST INT j PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

Table 6-510. Register Call Summary for PRUSS_INTC_HIPIR_7

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HIPIR_7 Register (Offset = 91Ch) [reset = 80000000h]: [0]

6.4.6.5.47 PRUSS_INTC_HIPIR_8 Register (Offset = 920h) [reset = 80000000h]

PRUSS_INTC_HIPIR_8 is shown in Figure 6-188 and described in Table 6-512.

The Host Interrupt Prioritized Index Register_j (where j=0 to 9) shows the highest priority current pending interrupt for the host interrupt j. There is one register per host interrupt.

Table 6-511. PRUSS_INTC_HIPIR_8 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0920h
PRU_ICSS_1_INTC	20AE 0920h

Figure 6-188. PRUSS_INTC_HIPIR_8 Register

31	30	29	28	27	26	25	24
NONE_HINT	RESERVED						
R-1h	R-0000 00h						
23	22	21	20	19	18	17	16
RESERVED							
R-0000 00h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT	
R-0000 00h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 6-512. PRUSS_INTC_HIPIR_8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT	R	1h	No pending interrupt.
30-10	RESERVED	R	0000 00h	Reserved
9-0	PRI_HINT	R	0h	HOST INT j PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

Table 6-513. Register Call Summary for PRUSS_INTC_HIPIR_8

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HIPIR_8 Register (Offset = 920h) [reset = 80000000h]: [0]

6.4.6.5.48 PRUSS_INTC_HIPIR_9 Register (Offset = 924h) [reset = 80000000h]

PRUSS_INTC_HIPIR_9 is shown in Figure 6-189 and described in Table 6-515.

The Host Interrupt Prioritized Index Register_j (where j=0 to 9) shows the highest priority current pending interrupt for the host interrupt j. There is one register per host interrupt.

Table 6-514. PRUSS_INTC_HIPIR_9 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0924h
PRU_ICSS_1_INTC	20AE 0924h

Figure 6-189. PRUSS_INTC_HIPIR_9 Register

31	30	29	28	27	26	25	24
NONE_HINT	RESERVED						
R-1h	R-0000 00h						
23	22	21	20	19	18	17	16
RESERVED							
R-0000 00h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT	
R-0000 00h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 6-515. PRUSS_INTC_HIPIR_9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT	R	1h	No pending interrupt.
30-10	RESERVED	R	0000 00h	Reserved
9-0	PRI_HINT	R	0h	HOST INT j PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

Table 6-516. Register Call Summary for PRUSS_INTC_HIPIR_9

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HIPIR_9 Register (Offset = 924h) [reset = 80000000h]: [0]

6.4.6.5.49 PRUSS_INTC_SIPR0 Register (Offset = D00h) [reset = 1h]

PRUSS_INTC_SIPR0 is shown in Figure 6-190 and described in Table 6-518.

The System Interrupt Polarity Register0 define the polarity of the system interrupts 0 to 31. There is a polarity for each system interrupt. The polarity of all system interrupts is active high; always write 1 to the bits of this register.

Table 6-517. PRUSS_INTC_SIPR0 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0D00h
PRU_ICSS_1_INTC	20AE 0D00h

Figure 6-190. PRUSS_INTC_SIPR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POLARITY_31_0																															
R/W-1h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-518. PRUSS_INTC_SIPR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	POLARITY_31_0	R/W	1h	Interrupt polarity of the system interrupts 0 to 31. 0h: Active low. 1h: Active high.

Table 6-519. Register Call Summary for PRUSS_INTC_SIPR0

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Basic Programming Model: [0]
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_SIPR0 Register (Offset = D00h) [reset = 1h]: [0]

6.4.6.5.50 PRUSS_INTC_SIPR1 Register (Offset = D04h) [reset = 1h]

PRUSS_INTC_SIPR1 is shown in Figure 6-191 and described in Table 6-521.

The System Interrupt Polarity Register1 define the polarity of the system interrupts 32 to 63. There is a polarity for each system interrupt. The polarity of all system interrupts is active high; always write 1 to the bits of this register.

Table 6-520. PRUSS_INTC_SIPR1 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0D04h
PRU_ICSS_1_INTC	20AE 0D04h

Figure 6-191. PRUSS_INTC_SIPR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POLARITY_63_32																															
R/W-1h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-521. PRUSS_INTC_SIPR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	POLARITY_63_32	R/W	1h	Interrupt polarity of the system interrupts 32 to 63. 0h: Active low. 1h: Active high.

Table 6-522. Register Call Summary for PRUSS_INTC_SIPR1

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Basic Programming Model: [0]
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_SIPR1 Register (Offset = D04h) [reset = 1h]: [0]

6.4.6.5.51 PRUSS_INTC_SITR0 Register (Offset = D80h) [reset = 0h]

PRUSS_INTC_SITR0 is shown in Figure 6-192 and described in Table 6-524.

The System Interrupt Type Register0 define the type of the system interrupts 0 to 31. There is a type for each system interrupt. The type of all system interrupts is pulse; always write 0 to the bits of this register.

Table 6-523. PRUSS_INTC_SITR0 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0D80h
PRU_ICSS_1_INTC	20AE 0D80h

Figure 6-192. PRUSS_INTC_SITR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TYPE_31_0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-524. PRUSS_INTC_SITR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TYPE_31_0	R/W	0h	Interrupt type of the system interrupts 0 to 31. 0h: Level or pulse interrupt. 1h: Edge interrupt (required edge detect).

Table 6-525. Register Call Summary for PRUSS_INTC_SITR0

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Basic Programming Model: [0]
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_SITR0 Register (Offset = D80h) [reset = 0h]: [0]

6.4.6.5.52 PRUSS_INTC_SITR1 Register (Offset = D84h) [reset = 0h]

PRUSS_INTC_SITR1 is shown in Figure 6-193 and described in Table 6-527.

The System Interrupt Type Register1 define the type of the system interrupts 32 to 63. There is a type for each system interrupt. The type of all system interrupts is pulse; always write 0 to the bits of this register.

Table 6-526. PRUSS_INTC_SITR1 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 0D84h
PRU_ICSS_1_INTC	20AE 0D84h

Figure 6-193. PRUSS_INTC_SITR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TYPE_63_32																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-527. PRUSS_INTC_SITR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TYPE_63_32	R/W	0h	Interrupt type of the system interrupts 32 to 63. 0h Level or pulse interrupt. 1h: Edge interrupt (required edge detect).

Table 6-528. Register Call Summary for PRUSS_INTC_SITR1

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Basic Programming Model: [0]
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_SITR1 Register (Offset = D84h) [reset = 0h]: [0]

6.4.6.5.53 PRUSS_INTC_HINLR_0 Register (Offset = 1100h) [reset = 100h]

PRUSS_INTC_HINLR_0 is shown in Figure 6-194 and described in Table 6-530.

The Host Interrupt Nesting Level Register_j (where j=0 to 9) display and control the nesting level for host interrupt j. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Table 6-529. PRUSS_INTC_HINLR_0 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 1100h
PRU_ICSS_1_INTC	20AE 1100h

Figure 6-194. PRUSS_INTC_HINLR_0 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT
R-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT							
R/W-100h							

LEGEND: R = Read Only; R/W = Read/Write; W = Write Only; -*n* = value after reset

Table 6-530. PRUSS_INTC_HINLR_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R	0h	Reserved
8-0	NEST_HINT	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

Table 6-531. Register Call Summary for PRUSS_INTC_HINLR_0

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HINLR_0 Register (Offset = 1100h) [reset = 100h]: [0]

6.4.6.5.54 PRUSS_INTC_HINLR_1 Register (Offset = 1104h) [reset = 100h]

PRUSS_INTC_HINLR_1 is shown in Figure 6-195 and described in Table 6-533.

The Host Interrupt Nesting Level Register_j (where j=0 to 9) display and control the nesting level for host interrupt j. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Table 6-532. PRUSS_INTC_HINLR_1 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 1104h
PRU_ICSS_1_INTC	20AE 1104h

Figure 6-195. PRUSS_INTC_HINLR_1 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT
R-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT							
R/W-100h							

LEGEND: R = Read Only; R/W = Read/Write; W = Write Only; -*n* = value after reset

Table 6-533. PRUSS_INTC_HINLR_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R	0h	Reserved
8-0	NEST_HINT	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

Table 6-534. Register Call Summary for PRUSS_INTC_HINLR_1

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HINLR_1 Register (Offset = 1104h) [reset = 100h]: [0]

6.4.6.5.55 PRUSS_INTC_HINLR_2 Register (Offset = 1108h) [reset = 100h]

PRUSS_INTC_HINLR_2 is shown in Figure 6-196 and described in Table 6-536.

The Host Interrupt Nesting Level Register_j (where j=0 to 9) display and control the nesting level for host interrupt j. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Table 6-535. PRUSS_INTC_HINLR_2 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 1108h
PRU_ICSS_1_INTC	20AE 1108h

Figure 6-196. PRUSS_INTC_HINLR_2 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT
R-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT							
R/W-100h							

LEGEND: R = Read Only; R/W = Read/Write; W = Write Only; -*n* = value after reset

Table 6-536. PRUSS_INTC_HINLR_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R	0h	Reserved
8-0	NEST_HINT	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

Table 6-537. Register Call Summary for PRUSS_INTC_HINLR_2

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HINLR_2 Register (Offset = 1108h) [reset = 100h]: [0]

6.4.6.5.56 PRUSS_INTC_HINLR_3 Register (Offset = 110Ch) [reset = 100h]

PRUSS_INTC_HINLR_3 is shown in Figure 6-197 and described in Table 6-539.

The Host Interrupt Nesting Level Register_j (where j=0 to 9) display and control the nesting level for host interrupt j. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Table 6-538. PRUSS_INTC_HINLR_3 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 110Ch
PRU_ICSS_1_INTC	20AE 110Ch

Figure 6-197. PRUSS_INTC_HINLR_3 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT
R-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT							
R/W-100h							

LEGEND: R = Read Only; R/W = Read/Write; W = Write Only; -*n* = value after reset

Table 6-539. PRUSS_INTC_HINLR_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R	0h	Reserved
8-0	NEST_HINT	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

Table 6-540. Register Call Summary for PRUSS_INTC_HINLR_3

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HINLR_3 Register (Offset = 110Ch) [reset = 100h]: [0]

6.4.6.5.57 PRUSS_INTC_HINLR_4 Register (Offset = 1110h) [reset = 100h]

PRUSS_INTC_HINLR_4 is shown in Figure 6-198 and described in Table 6-542.

The Host Interrupt Nesting Level Register_j (where j=0 to 9) display and control the nesting level for host interrupt j. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Table 6-541. PRUSS_INTC_HINLR_4 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 1110h
PRU_ICSS_1_INTC	20AE 1110h

Figure 6-198. PRUSS_INTC_HINLR_4 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT
R-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT							
R/W-100h							

LEGEND: R = Read Only; R/W = Read/Write; W = Write Only; -*n* = value after reset

Table 6-542. PRUSS_INTC_HINLR_4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R	0h	Reserved
8-0	NEST_HINT	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

Table 6-543. Register Call Summary for PRUSS_INTC_HINLR_4

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HINLR_4 Register (Offset = 1110h) [reset = 100h]: [0]

6.4.6.5.58 PRUSS_INTC_HINLR_5 Register (Offset = 1114h) [reset = 100h]

PRUSS_INTC_HINLR_5 is shown in Figure 6-199 and described in Table 6-545.

The Host Interrupt Nesting Level Register_j (where j=0 to 9) display and control the nesting level for host interrupt j. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Table 6-544. PRUSS_INTC_HINLR_5 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 1114h
PRU_ICSS_1_INTC	20AE 1114h

Figure 6-199. PRUSS_INTC_HINLR_5 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT
R-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT							
R/W-100h							

LEGEND: R = Read Only; R/W = Read/Write; W = Write Only; -*n* = value after reset

Table 6-545. PRUSS_INTC_HINLR_5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R	0h	Reserved
8-0	NEST_HINT	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

Table 6-546. Register Call Summary for PRUSS_INTC_HINLR_5

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HINLR_5 Register (Offset = 1114h) [reset = 100h]: [0]

6.4.6.5.59 PRUSS_INTC_HINLR_6 Register (Offset = 1118h) [reset = 100h]

PRUSS_INTC_HINLR_6 is shown in Figure 6-200 and described in Table 6-548.

The Host Interrupt Nesting Level Register_j (where j=0 to 9) display and control the nesting level for host interrupt j. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Table 6-547. PRUSS_INTC_HINLR_6 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 1118h
PRU_ICSS_1_INTC	20AE 1118h

Figure 6-200. PRUSS_INTC_HINLR_6 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT
R-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT							
R/W-100h							

LEGEND: R = Read Only; R/W = Read/Write; W = Write Only; -*n* = value after reset

Table 6-548. PRUSS_INTC_HINLR_6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R	0h	Reserved
8-0	NEST_HINT	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

Table 6-549. Register Call Summary for PRUSS_INTC_HINLR_6

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HINLR_6 Register (Offset = 1118h) [reset = 100h]: [0]

6.4.6.5.60 PRUSS_INTC_HINLR_7 Register (Offset = 111Ch) [reset = 100h]

PRUSS_INTC_HINLR_7 is shown in Figure 6-201 and described in Table 6-551.

The Host Interrupt Nesting Level Register_j (where j=0 to 9) display and control the nesting level for host interrupt j. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Table 6-550. PRUSS_INTC_HINLR_7 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 111Ch
PRU_ICSS_1_INTC	20AE 111Ch

Figure 6-201. PRUSS_INTC_HINLR_7 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT
R-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT							
R/W-100h							

LEGEND: R = Read Only; R/W = Read/Write; W = Write Only; -*n* = value after reset

Table 6-551. PRUSS_INTC_HINLR_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R	0h	Reserved
8-0	NEST_HINT	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

Table 6-552. Register Call Summary for PRUSS_INTC_HINLR_7

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HINLR_7 Register (Offset = 111Ch) [reset = 100h]: [0]

6.4.6.5.61 PRUSS_INTC_HINLR_8 Register (Offset = 1120h) [reset = 100h]

PRUSS_INTC_HINLR_8 is shown in Figure 6-202 and described in Table 6-554.

The Host Interrupt Nesting Level Register_j (where j=0 to 9) display and control the nesting level for host interrupt j. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Table 6-553. PRUSS_INTC_HINLR_8 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 1120h
PRU_ICSS_1_INTC	20AE 1120h

Figure 6-202. PRUSS_INTC_HINLR_8 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT
R-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT							
R/W-100h							

LEGEND: R = Read Only; R/W = Read/Write; W = Write Only; -*n* = value after reset

Table 6-554. PRUSS_INTC_HINLR_8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R	0h	Reserved
8-0	NEST_HINT	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

Table 6-555. Register Call Summary for PRUSS_INTC_HINLR_8

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HINLR_8 Register (Offset = 1120h) [reset = 100h]: [0]

6.4.6.5.62 PRUSS_INTC_HINLR_9 Register (Offset = 1124h) [reset = 100h]

PRUSS_INTC_HINLR_9 is shown in Figure 6-203 and described in Table 6-557.

The Host Interrupt Nesting Level Register_j (where j=0 to 9) display and control the nesting level for host interrupt j. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Table 6-556. PRUSS_INTC_HINLR_9 Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 1124h
PRU_ICSS_1_INTC	20AE 1124h

Figure 6-203. PRUSS_INTC_HINLR_9 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT
R-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT							
R/W-100h							

LEGEND: R = Read Only; R/W = Read/Write; W = Write Only; -*n* = value after reset

Table 6-557. PRUSS_INTC_HINLR_9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R	0h	Reserved
8-0	NEST_HINT	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

Table 6-558. Register Call Summary for PRUSS_INTC_HINLR_9

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HINLR_9 Register (Offset = 1124h) [reset = 100h]: [0]

6.4.6.5.63 PRUSS_INTC_HIER Register (Offset = 1500h) [reset = 0h]

PRUSS_INTC_HIER is shown in Figure 6-204 and described in Table 6-560.

The Host Interrupt Enable Registers enable or disable individual host interrupts. These work separately from the global enables. There is one bit per host interrupt. These bits are updated when writing to the Host Interrupt Enable Index Set and Host Interrupt Enable Index Clear registers.

Table 6-559. PRUSS_INTC_HIER Instances

Instance	Physical Address
PRU_ICSS_0_INTC	20AA 1500h
PRU_ICSS_1_INTC	20AE 1500h

Figure 6-204. PRUSS_INTC_HIER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0000 00h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						ENABLE_HINT									
R-0000 00h						R/W-0h									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-560. PRUSS_INTC_HIER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0000 00h	Reserved
9-0	ENABLE_HINT	R/W	0h	The enable of the host interrupts (one per bit). 0h: Disabled 1h: Enabled

Table 6-561. Register Call Summary for PRUSS_INTC_HIER

PRU-ICSS Local Interrupt Controller
- PRU-ICSS Interrupt Controller Registers: [0]
- PRUSS_INTC_HIER Register (Offset = 1500h) [reset = 0h]: [0]

6.4.7 PRU-ICSS UART Module

This section describes an Universal Asynchronous Receive and Transmit (UART) module which is part of the device integrated PRU-ICSS_0 and PRU-ICSS_1 - PRU-ICSS_0_UART0 and PRU-ICSS_1_UART0, respectively.

6.4.7.1 PRU-ICSS UART Module Overview

6.4.7.1.1 Purpose of the PRU-ICSS integrated UART Peripheral

The PRU-ICSS_UART0 peripheral is based on the industry standard TL16C550 asynchronous communications element, which in turn is a functional upgrade of the TL16C450. Functionally similar to the TL16C450 on power up (single character or TL16C450 mode), the PRU-ICSS_UART0 can be placed in an alternate FIFO (TL16C550) mode. This relieves the CPU of excessive software overhead by buffering received and transmitted characters. The receiver and transmitter FIFOs store up to 16 bytes including three additional bits of error status per byte for the receiver FIFO.

The PRU-ICSS_UART0 performs serial-to-parallel conversions on data received from a peripheral device and parallel-to-serial conversion on data received from the CPU. The CPU can read the PRU-ICSS_UART0 status at any time. The PRU-ICSS_UART0 includes control capability and a processor interrupt system that can be tailored to minimize software management of the communications link.

The PRU-ICSS_UART0 includes a programmable baud generator capable of dividing the PRU-ICSS_UART0 input clock by divisors from 1 to 65535 and producing a 16× reference clock or a 13× reference clock for the internal transmitter and receiver logic.

6.4.7.1.2 PRU-ICSS UART Key Features

6.4.7.1.2.1 PRU-ICSS UART Module Industry Standard Compliance Statement

The PRU-ICSS_UART0 peripheral is based on the industry standard TL16C550 asynchronous communications element, which is a functional upgrade of the TL16C450. The information in this chapter assumes that user is familiar with these standards.

6.4.7.2 PRU-ICSS UART Environment

This section describes the PRU-ICSS_UART0 module interface to the device environment

6.4.7.2.1 PRU-ICSS UART Pin Multiplexing

Extensive pin multiplexing is used to accommodate the largest number of peripheral functions in the smallest possible package. Pin multiplexing is controlled using a combination of hardware configuration at device reset and software programmable register settings. For more information on the PRU-ICSS_UART0 pin multiplexing, refer to the Section 5.1.3.1.1, *Pad Configuration Registers* in the chapter, *BOOT_CFG*.

6.4.7.2.2 PRU-ICSS UART Signal Descriptions

The PRU-ICSS_UART0 utilize a minimal number of signal connections to interface with external devices. The PRU-ICSS_UART0 signal descriptions are included in Table 6-562.

Table 6-562. PRUSS_UART0 Signal Descriptions

Signal Name	Signal Type	Function
UART0_TXD	Output	Serial data transmit
UART0_RXD	Input	Serial data receive
$\overline{\text{UART0\_CTS}}$	Input	Clear-to-Send handshaking signal
$\overline{\text{UART0\_RTS}}$	Output	Request-to-Send handshaking signal

6.4.7.2.3 PRU-ICSS UART Data Format and Protocol Description

6.4.7.2.3.1 PRU-ICSS UART Transmission Protocol

The PRU-ICSS_UART0 transmitter section includes a transmitter hold register (THR), memory mapped in the register PRUSS_UART_RBR_THR_REGISTERS[7:0] DATA bitfield and a transmitter shift register (TSR), which is NOT memory mapped. When the PRU-ICSS_UART0 is in the FIFO mode, THR is a 16-byte FIFO. Transmitter section control is a function of the PRU-ICSS_UART0 line control register PRUSS_UART_LINE_CONTROL_REGISTER. Based on the settings chosen in this register, the PRU-ICSS_UART0 transmitter sends the following to the receiving device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1, 1.5, or 2 STOP bits

6.4.7.2.3.2 PRU-ICSS UART Reception Protocol

The PRU-ICSS_UART0 receiver section includes a receiver shift register (RSR), that is not memory mapped, and a receiver buffer register (RBR), memory mapped as the register PRUSS_UART_RBR_THR_REGISTERS [7:0] DATA bitfield. When the PRU-ICSS_UART0 is in the FIFO mode, RBR is a 16-byte FIFO. Receiver section control is a function of the PRU-ICSS_UART0 line control register - PRUSS_UART_LINE_CONTROL_REGISTER. Based on the settings chosen in this register, the PRU-ICSS_UART0 receiver accepts the following from the transmitting device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1 STOP bit (any other STOP bits transferred with the above data are not detected)

6.4.7.2.3.3 PRU-ICSS UART Data Format

The PRU-ICSS_UART0 transmits in the following format:

1 START bit + data bits (5, 6, 7, 8) + 1 PARITY bit (optional) + STOP bit (1, 1.5, 2)

It transmits 1 START bit; 5, 6, 7, or 8 data bits, depending on the data width selection; 1 PARITY bit, if parity is selected; and 1, 1.5, or 2 STOP bits, depending on the STOP bit selection.

The PRU-ICSS_UART0 receives in the following format:

1 START bit + data bits (5, 6, 7, 8) + 1 PARITY bit (optional) + 1 STOP bit

It receives 1 START bit; 5, 6, 7, or 8 data bits, depending on the data width selection; 1 PARITY bit, if parity is selected; and 1 STOP bit.

The protocol formats are shown in Figure 6-205.

Figure 6-205. PRU-ICSS UART Protocol Formats

Transmit/Receive for 5-bit data, parity Enable, 1 STOP bit

		D0	D1	D2	D3	D4	PARITY	STOP1

Transmit/Receive for 6-bit data, parity Enable, 1 STOP bit

		D0	D1	D2	D3	D4	D5	PARITY	STOP1

Transmit/Receive for 7-bit data, parity Enable, 1 STOP bit

		D0	D1	D2	D3	D4	D5	D6	PARITY	STOP1

Transmit/Receive for 8-bit data, parity Enable, 1 STOP bit

		D0	D1	D2	D3	D4	D5	D6	D7	PARITY	STOP1

6.4.7.2.3.3.1 Frame Formatting

Character length is specified using the PRUSS_UART_LINE_CONTROL_REGISTER[1-0] WLS bit field (see Table 11-3499).

The number of stop-bits is specified using the PRUSS_UART_LINE_CONTROL_REGISTER[2] STB bit (see Table 11-3499).

The parity bit is programmed using the PRUSS_UART_LINE_CONTROL_REGISTER[5-3] PEN, EPS, and SP bits (see Table 11-3498).

Table 6-563. Relationship Between ST, EPS, and PEN Bits in UART_LCR

ST Bit	EPS Bit	PEN Bit	Parity Option
x	x	0	Parity disabled: No PARITY bit is transmitted or checked.
0	0	1	Odd parity selected: Odd number of logic 1s.
0	1	1	Even parity selected: Even number of logic 1s.
1	0	1	Stick parity selected with PARITY bit transmitted and checked as set.
1	1	1	Stick parity selected with PARITY bit transmitted and checked as cleared.

Table 6-564. Number of STOP Bits Generated

STB Bit	WLS Bit	Word Length Selected with WLS Bits	Number of STOP Bits Generated	Baud Clock (BCLK) Cycles
0	x	Any word length	1	16
1	0h	5 bits	1.5	24
1	1h	6 bits	2	32
1	2h	7 bits	2	32
1	3h	8 bits	2	32

6.4.7.2.4 PRU-ICSS UART Clock Generation and Control

The PRU-ICSS_UART0 bit clock is derived from an input clock to the PRU-ICSS_UART0. See the device Data Manual to check the maximum data rate supported by the PRU-ICSS_UART0.

Figure 6-206 is a conceptual clock generation diagram for the PRU-ICSS_UART0. The processor clock generator receives a signal from an external clock source and produces a PRU-ICSS_UART0 input clock with a programmed frequency. The PRU-ICSS_UART0 contains a programmable baud generator that takes an input clock and divides it by a divisor in the range between 1 and (2^{16} - 1) to produce a baud clock (BCLK). The frequency of BCLK is sixteen times (16×) the baud rate (each received or transmitted bit lasts 16 BCLK cycles) or thirteen times (13×) the baud rate (each received or transmitted bit lasts 13

BCLK cycles). When the PRU-ICSS_UART0 is receiving, the bit is sampled in the 8th BCLK cycle for 16× over sampling mode and on the 6th BCLK cycle for 13× over-sampling mode. The 16× or 13× reference clock is selected by configuring the mode definition register (MDR) - PRUSS_UART_MODE_DEFINITION_REGISTER [0] OSM_SEL bit. The formula to calculate the divisor is:

$$\text{Divisor} = \frac{\text{UART input clock frequency}}{\text{Desired baud rate} \times 16} \quad \text{[MDR.OSM\_SEL = 0]}$$

pruss-13

$$\text{Divisor} = \frac{\text{UART input clock frequency}}{\text{Desired baud rate} \times 13} \quad \text{[MDR.OSM\_SEL = 1]}$$

icss-14

Two 8-bit register fields:

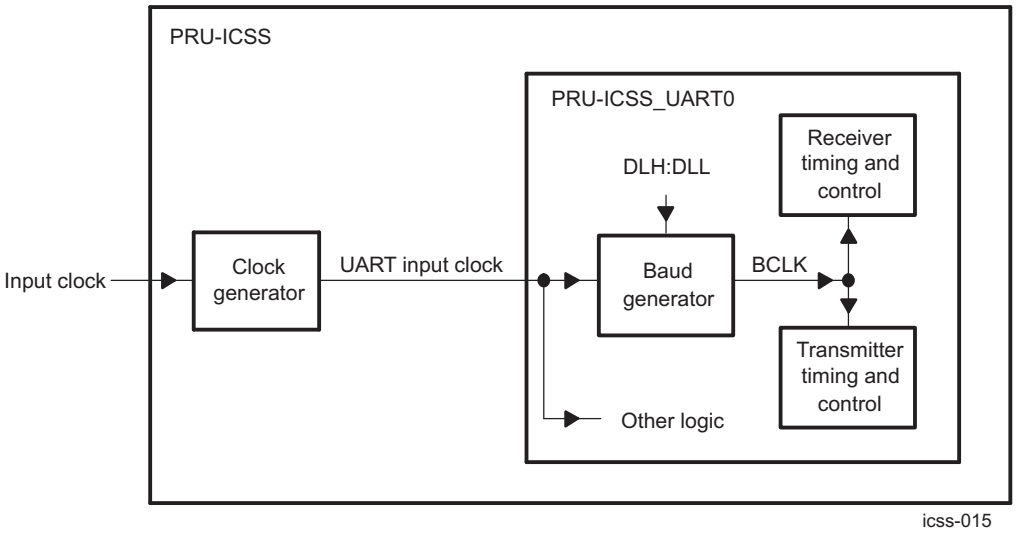
- PRUSS_UART_DIVISOR_REGISTER_MSB_[7:0] DLH
- PRUSS_UART_DIVISOR_REGISTER_LSB_[7:0] DLL,

called divisor latches, hold this 16-bit divisor. DLH holds the most significant bits of the divisor, and DLL holds the least significant bits of the divisor. For information about these register fields, see the PRU-ICSS_UART0 register descriptions in the Section 6.4.7.4, *PRU-ICSS UART Register Manual*. These divisor latches must be loaded during initialization of the PRU-ICSS_UART0 in order to ensure desired operation of the baud generator. Writing to the divisor latches results in two wait states being inserted during the write access while the baud generator is loaded with the new value.

Figure 6-207 summarizes the relationship between the transferred data bit, BCLK, and the PRU-ICSS_UART0 input clock. Note that the timing relationship depicted in Figure 6-207 shows that each bit lasts for 16 BCLK cycles . This is in case of 16x over-sampling mode. For 13× over-sampling mode each bit lasts for 13 BCLK cycles .

Example baud rates and divisor values relative to a 150-MHz PRU-ICSS_UART0 input clock and 16× over-sampling mode are shown in Table 6-565.

Figure 6-206. PRU-ICSS UART Clock Generation Diagram



icss-015

Figure 6-207. Relationships Between PRU-ICSS UART Data Bit, BCLK, and Input Clock

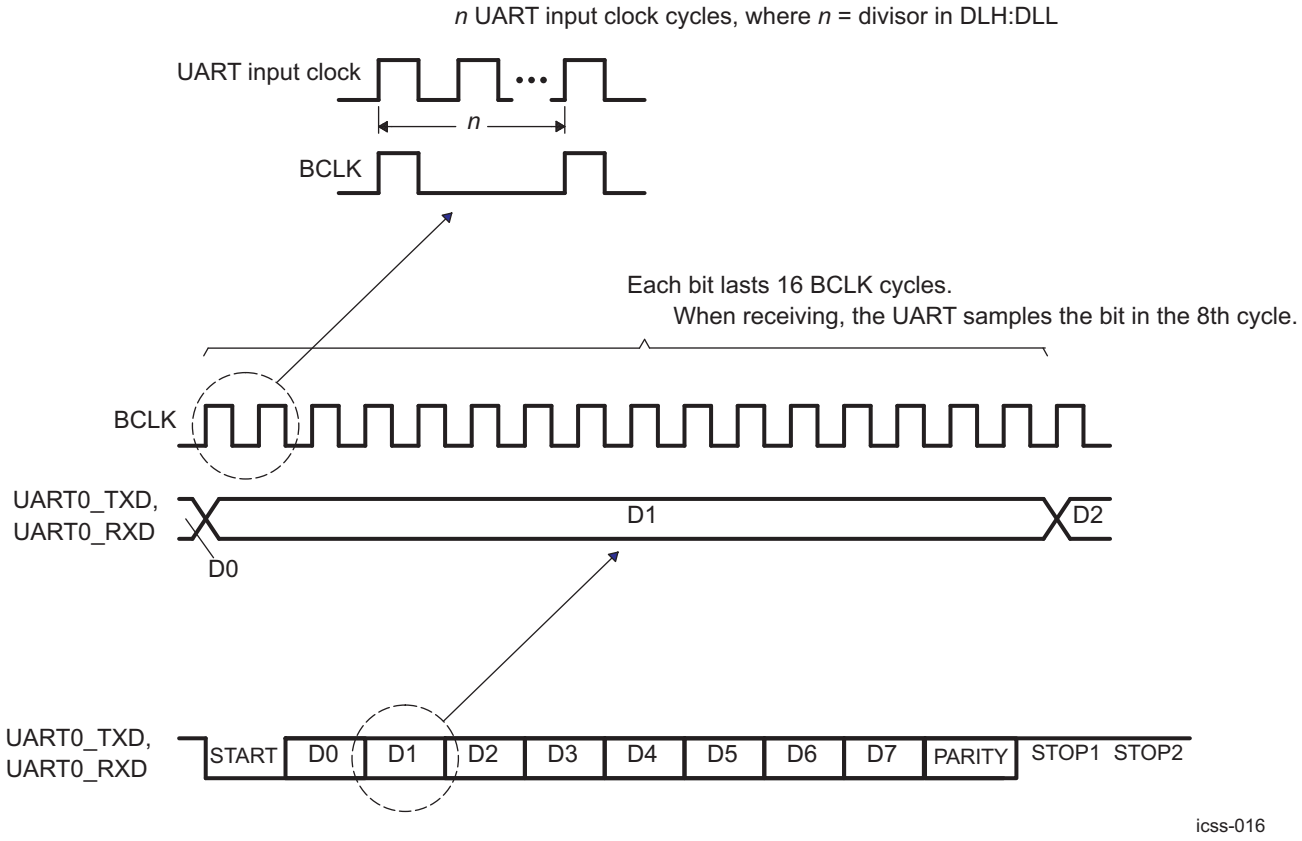


Table 6-565. Baud Rate Examples for 192-MHZ PRU-ICSS UART Input Clock and 16× Over-sampling Mode

Baud Rate	Divisor Value	Actual Baud Rate	Error (%)
2400	5000	2400	0.00
4800	2500	4800	0.00
9600	1250	9600	0.00
19200	625	19200	0.00
38400	313	38338.658	–0.16
56000	214	56074.766	0.13
115200	104	115384.6	0.16
128000	94	127659.574	–0.27
3000000	4	3000000	0.00
6000000	2	3000000	0.00
12000000	1	12000000	12000000

Table 6-566. Baud Rate Examples for 192-MHZ PRU-ICSS UART Input Clock and 13× Over-sampling Mode

Baud Rate	Divisor Value	Actual Baud Rate	Error (%)
2400	6154	2399.940	–0.0025
4800	3077	4799.880	–0.0025

Table 6-566. Baud Rate Examples for 192-MHZ PRU-ICSS UART Input Clock and 13× Over-sampling Mode (continued)

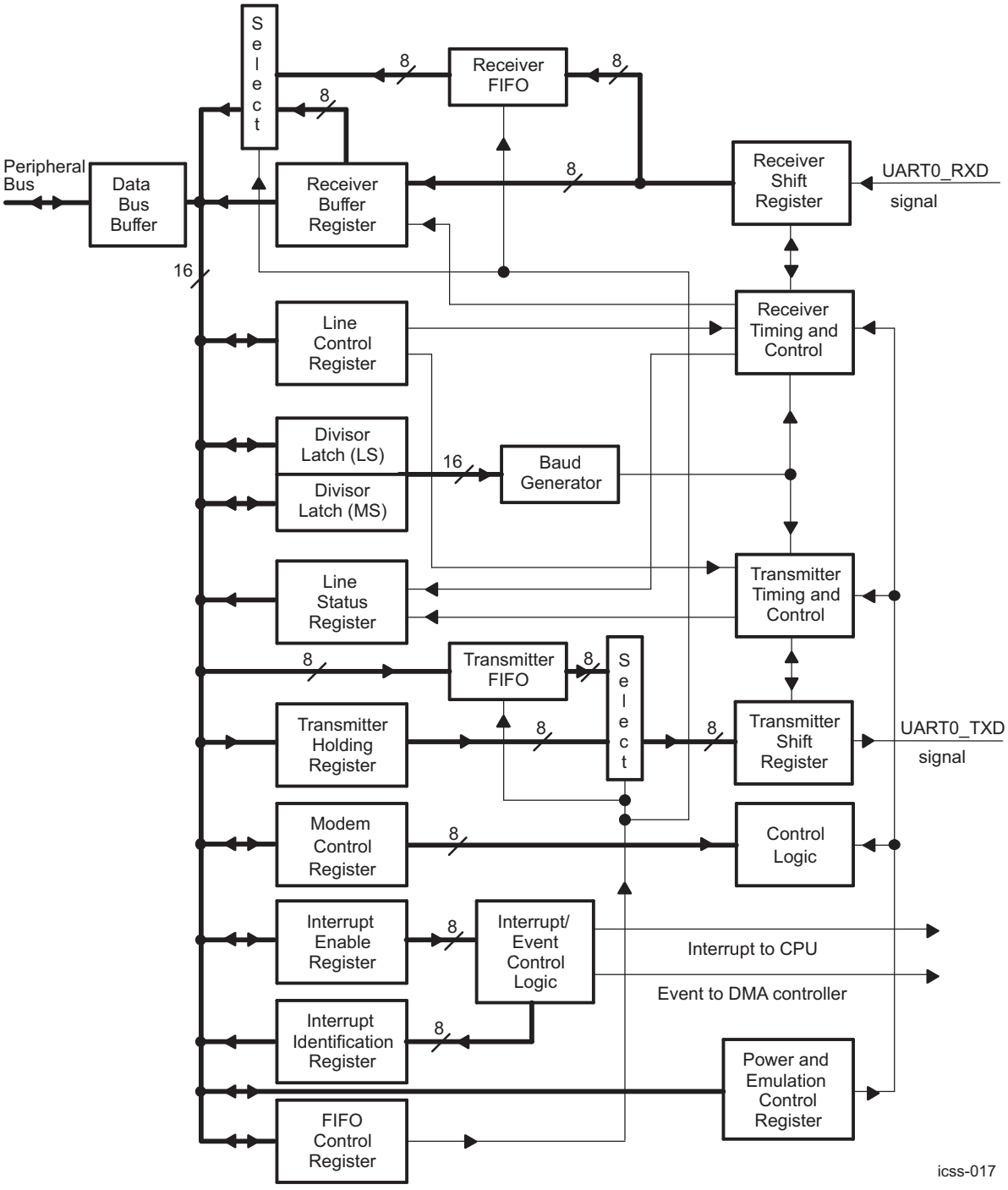
Baud Rate	Divisor Value	Actual Baud Rate	Error (%)
9600	1538	9602.881	0.03
19200	769	19205.762	0.03
38400	385	38361.638	–0.10
56000	264	55944.056	–0.10
115200	128	115384.6	0.16
128000	115	128428.094	0.33

6.4.7.3 PRU-ICSS UART Module Functional Description

6.4.7.3.1 PRU-ICSS UART Functional Block Diagram

A functional block diagram of the PRU-ICSS_UART0 is shown in Figure 6-208.

Figure 6-208. PRU-ICSS UART Block Diagram



NOTE: The value *n* indicates the applicable UART where there are multiple instances. For the PRU-ICSS, there is only one instance and all UART signals should reflect this (e.g., UART0_TXD instead of UARTn_TXD).

6.4.7.3.2 PRU-ICSS UART Reset Considerations

6.4.7.3.2.1 PRU-ICSS UART Software Reset Considerations

Two bits in the power and emulation management register - PRUSS_UART_POWERMANAGEMENT_AND_EMULATION_REGISTER, control resetting the parts of the PRU-ICSS_UART0:

- The bit [14] UTRST controls resetting the transmitter only. If UTRST = 1, the transmitter is active; if UTRST = 0, the transmitter is in reset.
- The bit [13] URRST controls resetting the receiver only. If URRST = 1, the receiver is active; if URRST = 0, the receiver is in reset.

In each case, putting the receiver and/or transmitter in reset will reset the state machine of the affected portion but does not affect the PRU-ICSS_UART0 registers.

6.4.7.3.2.2 PRU-ICSS UART Hardware Reset Considerations

When the processor RESET pin is asserted, the entire processor is reset and is held in the reset state until the RESET pin is released. As part of a device reset, the PRU-ICSS_UART0 state machine is reset and the PRU-ICSS_UART0 registers are forced to their default states. The default states of the registers are shown in .

6.4.7.3.3 PRU-ICSS UART Power Management

The PRU-ICSS_UART0 peripheral can be placed in reduced-power modes to conserve power during periods of low activity. The power management of the PRU-ICSS_UART0 peripheral and other PRU-ICSS peripherals is controlled by the device Power Sleep Controler (PSC). The PSC acts as a master controller for power management for all of the peripherals on the device. For more details on the power management procedures using the PSC, refer to the Section 5.2, *Power Management.*

6.4.7.3.4 PRU-ICSS UART Interrupt Support

6.4.7.3.4.1 PRU-ICSS UART Interrupt Events and Requests

The PRU-ICSS_UART0 generates the interrupt requests described in Table 6-567. All requests are multiplexed through an arbiter to a single PRU-ICSS_UART0 interrupt request to the CPU, as shown in Figure 6-209. Each of the interrupt requests has an enable bit in the interrupt enable register (IER) - PRUSS_UART_INTERRUPT_ENABLE_REGISTER and is recorded in INTID bitfield of PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER.

If an interrupt occurs and the corresponding enable bit is set to 1, the interrupt request is recorded in corresponding INTID bitfield and is forwarded to the CPU. If an interrupt occurs and the corresponding enable bit is cleared to 0, the interrupt request is blocked. The interrupt request is neither recorded in INTID, nor forwarded to the CPU.

6.4.7.3.4.2 PRU-ICSS UART Interrupt Multiplexing

The PRU-ICSS_UART0 have dedicated interrupt signals to the CPU and the interrupts are not multiplexed with any other interrupt source.

Table 6-567. PRU-ICSS UART Interrupt Requests Descriptions

PRU-ICSS_UART0 Interrupt Request	Interrupt Source	Comment
THREINT	THR-empty condition: The transmitter holding register (THR) or the transmitter FIFO is empty. All of the data has been copied from THR, (i.e. PRUSS_UART_RBR_THR_REGISTERS [7:0] DATA) to the transmitter shift register (TSR).	If THREINT is enabled in PRUSS_UART_INTERRUPT_ENABLE_REGISTER, by setting the ETBEI bit, it is recorded in INTID bitfield. As an alternative to using THREINT, the CPU can poll the THRE bit in the line status register PRUSS_UART_LINE_STATUS_REGISTER.

Table 6-567. PRU-ICSS UART Interrupt Requests Descriptions (continued)

PRU-ICSS_UART0 Interrupt Request	Interrupt Source	Comment
RDAINT	Receive data available in non-FIFO mode or trigger level reached in the FIFO mode.	If RDAINT is enabled in PRUSS_UART_INTERRUPT_ENABLE_REGISTER, by setting the ERBI bit, it is recorded in INTID bitfield. As an alternative to using RDAINT, the CPU can poll the DR bit in the line status register PRUSS_UART_LINE_STATUS_REGISTER. In the FIFO mode, this is not a functionally equivalent alternative because the DR bit does not respond to the FIFO trigger level. The DR bit only indicates the presence or absence of unread characters.
RTOINT	Receiver time-out condition (in the FIFO mode only): No characters have been removed from or input to the receiver FIFO during the last four character times (see Table 6-569), and there is at least one character in the receiver FIFO during this time.	The receiver time-out interrupt prevents the PRUSS_UART0 from waiting indefinitely, in the case when the receiver FIFO level is below the trigger level and thus does not generate a receiver data-ready interrupt. If RTOINT is enabled in PRUSS_UART_INTERRUPT_ENABLE_REGISTER, by setting the ERBI bit, it is recorded in INTID bitfield. There is no status bit to reflect the occurrence of a time-out condition.
RLSINT	Receiver line status condition: An overrun error, parity error, framing error, or break has occurred.	If RLSINT is enabled in PRUSS_UART_INTERRUPT_ENABLE_REGISTER, by setting the ELSI bit, it is recorded in INTID bitfield. As an alternative to using RLSINT, the CPU can poll the following bits in the line status register PRUSS_UART_LINE_STATUS_REGISTER: overrun error indicator (OE), parity error indicator (PE), framing error indicator (FE), and break indicator (BI).

Figure 6-209. PRU-ICSS UART Interrupt Request Enable Paths

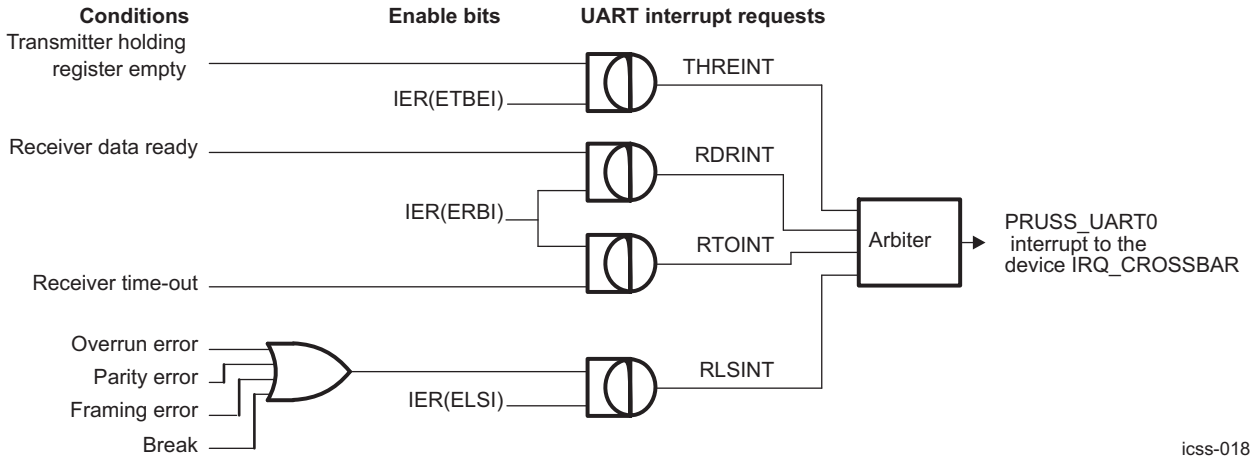


Table 6-568. Interrupt Identification and Interrupt Clearing Information

Priority Level	IIR Bits				Interrupt Type	Interrupt Source	Event That Clears Interrupt
	3	2	1	0			
None	0	0	0	1	None	None	None
1	0	1	1	0	Receiver line status	Overrun error, parity error, framing error, or break is detected.	For an overrun error, reading the PRUSS_UART_LINE_STATUS_REGISTER clears the interrupt. For a parity error, framing error, or break, the interrupt is cleared only after all the erroneous data have been read.
2	0	1	0	0	Receiver data-ready	Non-FIFO mode: Receiver data is ready.	Non-FIFO mode: The receiver buffer register (RBR) is read.
						FIFO mode: Trigger level reached. If four character times (see) pass with no access of the FIFO, the interrupt is asserted again.	FIFO mode: The FIFO drops below the trigger level. (1)
2	1	1	0	0	Receiver time-out	FIFO mode only: No characters have been removed from or input to the receiver FIFO during the last four character times (see), and there is at least one character in the receiver FIFO during this time.	One of the following events: • A character is read from the receiver FIFO (1) • A new character arrives in the receiver FIFO • The URRST bit in the power and emulation management register (PRUSS_UART_POWERMANAGEMENT_AND_EMULATION_REGISTER) is loaded with 0.
3	0	0	1	0	Transmitter holding register empty	Non-FIFO mode: Transmitter holding register (THR) is empty.	A character is written to the transmitter holding register (THR) or the interrupt identification register (IIR) is read.
						FIFO mode: Transmitter FIFO is empty.	

(1) In the FIFO mode, the receiver data-ready interrupt or receiver time-out interrupt is cleared by the CPU or by the DMA controller, whichever reads from the receiver FIFO first.

6.4.7.3.5 PRU-ICSS UART DMA Event Support

In the FIFO mode, the PRU-ICSS_UART0 generates the following two DMA events:

- **Receive event (URXEVT):** The trigger level for the receiver FIFO (1, 4, 8, or 14 characters) is set with the FIFO control PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER [7:6] FIFOEN_RXFIFTL bitfield. Every time the trigger level is reached or a receiver time-out occurs, the PRU-ICSS_UART0 sends a receive event to the EDMA controller. In response, the EDMA controller reads the data from the receiver FIFO by way of the receiver buffer register PRUSS_UART_RBR_THR_REGISTERS [7:0] DATA. Note that the receive event is not asserted if the data at the top of the receiver FIFO is erroneous even if the trigger level has been reached.
- **Transmit event (UTXEVT):** When the transmitter FIFO is empty (when the last byte in the transmitter FIFO has been copied to the transmitter shift register), the PRU-ICSS_UART0 sends an UTXEVT signal to the EDMA controller. In response, the EDMA controller refills the transmitter FIFO by way of the transmitter holding register (THR) - PRUSS_UART_RBR_THR_REGISTERS [7:0] DATA. The UTXEVT signal is also sent to the DMA controller when the PRU-ICSS_UART0 is taken out of reset using the UTRST bit in the power and emulation management register (PRUSS_UART_POWERMANAGEMENT_AND_EMULATION_REGISTER).

Activity in DMA channels can be synchronized to these events. In the non-FIFO mode, the PRU-ICSS_UART0 generates no DMA events. Any DMA channel synchronized to either of these events must be enabled at the time the PRU-ICSS_UART0 event is generated. Otherwise, the DMA channel will miss the event and, unless the PRU-ICSS_UART0 generates a new event, no data transfer will occur.

6.4.7.3.6 PRU-ICSS UART Operations

6.4.7.3.6.1 PRU-ICSS UART Transmission

The PRU-ICSS_UART0 transmitter section includes a transmitter hold register (THR) mapped in the PRUSS_UART_RBR_THR_REGISTERS [7:0] DATA bitfield and a transmitter shift register (TSR). When the PRU-ICSS_UART0 is in the FIFO mode, THR is a 16-byte FIFO. Transmitter section control is a function of the PRU-ICSS_UART0 line control register PRUSS_UART_LINE_CONTROL_REGISTER. Based on the settings chosen in this register, the PRU-ICSS_UART0 transmitter sends the following to the receiving device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1, 1.5, or 2 STOP bits

THR receives data from the internal data bus, and when TSR is ready, the PRU-ICSS_UART0 moves the data from THR to TSR. The PRU-ICSS_UART0 serializes the data in TSR and transmits the data on the UART0_TXD pin.

In the non-FIFO mode, if THR is empty and the THR empty (THRE) interrupt is enabled in the interrupt enable register PRUSS_UART_INTERRUPT_ENABLE_REGISTER, an interrupt is generated. This interrupt is cleared when a character is loaded into THR or the interrupt identification register PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER bitfield INTID is read. In the FIFO mode, the interrupt is generated when the transmitter FIFO is empty, and it is cleared when at least one byte is loaded into the FIFO or INTID bitfield is read.

6.4.7.3.6.2 PRU-ICSS UART Reception

The PRU-ICSS_UART0 receiver section includes a receiver shift register (RSR) and a receiver buffer register (RBR) mapped in PRUSS_UART_RBR_THR_REGISTERS [7:0] DATA bitfield. When the PRU-ICSS_UART0 is in the FIFO mode, RBR is a 16-byte FIFO. Timing is supplied by the 16× receiver clock. Receiver section control is a function of the PRU-ICSS_UART0 line control register PRUSS_UART_LINE_CONTROL_REGISTER. Based on the settings chosen in this register, the PRU-ICSS_UART0 receiver accepts the following from the transmitting device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1 STOP bit (any other STOP bits transferred with the above data are not detected)

RSR receives the data bits from the UART0_RXD pin. Then RSR concatenates the data bits and moves the resulting value into RBR (or the receiver FIFO), accessible in the PRUSS_UART_RBR_THR_REGISTERS [7:0] DATA register bitfield. The PRU-ICSS_UART0 also stores three bits of error status information next to each received character, to record a parity error, framing error, or break.

In the non-FIFO mode, when a character is placed in RBR and the receiver data-ready interrupt is enabled in the interrupt enable register - PRUSS_UART_INTERRUPT_ENABLE_REGISTER, an interrupt is generated. This interrupt is cleared when the character is read from RBR. In the FIFO mode, the interrupt is generated when the FIFO is filled to the trigger level selected in the FIFO control MSB part of the register PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER, and it is cleared when the FIFO contents drop below the trigger level.

6.4.7.3.6.3 PRU-ICSS UART FIFO Modes

The following two modes can be used for servicing the receiver and transmitter FIFOs:

- FIFO interrupt mode. The FIFO is enabled and the associated interrupts are enabled. Interrupts are sent to the CPU to indicate when specific events occur.
- FIFO poll mode. The FIFO is enabled but the associated interrupts are disabled. The CPU polls status bits to detect specific events.

Because the receiver FIFO and the transmitter FIFO are controlled separately, either one or both can be placed into the interrupt mode or the poll mode.

6.4.7.3.6.3.1 PRU-ICSS UART FIFO Interrupt Mode

When the receiver FIFO is enabled in the FIFO control register (FCR), mapped in the MSB part of the register PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER, and the receiver interrupts are enabled in the interrupt enable register PRUSS_UART_INTERRUPT_ENABLE_REGISTER, the interrupt mode is selected for the receiver FIFO. The following are important points about the receiver interrupts:

- The receiver data-ready interrupt is issued to the CPU when the FIFO has reached the trigger level that is programmed in FCR. It is cleared when the CPU or the DMA controller reads enough characters from the FIFO such that the FIFO drops below its programmed trigger level.
- The receiver line status interrupt is generated in response to an overrun error, a parity error, a framing error, or a break. This interrupt has higher priority than the receiver data-ready interrupt. For details, see Section 6.4.7.3.4.
- The data-ready (DR) bit in the line status register (LSR) - PRUSS_UART_LINE_STATUS_REGISTER, indicates the presence or absence of characters in the receiver FIFO. The DR bit is set when a character is transferred from the receiver shift register (RSR) to the empty receiver FIFO. The DR bit remains set until the FIFO is empty again.
- A receiver time-out interrupt occurs if all of the following conditions exist:
  - At least one character is in the FIFO,
  - The most recent character was received more than four continuous character times ago. A character time is the time allotted for 1 START bit, *n* data bits, 1 PARITY bit, and 1 STOP bit, where *n* depends on the word length selected with the WLS0 and WLS1 bits of the line control register PRUSS_UART_LINE_CONTROL_REGISTER. See Table 6-569.
  - The most recent read of the FIFO has occurred more than four continuous character times before.
- Character times are calculated by using the baud rate.
- When a receiver time-out interrupt has occurred, it is cleared and the time-out timer is cleared when the CPU or the EDMA controller reads one character from the receiver FIFO. The interrupt is also cleared if a new character is received in the FIFO or if the URRST bit is cleared in the power and emulation management register PRUSS_UART_POWERMANAGEMENT_AND_EMULATION_REGISTER.
- If a receiver time-out interrupt has not occurred, the time-out timer is cleared after a new character is received or after the CPU or EDMA reads the receiver FIFO.

When the transmitter FIFO is enabled in PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER [0] IPEND_FIFOEN bit and the transmitter holding register empty (THRE) interrupt is enabled in PRUSS_UART_INTERRUPT_ENABLE_REGISTER[1] ETBEI bit, the interrupt mode is selected for the transmitter FIFO. The THRE interrupt occurs when the transmitter FIFO is empty. It is cleared when the transmitter hold register (THR) PRUSS_UART_RBR_THR_REGISTERS [7:0] DATA bitfield is loaded (1 to 16 characters may be written to the transmitter FIFO while servicing this interrupt) or the interrupt identification register INTID bitfield is read in the PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER.

Table 6-569. Character Time for Word Lengths

Word Length (*n*)	Character Time	Four Character Times
5	Time for 8 bits	Time for 32 bits
6	Time for 9 bits	Time for 36 bits
7	Time for 10 bits	Time for 40 bits
8	Time for 11 bits	Time for 44 bits

6.4.7.3.6.3.2 PRU-ICSS UART FIFO Poll Mode

When the receiver FIFO is enabled in the FIFO control register (via setting the PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER [0] IPEND_FIFOEN to 0b1) and the receiver interrupts are disabled in the interrupt enable register (PRUSS_UART_INTERRUPT_ENABLE_REGISTER), the poll mode is selected for the receiver FIFO. Similarly, when the transmitter FIFO is enabled via setting the same bit (PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER [0] IPEND_FIFOEN to 0b1) and the transmitter interrupts are disabled, the transmitted FIFO is in the poll mode. In the poll mode, the CPU detects events by checking bits in the line status register - PRUSS_UART_LINE_STATUS_REGISTER:

- The PRUSS_UART_LINE_STATUS_REGISTER[7] RXFIFOE bit indicates whether there are any errors in the receiver FIFO.
- The PRUSS_UART_LINE_STATUS_REGISTER[6] TEMT bit indicates that both the transmitter holding register (THR) and the transmitter shift register (TSR) are empty.
- The PRUSS_UART_LINE_STATUS_REGISTER[5] THRE bit indicates when THR (mapped in the PRUSS_UART_RBR_THR_REGISTERS [7:0] DATA bitfield) is empty.
- The following PRUSS_UART_LINE_STATUS_REGISTER bits specify which error or errors have occurred:
  - PRUSS_UART_LINE_STATUS_REGISTER[4] BI - Break Interrupt
  - PRUSS_UART_LINE_STATUS_REGISTER[3] FE – Framing Error
  - PRUSS_UART_LINE_STATUS_REGISTER[2] PE – Parity Error
  - PRUSS_UART_LINE_STATUS_REGISTER[1] OE – Overrun Error
- The PRUSS_UART_LINE_STATUS_REGISTER[0] DR (data-ready) bit is set as long as there is at least one byte in the receiver FIFO.

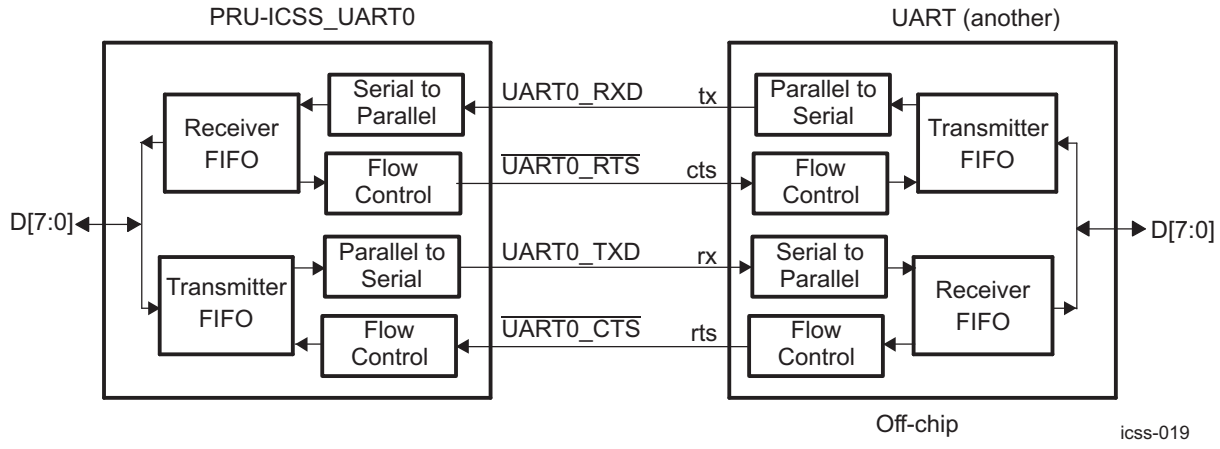
Also, in the FIFO poll mode:

- The interrupt identification register (INTID) bitfields are not affected by any events because the interrupts are disabled.
- The PRU-ICSS_UART0 does not indicate when the receiver FIFO trigger level is reached or when a receiver time-out occurs.

6.4.7.3.6.4 PRU-ICSS UART Autoflow Control

The PRU-ICSS_UART0 can employ autoflow control by connecting the $\overline{\text{UART0\_CTS}}$ and $\overline{\text{UART0\_RTS}}$ signals. The $\overline{\text{UART0\_CTS}}$ input must be active before the transmitter FIFO can transmit data. The $\overline{\text{UART0\_RTS}}$ becomes active when the receiver needs more data and notifies the sending device. When $\overline{\text{UART0\_RTS}}$ is connected to $\overline{\text{UART0\_CTS}}$, data transmission does not occur unless the receiver FIFO has space for the data. Therefore, when two UARTs are connected as shown in Figure 6-210 with autoflow enabled, overrun errors are eliminated.

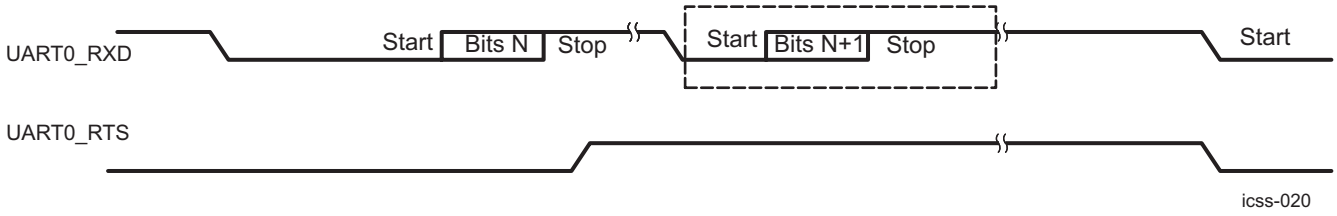
Figure 6-210. UART Interface Using Autoflow Diagram



6.4.7.3.6.4.1 PRU-ICSS UART Signal UART0_RTS Behavior

UART0_RTS data flow control originates in the receiver block (see Figure 6-208). When the receiver FIFO level reaches a trigger level of 1, 4, 8, or 14 (see Figure 6-211), UART0_RTS is deasserted. The sending UART may send an additional byte after the trigger level is reached (assuming the sending UART has another byte to send), because it may not recognize the deassertion of UART0_RTS until after it has begun sending the additional byte. For trigger level 1, 4, and 8, UART0_RTS is automatically reasserted once the receiver FIFO is emptied. For trigger level 14, UART0_RTS is automatically reasserted once the receiver FIFO drops below the trigger level.

Figure 6-211. Autoflow Functional Timing Waveforms for UART0_RTS



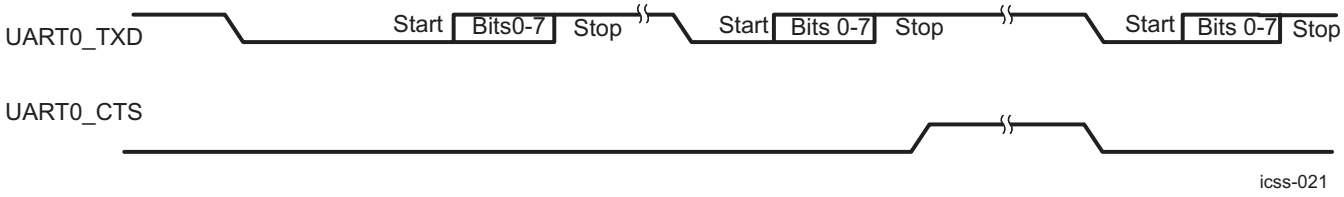
(1) N = Receiver FIFO trigger level.

(2) The two blocks in dashed lines cover the case where an additional byte is sent.

6.4.7.3.6.4.2 PRU-ICSS UART Signal UART0_CTS Behavior

The transmitter checks UART0_CTS before sending the next data byte. If UART0_CTS is active, the transmitter sends the next byte. To stop the transmitter from sending the following byte, UART0_CTS must be released before the middle of the last STOP bit that is currently being sent (see Figure 6-212). When flow control is enabled, UART0_CTS level changes do not trigger interrupts because the device automatically controls its own transmitter. Without autoflow control, the transmitter sends any data present in the transmitter FIFO and a receiver overrun error may result.

Figure 6-212. Autoflow Functional Timing Waveforms for UART0_CTS



(1) When UART0_CTS is active (low), the transmitter keeps sending serial data out.

(2) When UART0_CTS goes high before the middle of the last STOP bit of the current byte, the transmitter finishes sending the current byte but it does not send the next byte.

(3) When UART0_CTS goes from high to low, the transmitter begins sending data again.

6.4.7.3.6.5 PRU-ICSS UART Loopback Control

The PRU-ICSS_UART0 can be placed in the diagnostic mode using the LOOP bit in the modem control register - PRUSS_UART_MODEM_CONTROL_REGISTER, which internally connects the PRU-ICSS_UART0 output back to the PRU-ICSS_UART0's input. In this mode, the transmit and receive data paths, the transmitter and receiver interrupts, and the modem control interrupts can be verified without connecting to another UART.

6.4.7.3.7 PRU-ICSS UART Initialization

The following steps are required to initialize the PRU-ICSS_UART0:

1. Perform the necessary device pin multiplexing setup (see the device Data Manual).

2. Set the desired baud rate by writing the appropriate clock divisor values to the divisor latch registers PRUSS_UART_DIVISOR_REGISTER_MSB_ [7:0] DLH and PRUSS_UART_DIVISOR_REGISTER_LSB_ [7:0] DLL.
3. If the FIFOs will be used, select the desired trigger level and enable the FIFOs by writing the appropriate values to the FIFO control register. The PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER[0] IPEND_FIFOEN bit must be set first, before the other bits in this register are configured.
4. Choose the desired protocol settings by writing the appropriate values to the line control register PRUSS_UART_LINE_CONTROL_REGISTER.
5. If autoflow control is desired, write appropriate values to the modem control register PRUSS_UART_MODEM_CONTROL_REGISTER.
6. Choose the desired response to emulation suspend events by configuring the FREE bit and enable the PRU-ICSS_UART0 by setting the UTRST and URRST bits in the power and emulation management register -PRUSS_UART_POWERMANAGEMENT_AND_EMULATION_REGISTER.

6.4.7.3.8 PRU-ICSS UART Exception Processing

6.4.7.3.8.1 PRU-ICSS UART Divisor Latch Not Programmed

Since the processor reset signal has no effect on the divisor latch, the divisor latch will have an unknown value after power up. If the divisor latch is not programmed after power up, the baud clock (BCLK) will not operate and will instead be set to a constant logic 1 state.

The divisor latch values should always be reinitialized following a processor reset.

6.4.7.3.8.2 Changing Operating Mode During Busy Serial Communication of PRU-ICSS UART

Since the serial link characteristics are based on how the control registers are programmed, the PRU-ICSS_UART0 will expect the control registers to be static while it is busy engaging in a serial communication. Therefore, changing the control registers while the module is still busy communicating with another serial device will most likely cause an error condition and should be avoided.

6.4.7.4 PRU-ICSS_UART Registers

Table 6-571 lists the memory-mapped registers for the PRU-ICSS_UART module. All register offset addresses not listed in Table 6-571 should be considered as reserved locations and the register contents should not be modified.

Table 6-570. PRU-ICSS_UART Instances

Instance	Base Address
PRU_ICSS_0_UART	20AA 8000h
PRU_ICSS_1_UART	20AE 8000h

Table 6-571. PRU-ICSS_UART Registers

Offset	Acronym	Register Name	PRU_ICSS_0_UART Physical Address	PRU_ICSS_1_UART Physical Address	Section
0h	PRUSS_UART_RBR_THR_REGISTERS	Transmitter Holding Register	20AA 8000h	20AE 8000h	Section 6.4.7.4.1
4h	PRUSS_UART_INTERRUPT_ENABLE_REGISTER	Interrupt Enable Register	20AA 8004h	20AE 8004h	Section 6.4.7.4.2
8h	PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER	Interrupt Identification Register/ FIFO Control Register	20AA 8008h	20AE 8008h	Section 6.4.7.4.3
Ch	PRUSS_UART_LINE_CONTROL_REGISTER	Line Control Register	20AA 800Ch	20AE 800Ch	Section 6.4.7.4.4
10h	PRUSS_UART_MODEM_CONTROL_REGISTER	Modem Control Register	20AA 8010h	20AE 8010h	Section 6.4.7.4.5
14h	PRUSS_UART_LINE_STATUS_REGISTER	Line Status Register	20AA 8014h	20AE 8014h	Section 6.4.7.4.6
18h	PRUSS_UART_MODEM_STATUS_REGISTER	Modem Status Register	20AA 8018h	20AE 8018h	Section 6.4.7.4.7
1Ch	PRUSS_UART_SCRATCH_REGISTER	Scratch Register	20AA 801Ch	20AE 801Ch	Section 6.4.7.4.8
20h	PRUSS_UART_DIVISOR_REGISTER_LSB_	Divisor Latch (LSB)	20AA 8020h	20AE 8020h	Section 6.4.7.4.9
24h	PRUSS_UART_DIVISOR_REGISTER_MSB_	Divisor Latch (MSB)	20AA 8024h	20AE 8024h	Section 6.4.7.4.10
28h	PRUSS_UART_PERIPHERAL_ID_REGISTER	Peripheral ID Register	20AA 8028h	20AE 8028h	Section 6.4.7.4.11
2Ch	RESERVED		20AA 802Ch	20AE 802Ch	
30h	PRUSS_UART_POWERMANAGEMENT_AND_EMULATION_REGISTER	Power Management and Emulation Register	20AA 8030h	20AE 8030h	Section 6.4.7.4.12
34h	PRUSS_UART_MODE_DEFINITION_REGISTER	Mode Definition Register	20AA 8034h	20AE 8034h	Section 6.4.7.4.13

6.4.7.4.1 PRUSS_UART_RBR_THR_REGISTERS Register (Offset = 0h) [reset = 0h]

PRUSS_UART_RBR_TBR_REGISTERS is shown in Figure 6-213 and described in Table 6-573.

Return to Summary Table.

In the non-FIFO mode, when a character is placed in Receiver buffer register and the receiver data-ready interrupt is enabled (DR = 1 in Interrupt identification register), an interrupt is generated. This interrupt is cleared when the character is read from Receiver buffer register. In the FIFO mode, the interrupt is generated when the FIFO is filled to the trigger level selected in the FIFO control register, and it is cleared when the FIFO contents drop below the trigger level.
In the non-FIFO mode, if Transmitter holding register is empty and the THR empty (THRE) interrupt is enabled (ETBEI = 1 in Interrupt enable register), an interrupt is generated. This interrupt is cleared when a character is loaded into Transmitter holding register or the Interrupt identification register is read. In the FIFO mode, the interrupt is generated when the transmitter FIFO is empty, and it is cleared when at least one byte is loaded into the FIFO or Interrupt identification register is read.

Table 6-572. PRUSS_UART_RBR_THR_REGISTERS Instances

Instance	Physical Address
PRU_ICSS_0_UART	20AA 8000h
PRU_ICSS_1_UART	20AE 8000h

Figure 6-213. PRUSS_UART_RBR_THR_REGISTERS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DATA							
R-0h																								RW-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-573. PRUSS_UART_RBR_THR_REGISTERS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	DATA	RW	0h	**Read:** Read Receive Buffer Register **Write:** Write Transmitter Holding Register

Table 6-574. Register Call Summary for PRUSS_UART_RBR_THR_REGISTERS

PRU-ICSS UART Module

- PRU-ICSS_UART Registers: [0]
- PRU-ICSS UART Transmission: [0]
- PRU-ICSS UART DMA Event Support: [0][1]
- PRU-ICSS UART FIFO Modes: [0][1]
- PRU-ICSS UART Interrupt Support: [0]
- PRU-ICSS UART Reception: [0][1]
- PRU-ICSS UART Transmission Protocol: [0]
- PRU-ICSS UART Reception Protocol: [0]

6.4.7.4.2 PRUSS_UART_INTERRUPT_ENABLE_REGISTER Register (Offset = 4h) [reset = 0h]

PRUSS_UART_INTERRUPT_ENABLE_REGISTER is shown in Figure 6-214 and described in Table 6-576.

Return to Summary Table.

The Interrupt enable register is used to individually enable or disable each type of interrupt request that can be generated by the UART. Each interrupt request that is enabled in Interrupt enable register is forwarded to the CPU.

Table 6-575. PRUSS_UART_INTERRUPT_ENABLE_REGISTER Instances

Instance	Physical Address
PRU_ICSS_0_UART	20AA 8004h
PRU_ICSS_1_UART	20AE 8004h

Figure 6-214. PRUSS_UART_INTERRUPT_ENABLE_REGISTER Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				EDSSI	ELSI	ETBEI	ERBI
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-576. PRUSS_UART_INTERRUPT_ENABLE_REGISTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	EDSSI	R/W	0h	Enable Modem Status Interrupt
2	ELSI	R/W	0h	Receiver line status interrupt enable. 0h: Receiver line status interrupt is disabled. 1h: Receiver line status interrupt is enabled.
1	ETBEI	R/W	0h	Transmitter holding register empty interrupt enable. 0h: Transmitter holding register empty interrupt is disabled. 1h: Transmitter holding register empty interrupt is enabled.
0	ERBI	R/W	0h	Receiver data available interrupt and character timeout indication interrupt enable. 0h: Receiver data available interrupt and character timeout indication interrupt is disabled. 1h: Receiver data available interrupt and character timeout indication interrupt is enabled.

Table 6-577. Register Call Summary for PRUSS_UART_INTERRUPT_ENABLE_REGISTER

PRU-ICSS UART Module
• PRUSS_UART_INTERRUPT_ENABLE_REGISTER Register (Offset = 4h) [reset = 0h]: [0] • PRU-ICSS_UART Registers: [0] • PRU-ICSS UART Transmission: [0] • PRU-ICSS UART Reception: [0] • PRU-ICSS UART FIFO Modes: [0][1][2] • PRU-ICSS UART Interrupt Support: [0][1][2][3] • PRU-ICSS UART Interrupt Events and Requests: [0]

6.4.7.4.3 PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER Register (Offset = 8h) [reset = 1h]

PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER is shown in Figure 6-215 and described in Table 6-579.

Return to Summary Table.

The Interrupt identification register is a read-only register at the same address as the FIFO control register, which is a write-only register. When an interrupt is generated and enabled in the Interrupt enable register, Interrupt identification register indicates that an interrupt is pending in the IPEND bit and encodes the type of interrupt in the INTID bits. Reading Interrupt identification register clears any THR empty (THRE) interrupts that are pending. The FIFOEN bit in Interrupt identification register can be checked to determine whether the UART is in the FIFO mode or the non-FIFO mode.
Use FIFO control register to enable and clear the FIFOs and to select the receiver FIFO trigger level. The FIFOEN bit in FIFO control register must be set to 1 before other FIFO control register bits are written to or the FIFO control register bits are not programmed.

Table 6-578. PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER Instances

Instance	Physical Address
PRU_ICSS_0_UART	20AA 8008h
PRU_ICSS_1_UART	20AE 8008h

Figure 6-215. PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
FIFOEN_RXFIFTL		RESERVED		INTID			IPEND_FIFOEN
RW-0h		R-0h		RW-0h			RW-1h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-579. PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved

Table 6-579. PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	FIFOEN_RXFIFTL	RW	0h	**Read:** FIFOs enabled. 0h: Non-FIFO mode 1h-2h: Reserved 3h: FIFOs are enabled. FIFOEN bit in the FIFO control register (FCR) is set to 1.
				Write: Receiver FIFO trigger level. RXFIFTL sets the trigger level for the receiver FIFO. When the trigger level is reached, a receiver data-ready interrupt is generated (if the interrupt request is enabled). Once the FIFO drops below the trigger level, the interrupt is cleared. 0h: 1 byte 1h: 4 bytes 2h: 8 bytes 3h: 14 bytes
5-4	RESERVED	R	0h	Reserved
3-1	INTID	RW	0h	**Read:** Interrupt type. See Table 11-3495. 0h: Reserved 1h: Transmitter holding register empty (priority 3) 2h: Receiver data available (priority 2) 3h: Receiver line status (priority 1, highest) 4h-5h: Reserved 6h: Character timeout indication (priority 2) 7h: Reserved
				Write: **Bit 3: DMAMODE1:** DMA MODE1 enable if FIFOs are enabled. Always write 1 to DMAMODE1. After a hardware reset, change DMAMODE1 from 0 to 1. DMAMODE1 = 1 is a requirement for proper communication between the UART and the EDMA controller. 0h: DMA MODE1 is disabled. 1h: DMA MODE1 is enabled. **Bit 2: TXCLR:** Transmitter FIFO clear. Write a 1 to TXCLR to clear the bit. 0h: No effect. 1h: Clears transmitter FIFO and resets the transmitter FIFO counter. The shift register is not cleared. **Bit 1: RXCLR:** Receiver FIFO clear. Write a 1 to RXCLR to clear the bit. 0h: No effect. 1h: Clears receiver FIFO and resets the receiver FIFO counter. The shift register is not cleared.
0	IPEND_FIFOEN	RW	1h	**Read:** Interrupt pending. When any UART interrupt is generated and is enabled in IER, IPEND is forced to 0. IPEND remains 0 until all pending interrupts are cleared or until a hardware reset occurs. If no interrupts are enabled, IPEND is never forced to 0. 0h: Interrupts pending. 1h: No interrupts pending.
				Write: Transmitter and receiver FIFOs mode enable. FIFOEN must be set before other FCR bits are written to or the FCR bits are not programmed. Clearing this bit clears the FIFO counters. 0h: Non-FIFO mode. The transmitter and receiver FIFOs are disabled, and the FIFO pointers are cleared. 1h: FIFO mode. The transmitter and receiver FIFOs are enabled.

Table 6-580. Register Call Summary for PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER

PRU-ICSS UART Module

- PRU-ICSS_UART Registers: [0]
- PRU-ICSS UART Transmission: [0]
- PRUSS_UART_INTERRUPT_IDENTIFICATION_REGISTER_FIFO_CONTROL_REGISTER Register (Offset = 8h) [reset = 1h]: [0]
- PRU-ICSS UART Initialization: [0]
- PRU-ICSS UART DMA Event Support: [0]
- PRU-ICSS UART FIFO Modes: [0][1][2][3][4]
- PRU-ICSS UART Reception: [0]
- PRU-ICSS UART Interrupt Events and Requests: [0]

6.4.7.4.4 PRUSS_UART_LINE_CONTROL_REGISTER Register (Offset = Ch) [reset = 0h]

PRUSS_UART_LINE_CONTROL_REGISTER is shown in Figure 6-216 and described in Table 6-582.

Return to Summary Table.

The system programmer controls the format of the asynchronous data communication exchange by using Line control register. In addition, the programmer can retrieve, inspect, and modify the content of line control register; this eliminates the need for separate storage of the line characteristics in system memory.

Table 6-581. PRUSS_UART_LINE_CONTROL_REGISTER Instances

Instance	Physical Address
PRU_ICSS_0_UART	20AA 800Ch
PRU_ICSS_1_UART	20AE 800Ch

Figure 6-216. PRUSS_UART_LINE_CONTROL_REGISTER Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
DLAB	BC	SP	EPS	PEN	STB	WLS1	WLS0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-582. PRUSS_UART_LINE_CONTROL_REGISTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	DLAB	R/W	0h	Divisor latch access bit. The divisor latch registers (DLL and DLH) can be accessed at dedicated addresses or at addresses shared by RBR, THR, and IER. Using the shared addresses requires toggling DLAB to change which registers are selected. If you use the dedicated addresses, you can keep DLAB = 0. 0h: Allows access to the receiver buffer register (RBR), the transmitter holding register (THR), and the interrupt enable register (IER) selected. At the address shared by RBR, THR, and DLL, the CPU can read from RBR and write to THR. At the address shared by IER and DLH, the CPU can read from and write to IER. 1h: Allows access to the divisor latches of the baud generator during a read or write operation (DLL and DLH). At the address shared by RBR, THR, and DLL, the CPU can read from and write to DLL. At the address shared by IER and DLH, the CPU can read from and write to DLH.
6	BC	R/W	0h	Break Control. 0h: Break condition is disabled. 1h: Break condition is transmitted to the receiving UART. A break condition is a condition where the UARTn_TXD signal is forced to the spacing (cleared) state.

Table 6-582. PRUSS_UART_LINE_CONTROL_REGISTER Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	SP	R/W	0h	Stick parity. The SP bit works in conjunction with the EPS and PEN bits. The relationship between the SP, EPS, and PEN bits is summarized in Table 11-3498. 0h: Stick parity is disabled. 1h: Stick parity is enabled. • When odd parity is selected (EPS = 0), the PARITY bit is transmitted and checked as set. • When even parity is selected (EPS = 1), the PARITY bit is transmitted and checked as cleared.
4	EPS	R/W	0h	Even parity select. Selects the parity when parity is enabled (PEN = 1). The EPS bit works in conjunction with the SP and PEN bits. The relationship between the SP, EPS, and PEN bits is summarized in Table 11-3498. 0h: Odd parity is selected (an odd number of logic 1s is transmitted or checked in the data and PARITY bits). 1h: Even parity is selected (an even number of logic 1s is transmitted or checked in the data and PARITY bits).
3	PEN	R/W	0h	Parity enable. The PEN bit works in conjunction with the SP and EPS bits. The relationship between the SP, EPS, and PEN bits is summarized in Table 11-3498. 0h: No PARITY bit is transmitted or checked. 1h: Parity bit is generated in transmitted data and is checked in received data between the last data word bit and the first STOP bit.
2	STB	R/W	0h	Number of STOP bits generated. STB specifies 1, 1.5, or 2 STOP bits in each transmitted character. When STB = 1, the WLS bit determines the number of STOP bits. The receiver clocks only the first STOP bit, regardless of the number of STOP bits selected. The number of STOP bits generated is summarized in Table 11-3499. 0h: 1 STOP bit is generated. 1h: WLS bit determines the number of STOP bits: • When WLS = 0, 1.5 STOP bits are generated. • When WLS = 1h, 2h, or 3h, 2 STOP bits are generated.
1-0	WLS	R/W	0h	Word length select. Number of bits in each transmitted or received serial character. When STB = 1, the WLS bit determines the number of STOP bits. 0h: 5 bits 1h: 6 bits 2h: 7 bits 3h: 8 bits

Table 6-583. Register Call Summary for PRUSS_UART_LINE_CONTROL_REGISTER

PRU-ICSS UART Module
- PRU-ICSS_UART Registers: [0]
- PRUSS_UART_LINE_CONTROL_REGISTER Register (Offset = Ch) [reset = 0h]: [0]
- PRU-ICSS UART Transmission: [0]
- PRU-ICSS UART Initialization: [0]
- PRU-ICSS UART Reception: [0]
- PRU-ICSS UART FIFO Modes: [0]
- PRU-ICSS UART Data Format: [0][1][2]
- PRU-ICSS UART Transmission Protocol: [0]
- PRU-ICSS UART Reception Protocol: [0]

6.4.7.4.5 PRUSS_UART_MODEM_CONTROL_REGISTER Register (Offset = 10h) [reset = 0h]

PRUSS_UART_MODEM_CONTROL_REGISTER is shown in Figure 6-217 and described in Table 6-585.

Return to Summary Table.

The Modem control register provides the ability to enable/disable the autoflow functions, and enable/disable the loopback function for diagnostic purposes.

Table 6-584. PRUSS_UART_MODEM_CONTROL_REGISTER Instances

Instance	Physical Address
PRU_ICSS_0_UART	20AA 8010h
PRU_ICSS_1_UART	20AE 8010h

Figure 6-217. PRUSS_UART_MODEM_CONTROL_REGISTER Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		AFE	LOOP	OUT2	OUT1	RTS	RESERVED
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-585. PRUSS_UART_MODEM_CONTROL_REGISTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	AFE	R/W	0h	Autoflow control enable. Autoflow control allows the $\overline{\text{UARTn\_RTS}}$ and $\overline{\text{UARTn\_CTS}}$ signals to provide handshaking between UARTs during data transfer. When AFE = 1, the RTS bit determines the autoflow control enabled. Note that all UARTs do not support this feature, see your device-specific data manual for supported features. If this feature is not available, this bit is reserved in this device and should be cleared to 0. 0h: Autoflow control is disabled. 1h:Autoflow control is enabled: • When RTS = 0, $\overline{\text{UARTn\_CTS}}$ is only enabled. • When RTS = 1, $\overline{\text{UARTn\_RTS}}$ and $\overline{\text{UARTn\_CTS}}$ are enabled.

Table 6-585. PRUSS_UART_MODEM_CONTROL_REGISTER Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	LOOP	R/W	0h	Loop back mode enable. LOOP is used for the diagnostic testing using the loop back feature. 0h: Loop back mode is disabled. 1h: Loop back mode is enabled. When LOOP is set, the following occur: • The UARTn_TXD signal is set high. • The UARTn_RXD pin is disconnected. • The output of the transmitter shift register (TSR) is lopped back in to the receiver shift register (RSR) input.
3	OUT2	R/W	0h	OUT2 Control Bit
2	OUT1	R/W	0h	OUT1 Control Bit
1	RTS	R/W	0h	RTS control. When AFE = 1, the RTS bit determines the autoflow control enabled. Note that all UARTs do not support this feature, see your device-specific data manual for supported features. If this feature is not available, this bit is reserved in this device and should be cleared to 0. 0h: $\overline{\text{UARTn\_RTS}}$ is disabled, $\overline{\text{UARTn\_CTS}}$ is only enabled. 1h: $\overline{\text{UARTn\_RTS}}$ and $\overline{\text{UARTn\_CTS}}$ are enabled.
0	RESERVED	R	0h	Reserved

Table 6-586. Register Call Summary for PRUSS_UART_MODEM_CONTROL_REGISTER

PRU-ICSS UART Module
- PRUSS_UART_MODEM_CONTROL_REGISTER Register (Offset = 10h) [reset = 0h]: [0]
- PRU-ICSS_UART Registers: [0]
- PRU-ICSS UART Initialization: [0]
- PRU-ICSS UART Loopback Control: [0]

6.4.7.4.6 PRUSS_UART_LINE_STATUS_REGISTER Register (Offset = 14h) [reset = 60h]

PRUSS_UART_LINE_STATUS_REGISTER is shown in Figure 6-218 and described in Table 6-588.

Return to Summary Table.

The Line status register provides information to the CPU concerning the status of data transfers. Line status register is intended for read operations only; do not write to this register. Bits 1 through 4 record the error conditions that produce a receiver line status interrupt.

Table 6-587. PRUSS_UART_LINE_STATUS_REGISTER Instances

Instance	Physical Address
PRU_ICSS_0_UART	20AA 8014h
PRU_ICSS_1_UART	20AE 8014h

Figure 6-218. PRUSS_UART_LINE_STATUS_REGISTER Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RXFIFOE	TEMT	THRE	BI	FE	PE	OE	DR
R-0h	R-1h	R-1h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-588. PRUSS_UART_LINE_STATUS_REGISTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	RXFIFOE	R	0h	Receiver FIFO error. **In non-FIFO mode:** 0h: There has been no error, or RXFIFOE was cleared because the CPU read the erroneous character from the receiver buffer register (RBR). 1h: There is a parity error, framing error, or break indicator in the receiver buffer register (RBR).
				In FIFO mode: 0h: There has been no error, or RXFIFOE was cleared because the CPU read the erroneous character from the receiver FIFO and there are no more errors in the receiver FIFO. 1h: At least one parity error, framing error, or break indicator in the receiver FIFO.

Table 6-588. PRUSS_UART_LINE_STATUS_REGISTER Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	TEMT	R	1h	Transmitter empty (TEMT) indicator. **In non-FIFO mode:** 0h: Either the transmitter holding register (THR) or the transmitter shift register (TSR) contains a data character. 1h: Both the transmitter holding register (THR) and the transmitter shift register (TSR) are empty.
				In FIFO mode: 0h: Either the transmitter FIFO or the transmitter shift register (TSR) contains a data character. 1h: Both the transmitter FIFO and the transmitter shift register (TSR) are empty.
5	THRE	R	1h	Transmitter holding register empty (THRE) indicator. If the THRE bit is set and the corresponding interrupt enable bit is set (ETBEI = 1 in IER), an interrupt request is generated. **In non-FIFO mode:** 0h: Transmitter holding register (THR) is not empty. THR has been loaded by the CPU. 1h: Transmitter holding register (THR) is empty (ready to accept a new character). The content of THR has been transferred to the transmitter shift register (TSR).
				In FIFO mode: 0h: Transmitter FIFO is not empty. At least one character has been written to the transmitter FIFO. You can write to the transmitter FIFO if it is not full. 1h: Transmitter FIFO is empty. The last character in the FIFO has been transferred to the transmitter shift register (TSR).
4	BI	R	0h	Break indicator. The BI bit is set whenever the receive data input (UARTn_RXD) was held low for longer than a full-word transmission time. A full-word transmission time is defined as the total time to transmit the START, data, PARITY, and STOP bits. If the BI bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in IER), an interrupt request is generated. **In non-FIFO mode:** 0h: No break has been detected, or the BI bit was cleared because the CPU read the erroneous character from the receiver buffer register (RBR). 1h: A break has been detected with the character in the receiver buffer register (RBR).
				In FIFO mode: 0h: No break has been detected, or the BI bit was cleared because the CPU read the erroneous character from the receiver FIFO and the next character to be read from the FIFO has no break indicator. 1h: A break has been detected with the character at the top of the receiver FIFO.

Table 6-588. PRUSS_UART_LINE_STATUS_REGISTER Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	FE	R	0h	Framing error (FE) indicator. A framing error occurs when the received character does not have a valid STOP bit. In response to a framing error, the UART sets the FE bit and waits until the signal on the RX pin goes high. Once the RX signal goes high, the receiver is ready to detect a new START bit and receive new data. If the FE bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in IER), an interrupt request is generated. **In non-FIFO mode:** 0h: No framing error has been detected, or the FE bit was cleared because the CPU read the erroneous data from the receiver buffer register (RBR). 1h: A framing error has been detected with the character in the receiver buffer register (RBR).
				In FIFO mode: 0h: No framing error has been detected, or the FE bit was cleared because the CPU read the erroneous data from the receiver FIFO and the next character to be read from the FIFO has no framing error. 1h: A framing error has been detected with the character at the top of the receiver FIFO.
2	PE	R	0h	Parity error (PE) indicator. A parity error occurs when the parity of the received character does not match the parity selected with the EPS bit in the line control register (LCR). If the PE bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in IER), an interrupt request is generated. **In non-FIFO mode:** 0h: No parity error has been detected, or the PE bit was cleared because the CPU read the erroneous data from the receiver buffer register (RBR). 1h: A parity error has been detected with the character in the receiver buffer register (RBR).
				In FIFO mode: 0h: No parity error has been detected, or the PE bit was cleared because the CPU read the erroneous data from the receiver FIFO and the next character to be read from the FIFO has no parity error. 1h: A parity error has been detected with the character at the top of the receiver FIFO.
1	OE	R	0h	Overrun error (OE) indicator. An overrun error in the non-FIFO mode is different from an overrun error in the FIFO mode. If the OE bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in IER), an interrupt request is generated. **In non-FIFO mode:** 0h: No overrun error has been detected, or the OE bit was cleared because the CPU read the content of the line status register (LSR). 1h: Overrun error has been detected. Before the character in the receiver buffer register (RBR) could be read, it was overwritten by the next character arriving in RBR.
				In FIFO mode: 0h: No overrun error has been detected, or the OE bit was cleared because the CPU read the content of the line status register (LSR). 1h: Overrun error has been detected. If data continues to fill the FIFO beyond the trigger level, an overrun error occurs only after the FIFO is full and the next character has been completely received in the shift register. An overrun error is indicated to the CPU as soon as it happens. The new character overwrites the character in the shift register, but it is not transferred to the FIFO.

Table 6-588. PRUSS_UART_LINE_STATUS_REGISTER Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	DR	R	0h	Data-ready (DR) indicator for the receiver. If the DR bit is set and the corresponding interrupt enable bit is set (ERBI = 1 in IER), an interrupt request is generated. **In non-FIFO mode:** 0h: Data is not ready, or the DR bit was cleared because the character was read from the receiver buffer register (RBR). 1h: Data is ready. A complete incoming character has been received and transferred into the receiver buffer register (RBR).
				In FIFO mode: 0h: Data is not ready, or the DR bit was cleared because all of the characters in the receiver FIFO have been read. 1h: Data is ready. There is at least one unread character in the receiver FIFO. If the FIFO is empty, the DR bit is set as soon as a complete incoming character has been received and transferred into the FIFO. The DR bit remains set until the FIFO is empty again.

Table 6-589. Register Call Summary for PRUSS_UART_LINE_STATUS_REGISTER

PRU-ICSS UART Module

- PRUSS_UART_LINE_STATUS_REGISTER Register (Offset = 14h) [reset = 60h]: [0]
- PRU-ICSS_UART Registers: [0]
- PRU-ICSS UART FIFO Modes: [0][1][2][3][4][5][6][7][8][9][10]
- PRU-ICSS UART Interrupt Support: [0][1][2]
- PRU-ICSS UART Module Functional Description: [0]

6.4.7.4.7 PRUSS_UART_MODEM_STATUS_REGISTER Register (Offset = 18h) [reset = 0h]

PRUSS_UART_MODEM_STATUS_REGISTER is shown in Figure 6-219 and described in Table 6-591.

Return to Summary Table.

The Modem status register provides information to the CPU concerning the status of modem control signals. Modem status register is intended for read operations only; do not write to this register.

Table 6-590. PRUSS_UART_MODEM_STATUS_REGISTER Instances

Instance	Physical Address
PRU_ICSS_0_UART	20AA 8018h
PRU_ICSS_1_UART	20AE 8018h

Figure 6-219. PRUSS_UART_MODEM_STATUS_REGISTER Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CD	RI	DSR	CTS	DCD	TERI	DDSR	DCTS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -*n* = value after reset

Table 6-591. PRUSS_UART_MODEM_STATUS_REGISTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	CD	R	0h	Complement of the Carrier Detect input. When the UART is in the diagnostic test mode (loopback mode MCR[4] = 1), this bit is equal to the MCR bit 3 (OUT2)..
6	RI	R	0h	Complement of the Ring Indicator input. When the UART is in the diagnostic test mode (loopback mode MCR[4] = 1), this bit is equal to the MCR bit 2 (OUT1).
5	DSR	R	0h	Complement of the Data Set Ready input. When the UART is in the diagnostic test mode (loopback mode MCR[4] = 1), this bit is equal to the MCR bit 0 (DTR).
4	CTS	R	0h	Complement of the Clear To Send input. When the UART is in the diagnostic test mode (loopback mode MCR[4] = 1), this bit is equal to the MCR bit 1 (RTS).
3	DCD	R	0h	Change in DCD indicator bit. DCD indicates that the DCD input has changed state since the last time it was read by the CPU. When DCD is set and the modem status interrupt is enabled, a modem status interrupt is generated.
2	TERI	R	0h	Trailing edge of RI (TERI) indicator bit. TERI indicates that the RI input has changed from a low to a high. When TERI is set and the modem status interrupt is enabled, a modem status interrupt is generated.

Table 6-591. PRUSS_UART_MODEM_STATUS_REGISTER Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	DDSR	R	0h	Change in DSR indicator bit. DDSR indicates that the DSR input has changed state since the last time it was read by the CPU. When DDSR is set and the modem status interrupt is enabled, a modem status interrupt is generated.
0	DCTS	R	0h	Change in CTS indicator bit. DCTS indicates that the CTS input has changed state since the last time it was read by the CPU. When DCTS is set (autoflow control is not enabled and the modem status interrupt is enabled), a modem status interrupt is generated. When autoflow control is enabled, no interrupt is generated.

Table 6-592. Register Call Summary for PRUSS_UART_MODEM_STATUS_REGISTER

PRU-ICSS UART Module
- PRU-ICSS_UART Registers: [0]
- PRUSS_UART_MODEM_STATUS_REGISTER Register (Offset = 18h) [reset = 0h]: [0]

6.4.7.4.8 PRUSS_UART_SCRATCH_REGISTER Register (Offset = 1Ch) [reset = 0h]

PRUSS_UART_SCRATCH_REGISTER is shown in Figure 6-220 and described in Table 6-594.

Return to Summary Table.

The Scratch Pad register is intended for programmer's use as a scratch pad. It temporarily holds the programmer's data without affecting UART operation.

Table 6-593. PRUSS_UART_SCRATCH_REGISTER Instances

Instance	Physical Address
PRU_ICSS_0_UART	20AA 801Ch
PRU_ICSS_1_UART	20AE 801Ch

Figure 6-220. PRUSS_UART_SCRATCH_REGISTER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DATA							
R-0h																								R/W-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-594. PRUSS_UART_SCRATCH_REGISTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	DATA	R/W	0h	These bits are intended for the programmer's use as a scratch pad in the sense that it temporarily holds the programmer's data without affecting any other UART operation.

Table 6-595. Register Call Summary for PRUSS_UART_SCRATCH_REGISTER

PRU-ICSS UART Module

- PRUSS_UART_SCRATCH_REGISTER Register (Offset = 1Ch) [reset = 0h]: [0]
- PRU-ICSS_UART Registers: [0]

6.4.7.4.9 PRUSS_UART_DIVISOR_REGISTER_LSB_ Register (Offset = 20h) [reset = 0h]

PRUSS_UART_DIVISOR_REGISTER_LSB_ is shown in Figure 6-221 and described in Table 6-597.

Return to Summary Table.

Two 8-bit register fields (DLL and DLH), called divisor latches, store the 16-bit divisor for generation of the baud clock in the baud generator. DLH holds the most-significant bits of the divisor, and DLL holds the least-significant bits of the divisor. These divisor latches must be loaded during initialization of the UART in order to ensure desired operation of the baud generator. Writing to the divisor latches results in two wait states being inserted during the write access while the baud generator is loaded with the new value.

Table 6-596. PRUSS_UART_DIVISOR_REGISTER_LSB_ Instances

Instance	Physical Address
PRU_ICSS_0_UART	20AA 8020h
PRU_ICSS_1_UART	20AE 8020h

Figure 6-221. PRUSS_UART_DIVISOR_REGISTER_LSB_ Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DLL							
R-0h																								R/W-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-597. PRUSS_UART_DIVISOR_REGISTER_LSB_ Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	DLL	R/W	0h	The 8 least-significant bits (LSBs) of the 16-bit divisor for generation of the baud clock in the baud rate generator.

Table 6-598. Register Call Summary for PRUSS_UART_DIVISOR_REGISTER_LSB_

PRU-ICSS UART Module

- PRUSS_UART_DIVISOR_REGISTER_LSB_ Register (Offset = 20h) [reset = 0h]: [0]
- PRU-ICSS UART Clock Generation and Control: [0]
- PRU-ICSS UART Initialization: [0]
- PRU-ICSS_UART Registers: [0]

6.4.7.4.10 PRUSS_UART_DIVISOR_REGISTER_MSB_ Register (Offset = 24h) [reset = 0h]

PRUSS_UART_DIVISOR_REGISTER_MSB_ is shown in Figure 6-222 and described in Table 6-600.

Return to Summary Table.

Two 8-bit register fields (DLL and DLH), called divisor latches, store the 16-bit divisor for generation of the baud clock in the baud generator. DLH holds the most-significant bits of the divisor, and DLL holds the least-significant bits of the divisor. These divisor latches must be loaded during initialization of the UART in order to ensure desired operation of the baud generator. Writing to the divisor latches results in two wait states being inserted during the write access while the baud generator is loaded with the new value.

Table 6-599. PRUSS_UART_DIVISOR_REGISTER_MSB_ Instances

Instance	Physical Address
PRU_ICSS_0_UART	20AA 8024h
PRU_ICSS_1_UART	20AE 8024h

Figure 6-222. PRUSS_UART_DIVISOR_REGISTER_MSB_ Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DLH							
R-0h																								R/W-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-600. PRUSS_UART_DIVISOR_REGISTER_MSB_ Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	DLH	R/W	0h	The 8 most-significant bits (MSBs) of the 16-bit divisor for generation of the baud clock in the baud rate generator.

Table 6-601. Register Call Summary for PRUSS_UART_DIVISOR_REGISTER_MSB_

PRU-ICSS UART Module

- PRU-ICSS UART Clock Generation and Control: [0]
- PRU-ICSS UART Initialization: [0]
- PRU-ICSS_UART Registers: [0]
- PRUSS_UART_DIVISOR_REGISTER_MSB_ Register (Offset = 24h) [reset = 0h]: [0]

6.4.7.4.11 PRUSS_UART_PERIPHERAL_ID_REGISTER Register (Offset = 28h) [reset = 44141102h]

PRUSS_UART_PERIPHERAL_ID_REGISTER is shown in Figure 6-223 and described in Table 6-603.

Return to Summary Table.

Peripheral Identification register.

Table 6-602. PRUSS_UART_PERIPHERAL_ID_REGISTER Instances

Instance	Physical Address
PRU_ICSS_0_UART	20AA 8028h
PRU_ICSS_1_UART	20AE 8028h

Figure 6-223. PRUSS_UART_PERIPHERAL_ID_REGISTER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PID																															
R-44141102h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-603. PRUSS_UART_PERIPHERAL_ID_REGISTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PID	R	44141102h	

Table 6-604. Register Call Summary for PRUSS_UART_PERIPHERAL_ID_REGISTER

PRU-ICSS UART Module
- PRU-ICSS_UART Registers: [0]
- PRUSS_UART_PERIPHERAL_ID_REGISTER Register (Offset = 28h) [reset = 44141102h]: [0]

6.4.7.4.12 PRUSS_UART_POWERMANAGEMENT_AND_EMULATION_REGISTER Register (Offset = 30h) [reset = 0h]

PRUSS_UART_POWERMANAGEMENT_AND_EMULATION_REGISTER is shown in Figure 6-224 and described in Table 6-606.

Return to Summary Table.

Power and emulation management register.

Table 6-605. PRUSS_UART_POWERMANAGEMENT_AND_EMULATION_REGISTER Instances

Instance	Physical Address
PRU_ICSS_0_UART	20AA 8030h
PRU_ICSS_1_UART	20AE 8030h

Figure 6-224. PRUSS_UART_POWERMANAGEMENT_AND_EMULATION_REGISTER Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	UTRST	URRST	RESERVED				
R/W-0h	R/W-0h	R/W-0h	R-0h				
7	6	5	4	3	2	1	0
RESERVED							FREE
R-0h							R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-606. PRUSS_UART_POWERMANAGEMENT_AND_EMULATION_REGISTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	RESERVED	R/W	0h	Reserved. This bit must always be written with a 0.
14	UTRST	R/W	0h	UART transmitter reset. Resets and enables the transmitter. 0h: Transmitter is disabled and in reset state. 1h: Transmitter is enabled.
13	URRST	R/W	0h	UART receiver reset. Resets and enables the receiver. 0h: Receiver is disabled and in reset state. 1h: Receiver is enabled.
12-1	RESERVED	R	0h	Reserved
0	FREE	R/W	0h	Free-running enable mode bit. This bit determines the emulation mode functionality of the UART. When halted, the UART can handle register read/write requests, but does not generate any transmission/reception, interrupts or events. 0h: If a transmission is not in progress, the UART halts immediately. If a transmission is in progress, the UART halts after completion of the one-word transmission. 1h: Free-running mode is enabled; UART continues to run normally.

Table 6-607. Register Call Summary for PRUSS_UART_POWERMANAGEMENT_AND_EMULATION_REGISTER

PRU-ICSS UART Module

- PRU-ICSS_UART Registers: [0]
- PRU-ICSS UART Initialization: [0]
- PRU-ICSS UART Software Reset Considerations: [0]
- PRU-ICSS UART DMA Event Support: [0]
- PRU-ICSS UART FIFO Modes: [0]
- PRU-ICSS UART Module Functional Description: [0]
- PRUSS_UART_POWERMANAGEMENT_AND_EMULATION_REGISTER Register (Offset = 30h) [reset = 0h]: [0]

6.4.7.4.13 PRUSS_UART_MODE_DEFINITION_REGISTER Register (Offset = 34h) [reset = 0h]

PRUSS_UART_MODE_DEFINITION_REGISTER is shown in Figure 6-225 and described in Table 6-609.

Return to Summary Table.

The Mode definition register determines the over-sampling mode for the UART.

Table 6-608. PRUSS_UART_MODE_DEFINITION_REGISTER Instances

Instance	Physical Address
PRU_ICSS_0_UART	20AA 8034h
PRU_ICSS_1_UART	20AE 8034h

Figure 6-225. PRUSS_UART_MODE_DEFINITION_REGISTER Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							OSM_SEL
R-0h							R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-609. PRUSS_UART_MODE_DEFINITION_REGISTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	OSM_SEL	R/W	0h	Over-Sampling Mode Select. 0h: 16× over-sampling. 1h: 13× over-sampling.

Table 6-610. Register Call Summary for PRUSS_UART_MODE_DEFINITION_REGISTER

PRU-ICSS UART Module
- PRU-ICSS UART Clock Generation and Control: [0]
- PRU-ICSS_UART Registers: [0]
- PRUSS_UART_MODE_DEFINITION_REGISTER Register (Offset = 34h) [reset = 0h]: [0]

6.4.8 PRU-ICSS eCAP Module

6.4.8.1 PRU-ICSS eCAP Functional Description

A single instance of an **enhanced capture** event module is integrated in the device PRU-ICSS_0 subsystem - ICSS_0_eCAP0 and PRU-ICSS_1 subsystem - ICSS_1_eCAP0.

For more details on the ICSS_0_eCAP0 and ICSS_1_eCAP0I/O signals available at device level, refer to the Section 6.4.2. For ICSS_0_eCAP0 and ICSS_1_eCAP0 integration details and functionalities controlled at PRU-ICSS top level (functional clock control, etc.), refer to the Section 6.4.3 and the Section 6.4.4 .

NOTE: The ICSS_0_eCAP0 and ICSS_1_eCAP0 "SYNCIn" hardware event synchronization input and "SYNCOut" hardware synchronization output are implemented in the device PRU-ICSS_0 and PRU-ICSS_1, respectively. However, a software-forced synchronization via bit PRUSS_ECAP_ECCTL2[8] SWSYNC, can be used as an alternative, provided that PRUSS_ECAP_ECCTL2[5] SYNCI_EN bit is set to 0b1.

For full description of the ICSS_0_eCAP0 and ICSS_1_eCAP0 modules functionalities, refer to the Section 11.4, *Enhanced Capture (eCAP) Module.*

6.4.8.2 PRU-ICSS_ECAP Registers

Table 6-612 lists the memory-mapped registers for the PRU-ICSS_eCAP0 module. All register offset addresses not listed in Table 6-612 should be considered as reserved locations and the register contents should not be modified.

Table 6-611. PRU-ICSS_ECAP Instances

Instance	Base Address
PRU_ICSS_0_ECAP	20AB 0000h
PRU_ICSS_1_ECAP	20AF 0000h

Table 6-612. PRU-ICSS_ECAP Registers

Offset	Acronym	Register Name	PRU_ICSS_0_ECAP Physical Address	PRU_ICSS_1_ECAP Physical Address	Section
0h	PRUSS_ECAP_TSCNT	Time Stamp Counter Register	20AB 0000h	20AF 0000h	Section 6.4.8.2.1
4h	PRUSS_ECAP_CNTPHS	Counter Phase Control Register	20AB 0004h	20AF 0004h	Section 6.4.8.2.2
8h	PRUSS_ECAP_CAP1	Capture-1 Register	20AB 0008h	20AF 0008h	Section 6.4.8.2.3
Ch	PRUSS_ECAP_CAP2	Capture-2 Register	20AB 000Ch	20AF 000Ch	Section 6.4.8.2.4
10h	PRUSS_ECAP_CAP3	Capture-3 Register	20AB 0010h	20AF 0010h	Section 6.4.8.2.5
14h	PRUSS_ECAP_CAP4	Capture-4 Register	20AB 0014h	20AF 0014h	Section 6.4.8.2.6
28h	PRUSS_ECAP_ECCTL1	ECAP Control Register1	20AB 0028h	20AF 0028h	Section 6.4.8.2.7
2Ah	PRUSS_ECAP_ECCTL2	ECAP Control Register 2	20AB 002Ah	20AF 002Ah	Section 6.4.8.2.8
2Ch	PRUSS_ECAP_ECEINT	ECAP Interrupt Enable Register	20AB 002Ch	20AF 002Ch	Section 6.4.8.2.9
2Eh	PRUSS_ECAP_ECFLG	ECAP Interrupt Flag Register	20AB 002Eh	20AF 002Eh	Section 6.4.8.2.10
30h	PRUSS_ECAP_ECCLR	ECAP Interrupt Clear Register	20AB 0030h	20AF 0030h	Section 6.4.8.2.11
34h	PRUSS_ECAP_ECFRC	ECAP Interrupt Forcing Register	20AB 0034h	20AF 0034h	Section 6.4.8.2.12
5Ch	PRUSS_ECAP_PID	ECAP Revision ID	20AB 005Ch	20AF 005Ch	Section 6.4.8.2.13

6.4.8.2.1 PRUSS_ECAP_TSCNT Register (Offset = 0h) [reset = 0h]

PRUSS_ECAP_TSCNT is shown in Figure 6-226 and described in Table 6-614.

Return to Summary Table.

Time Stamp Counter Register

Table 6-613. PRUSS_ECAP_TSCNT Instances

Instance	Physical Address
PRU_ICSS_0_ECAP	20AB 0000h
PRU_ICSS_1_ECAP	20AF 0000h

Figure 6-226. PRUSS_ECAP_TSCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSCNT																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-614. PRUSS_ECAP_TSCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TSCNT	R/W	0h	Active 32 bit-counter register that is used as the capture time-base

Table 6-615. Register Call Summary for PRUSS_ECAP_TSCNT

PRU-ICSS eCAP Module

- PRUSS_ECAP_TSCNT Register (Offset = 0h) [reset = 0h]: [0]
- PRUSS_ECAP_CNTPHS Register (Offset = 4h) [reset = 0h]: [0]
- PRU-ICSS_ECAP Registers: [0]

6.4.8.2.2 PRUSS_ECAP_CNTPHS Register (Offset = 4h) [reset = 0h]

PRUSS_ECAP_CNTPHS is shown in Figure 6-227 and described in Table 6-617.

Return to Summary Table.

Counter Phase Control Register

Table 6-616. PRUSS_ECAP_CNTPHS Instances

Instance	Physical Address
PRU_ICSS_0_ECAP	20AB 0004h
PRU_ICSS_1_ECAP	20AF 0004h

Figure 6-227. PRUSS_ECAP_CNTPHS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNTPHS																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-617. PRUSS_ECAP_CNTPHS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CNTPHS	R/W	0h	Counter phase value register that can be programmed for phase lag/lead. This register shadows TSCNT and is loaded into PRUSS_ECAP_TSCNT upon either a SYNCI event or S/W force via a control bit. Used to achieve phase control synchronization with respect to other eCAP and EPWM time-bases.

Table 6-618. Register Call Summary for PRUSS_ECAP_CNTPHS

PRU-ICSS eCAP Module

- PRUSS_ECAP_CNTPHS Register (Offset = 4h) [reset = 0h]: [0]
- PRU-ICSS_ECAP Registers: [0]
- PRUSS_ECAP_ECCTL2 Register (Offset = 2Ah) [reset = 6h]: [0]

6.4.8.2.3 PRUSS_ECAP_CAP1 Register (Offset = 8h) [reset = 0h]

PRUSS_ECAP_CAP1 is shown in Figure 6-228 and described in Table 6-620.

Return to Summary Table.

Capture-1 Register

Table 6-619. PRUSS_ECAP_CAP1 Instances

Instance	Physical Address
PRU_ICSS_0_ECAP	20AB 0008h
PRU_ICSS_1_ECAP	20AF 0008h

Figure 6-228. PRUSS_ECAP_CAP1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP1																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-620. PRUSS_ECAP_CAP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP1	R/W	0h	This register can be loaded (written) by the following. (a) Time-Stamp (that is, counter value) during a capture event. (b) Software may be useful for test purposes. (c) APRD active register when used in APWM mode.

Table 6-621. Register Call Summary for PRUSS_ECAP_CAP1

PRU-ICSS eCAP Module

- PRUSS_ECAP_ECCTL1 Register (Offset = 28h) [reset = 0h]: [0][1][2]
- PRUSS_ECAP_ECCTL2 Register (Offset = 2Ah) [reset = 6h]: [0][1][2][3]
- PRUSS_ECAP_CAP1 Register (Offset = 8h) [reset = 0h]: [0]
- PRU-ICSS_ECAP Registers: [0]

6.4.8.2.4 PRUSS_ECAP_CAP2 Register (Offset = Ch) [reset = 0h]

PRUSS_ECAP_CAP2 is shown in Figure 6-229 and described in Table 6-623.

Return to Summary Table.

Capture-2 Register

Table 6-622. PRUSS_ECAP_CAP2 Instances

Instance	Physical Address
PRU_ICSS_0_ECAP	20AB 000Ch
PRU_ICSS_1_ECAP	20AF 000Ch

Figure 6-229. PRUSS_ECAP_CAP2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP2																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-623. PRUSS_ECAP_CAP2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP2	R/W	0h	This register can be loaded (written) by the following. (a) Time-Stamp (that is, counter value) during a capture event. (b) Software may be useful for test purposes. (c) APRD active register when used in APWM mode.

Table 6-624. Register Call Summary for PRUSS_ECAP_CAP2

PRU-ICSS eCAP Module

- PRU-ICSS_ECAP Registers: [0]
- PRUSS_ECAP_CAP2 Register (Offset = Ch) [reset = 0h]: [0]
- PRUSS_ECAP_ECCTL2 Register (Offset = 2Ah) [reset = 6h]: [0][1]

6.4.8.2.5 PRUSS_ECAP_CAP3 Register (Offset = 10h) [reset = 0h]

PRUSS_ECAP_CAP3 is shown in Figure 6-230 and described in Table 6-626.

Return to Summary Table.

Capture-3 Register

Table 6-625. PRUSS_ECAP_CAP3 Instances

Instance	Physical Address
PRU_ICSS_0_ECAP	20AB 0010h
PRU_ICSS_1_ECAP	20AF 0010h

Figure 6-230. PRUSS_ECAP_CAP3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP3																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-626. PRUSS_ECAP_CAP3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP3	R/W	0h	In CMP mode, this is a time-stamp capture register. In APWM mode, this is the period shadow (APRD) register. User software updates the PWM period value through this register. In this mode, CAP3 shadows CAP1.

Table 6-627. Register Call Summary for PRUSS_ECAP_CAP3

PRU-ICSS eCAP Module

- PRU-ICSS_ECAP Registers: [0]
- PRUSS_ECAP_CAP3 Register (Offset = 10h) [reset = 0h]: [0]

6.4.8.2.6 PRUSS_ECAP_CAP4 Register (Offset = 14h) [reset = 0h]

PRUSS_ECAP_CAP4 is shown in Figure 6-231 and described in Table 6-629.

Return to Summary Table.

Capture-4 Register

Table 6-628. PRUSS_ECAP_CAP4 Instances

Instance	Physical Address
PRU_ICSS_0_ECAP	20AB 0014h
PRU_ICSS_1_ECAP	20AF 0014h

Figure 6-231. PRUSS_ECAP_CAP4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP4																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-629. PRUSS_ECAP_CAP4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP4	R/W	0h	In CMP mode, this is a time-stamp capture register. In APWM mode, this is the compare shadow (ACMP) register. User software updates the PWM compare value through this register. In this mode, CAP4 shadows CAP2.

Table 6-630. Register Call Summary for PRUSS_ECAP_CAP4

PRU-ICSS eCAP Module

- PRUSS_ECAP_CAP4 Register (Offset = 14h) [reset = 0h]: [0]
- PRU-ICSS_ECAP Registers: [0]
- PRUSS_ECAP_ECCTL2 Register (Offset = 2Ah) [reset = 6h]: [0][1]
- PRUSS_ECAP_ECCTL1 Register (Offset = 28h) [reset = 0h]: [0][1][2]

6.4.8.2.7 PRUSS_ECAP_ECCTL1 Register (Offset = 28h) [reset = 0h]

PRUSS_ECAP_ECCTL1 is shown in Figure 6-232 and described in Table 6-632.

Return to Summary Table.

ECAP Control Register1

Table 6-631. PRUSS_ECAP_ECCTL1 Instances

Instance	Physical Address
PRU_ICSS_0_ECAP	20AB 0028h
PRU_ICSS_1_ECAP	20AF 0028h

Figure 6-232. PRUSS_ECAP_ECCTL1 Register

15	14	13	12	11	10	9	8
FREE_SOFT		EVTFLTPS					CAPLDEN
R/W-0h		R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
CTRRST4	CAP4POL	CTRRST3	CAP3POL	CTRRST2	CAP2POL	CTRRST1	CAP1POL
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-632. PRUSS_ECAP_ECCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	FREE_SOFT	R/W	0h	Emulation Control 0h: TSCNT counter stops immediately on emulation suspend. 1h: TSCNT counter runs until = 0. 2h: TSCNT counter is unaffected by emulation suspend (Run Free). 3h: TSCNT counter is unaffected by emulation suspend (Run Free).
13-9	EVTFLTPS	R/W	0h	Event Filter prescale select: 0h: Divide by 1 (i.e,. no prescale, by-pass the prescaler) 1h: Divide by 2 2h: Divide by 4 3h: Divide by 6 4h: Divide by 8 5h: Divide by 10 1Eh: Divide by 60 1Fh: Divide by 62
8	CAPLDEN	R/W	0h	Enable Loading of PRUSS_ECAP_CAP1 to PRUSS_ECAP_CAP4 registers on a capture event 0h: Disable PRUSS_ECAP_CAP1-PRUSS_ECAP_CAP4 register loads at capture event time. 1h: Enable PRUSS_ECAP_CAP1-PRUSS_ECAP_CAP4 register loads at capture event time.
7	CTRRST4	R/W	0h	Counter Reset on Capture Event 4 0h: Do not reset counter on Capture Event 4 (absolute time stamp operation) 1h: Reset counter after Capture Event 4 time-stamp has been captured (used in difference mode operation)
6	CAP4POL	R/W	0h	Capture Event 4 Polarity select 0h: Capture Event 4 triggered on a rising edge (RE) 1h: Capture Event 4 triggered on a falling edge (FE)

Table 6-632. PRUSS_ECAP_ECCTL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	CTRRST3	R/W	0h	Counter Reset on Capture Event 3 0h: Do not reset counter on Capture Event 3 (absolute time stamp) 1h: Reset counter after Event 3 time-stamp has been captured (used in difference mode operation)
4	CAP3POL	R/W	0h	Capture Event 3 Polarity select 0h: Capture Event 3 triggered on a rising edge (RE) 1h: Capture Event 3 triggered on a falling edge (FE)
3	CTRRST2	R/W	0h	Counter Reset on Capture Event 2 0h: Do not reset counter on Capture Event 2 (absolute time stamp) 1h: Reset counter after Event 2 time-stamp has been captured (used in difference mode operation)
2	CAP2POL	R/W	0h	Capture Event 2 Polarity select 0h: Capture Event 2 triggered on a rising edge (RE) 1h: Capture Event 2 triggered on a falling edge (FE)
1	CTRRST1	R/W	0h	Counter Reset on Capture Event 1 0h: Do not reset counter on Capture Event 1 (absolute time stamp) 1h: Reset counter after Event 1 time-stamp has been captured (used in difference mode operation)
0	CAP1POL	R/W	0h	Capture Event 1 Polarity select 0h: Capture Event 1 triggered on a rising edge (RE) 1h: Capture Event 1 triggered on a falling edge (FE)

Table 6-633. Register Call Summary for PRUSS_ECAP_ECCTL1

PRU-ICSS eCAP Module
- PRU-ICSS_ECAP Registers: [0]
- PRUSS_ECAP_ECCTL1 Register (Offset = 28h) [reset = 0h]: [0]

6.4.8.2.8 PRUSS_ECAP_ECCTL2 Register (Offset = 2Ah) [reset = 6h]

PRUSS_ECAP_ECCTL2 is shown in Figure 6-233 and described in Table 6-635.

Return to Summary Table.

ECAP Control Register 2

Table 6-634. PRUSS_ECAP_ECCTL2 Instances

Instance	Physical Address
PRU_ICSS_0_ECAP	20AB 002Ah
PRU_ICSS_1_ECAP	20AF 002Ah

Figure 6-233. PRUSS_ECAP_ECCTL2 Register

15	14	13	12	11	10	9	8
RESERVED					APWMPOL	CAPAPWM	SWSYNC
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
SYNCO_SEL		SYNCI_EN	TSCNTSTP	REARMRESET	STOPVALUE		CONTONESHT
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-3h		R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-635. PRUSS_ECAP_ECCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10	APWMPOL	R/W	0h	APWM output polarity select. This is applicable only in APWM operating mode 0h: Output is active high (Compare value defines high time) 1h: Output is active low (Compare value defines low time)
9	CAPAPWM	R/W	0h	CAP/APWM operating mode select 0h: ECAP module operates in capture mode. This mode forces the following configuration. (a) Inhibits TSCNT resets via CTR = PRD event. (b) Inhibits shadow loads on PRUSS_ECAP_CAP1 and PRUSS_ECAP_CAP2 registers. (c) Permits user to enable PRUSS_ECAP_CAP1-PRUSS_ECAP_CAP4 register load. (d) ECAP input/APWM output pin operates as a capture input. 1h: ECAP module operates in APWM mode. This mode forces the following configuration. (a) Resets TSCNT on CTR = PRD event (period boundary). (b) Permits shadow loading on PRUSS_ECAP_CAP1 and PRUSS_ECAP_CAP2 registers. (c) Disables loading of time-stamps into PRUSS_ECAP_CAP1 - PRUSS_ECAP_CAP4 registers. (d) ECAP input/APWM ouput pin operates as a APWM output.

Table 6-635. PRUSS_ECAP_ECCTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	SWSYNC	R/W	0h	Software-forced Counter (TSCNT) Synchronizing. This provides a convenient software method to synchronize some or all ECAP time bases. In APWM mode, the synchronizing can also be done via the CTR = PRD event. Note: Selection CTR = PRD is meaningful only in APWM mode. However, you can choose it in CAP mode if you find doing so useful. 0h: Writing a zero has no effect. Reading always returns a zero 1h: Writing a one forces a TSCNT shadow load of current ECAP module and any ECAP modules down-stream providing the SYNCO_SEL bits are 1'b00. After writing a 1, this bit returns to a zero.
7-6	SYNCO_SEL	R/W	0h	Sync-Out Select 0h: Select sync-in event to be the sync-out signal (pass through) 1h: Select CTR = PRD event to be the sync-out signal 2h: Disable sync out signal 3h: Disable sync out signal
5	SYNCI_EN	R/W	0h	Counter (TSCNT) Sync-In select mode 0h: Disable sync-in option 1h: Enable counter (TSCNT) to be loaded from PRUSS_ECAP_CNTPHS register upon either a SYNCI signal or a S/W force event.
4	TSCNTSTP	R/W	0h	Time Stamp (TSCNT) Counter Stop (freeze) Control 0h: TSCNT stopped 1h: TSCNT free-running
3	REARMRESET	R/W	0h	One-Shot Re-Arming Control, that is, wait for stop trigger. Note: The re-arm function is valid in one shot or continuous mode. 0h: Has no effect (reading always returns a 0) 1h: Arms the one-shot sequence as follows: 1) Resets the Mod4 counter to zero. 2) Unfreezes the Mod4 counter. 3) Enables capture register loads.
2-1	STOPVALUE	R/W	3h	Stop value for one-shot mode. This is the number (between 1 and 4) of captures allowed to occur before the CAP (1 through 4) registers are frozen, that is, capture sequence is stopped. Wrap value for continuous mode. This is the number (between 1 and 4) of the capture register in which the circular buffer wraps around and starts again. Notes: STOPVALUE is compared to Mod4 counter and, when equal, the following two actions occur. (1) Mod4 counter is stopped (frozen). (2) Capture register loads are inhibited. In one-shot mode, further interrupt events are blocked until re-armed. 0h: Stop after Capture Event 1 in one-shot mode. Wrap after Capture Event 1 in continuous mode. 1h: Stop after Capture Event 2 in one-shot mode. Wrap after Capture Event 2 in continuous mode. 2h: Stop after Capture Event 3 in one-shot mode. Wrap after Capture Event 3 in continuous mode. 3h: Stop after Capture Event 4 in one-shot mode. Wrap after Capture Event 4 in continuous mode.
0	CONTONESHT	R/W	0h	Continuous or one-shot mode control (applicable only in capture mode) 0h: Operate in continuous mode 1h: Operate in one-shot mode

Table 6-636. Register Call Summary for PRUSS_ECAP_ECCTL2

PRU-ICSS eCAP Module
• PRU-ICSS eCAP Functional Description: [0][1] • PRU-ICSS_ECAP Registers: [0] • PRUSS_ECAP_ECCTL2 Register (Offset = 2Ah) [reset = 6h]: [0]

6.4.8.2.9 PRUSS_ECAP_ECEINT Register (Offset = 2Ch) [reset = 0h]

PRUSS_ECAP_ECEINT is shown in Figure 6-234 and described in Table 6-638.

Return to Summary Table.

ECAP Interrupt Enable Register

Table 6-637. PRUSS_ECAP_ECEINT Instances

Instance	Physical Address
PRU_ICSS_0_ECAP	20AB 002Ch
PRU_ICSS_1_ECAP	20AF 002Ch

Figure 6-234. PRUSS_ECAP_ECEINT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CMPEQ	PRDEQ	CNTOVF	CEVT4	CEVT3	CEVT2	CEVT1	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-638. PRUSS_ECAP_ECEINT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	CMPEQ	R/W	0h	Counter Equal **Compare** Interrupt Enable. 0h: Disable Compare Equal as an Interrupt source. 1h: Enable Compare Equal as an Interrupt source.
6	PRDEQ	R/W	0h	Counter Equal **Period** Interrupt Enable. 0h: Disable Period Equal as an Interrupt source. 1h: Enable Period Equal as an Interrupt source.
5	CNTOVF	R/W	0h	Counter Overflow Interrupt Enable. 0h: Disable counter Overflow as an Interrupt source. 1h: Enable counter Overflow as an Interrupt source.
4	CEVT4	R/W	0h	Capture Event 4 Interrupt Enable. 0h: Disable Capture Event 4 as an Interrupt source. 1h: Enable Capture Event 4 as an Interrupt source.
3	CEVT3	R/W	0h	Capture Event 3 Interrupt Enable. 0h: Disable Capture Event 3 as an Interrupt source. 1h: Enable Capture Event 3 as an Interrupt source.
2	CEVT2	R/W	0h	Capture Event 2 Interrupt Enable. 0h: Disable Capture Event 2 as an Interrupt source. 1h: Enable Capture Event 2 as an Interrupt source.
1	CEVT1	R/W	0h	Capture Event 1 Interrupt Enable . 0h: Disable Capture Event 1 as an Interrupt source. 1h: Enable Capture Event 1 as an Interrupt source.
0	RESERVED	R	0h	Reserved

Table 6-639. Register Call Summary for PRUSS_ECAP_ECEINT

PRU-ICSS eCAP Module
• PRU-ICSS_ECAP Registers: [0] • PRUSS_ECAP_ECEINT Register (Offset = 2Ch) [reset = 0h]: [0]

6.4.8.2.10 PRUSS_ECAP_ECFLG Register (Offset = 2Eh) [reset = 0h]

PRUSS_ECAP_ECFLG is shown in Figure 6-235 and described in Table 6-641.

Return to Summary Table.

ECAP Interrupt Flag Register

Table 6-640. PRUSS_ECAP_ECFLG Instances

Instance	Physical Address
PRU_ICSS_0_ECAP	20AB 002Eh
PRU_ICSS_1_ECAP	20AF 002Eh

Figure 6-235. PRUSS_ECAP_ECFLG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CMPEQ	PRDEQ	CNTOVF	CEVT4	CEVT3	CEVT2	CEVT1	INT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-641. PRUSS_ECAP_ECFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	CMPEQ	R	0h	Compare Equal Compare Status Flag. This flag is only active in APWM mode. 0h: Indicates no event occurred 1h: Indicates the counter (TSCNT) reached the compare register value (ACMP)
6	PRDEQ	R	0h	Counter Equal Period Status Flag. This flag is only active in APWM mode. 0h: Indicates no event occurred 1h: Indicates the counter (TSCNT) reached the period register value (APRD) and was reset.
5	CNTOVF	R	0h	Counter Overflow Status Flag. This flag is active in CAP and APWM mode. 0h: Indicates no event occurred. 1h: Indicates the counter (TSCNT) has made the transition from FFFFFFFFh to 00000000h
4	CEVT4	R	0h	Capture Event 4 Status Flag This flag is only active in CAP mode. 0h: Indicates no event occurred 1h: Indicates the fourth event occurred at ECAPn pin
3	CEVT3	R	0h	Capture Event 3 Status Flag. This flag is active only in CAP mode. 0h: Indicates no event occurred. 1h: Indicates the third event occurred at ECAPn pin.
2	CEVT2	R	0h	Capture Event 2 Status Flag. This flag is only active in CAP mode. 0h: Indicates no event occurred. 1h: Indicates the second event occurred at ECAPn pin.
1	CEVT1	R	0h	Capture Event 1 Status Flag. This flag is only active in CAP mode. 0h: Indicates no event occurred. 1h: Indicates the first event occurred at ECAPn pin.

Table 6-641. PRUSS_ECAP_ECFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INT	R	0h	Global Interrupt Status Flag 0h: Indicates no interrupt generated. 1h: Indicates that an interrupt was generated.

Table 6-642. Register Call Summary for PRUSS_ECAP_ECFLG

PRU-ICSS eCAP Module
- PRUSS_ECAP_ECFLG Register (Offset = 2Eh) [reset = 0h]: [0]
- PRU-ICSS_ECAP Registers: [0]

6.4.8.2.11 PRUSS_ECAP_ECCLR Register (Offset = 30h) [reset = 0h]

PRUSS_ECAP_ECCLR is shown in Figure 6-236 and described in Table 6-644.

Return to Summary Table.

ECAP Interrupt Clear Register

Table 6-643. PRUSS_ECAP_ECCLR Instances

Instance	Physical Address
PRU_ICSS_0_ECAP	20AB 0030h
PRU_ICSS_1_ECAP	20AF 0030h

Figure 6-236. PRUSS_ECAP_ECCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CMPEQ	PRDEQ	CNTOVF	CEVT4	CEVT3	CEVT2	CEVT1	INT
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-644. PRUSS_ECAP_ECCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	CMPEQ	R/W	0h	Counter Equal Compare Status Flag 0h: Writing a 0 has no effect. Always reads back a 0 1h: Writing a 1 clears the CTR=CMP flag condition
6	PRDEQ	R/W	0h	Counter Equal Period Status Flag 0h: Writing a 0 has no effect. Always reads back a 0 1h: Writing a 1 clears the CTR=PRD flag condition
5	CNTOVF	R/W	0h	Counter Overflow Status Flag 0h: Writing a 0 has no effect. Always reads back a 0 1h: Writing a 1 clears the CNTOVF flag condition
4	CEVT4	R/W	0h	Capture Event 4 Status Flag 0h: Writing a 0 has no effect. Always reads back a 0. 1h: Writing a 1 clears the CEVT3 flag condition.
3	CEVT3	R/W	0h	Capture Event 3 Status Flag 0h: Writing a 0 has no effect. Always reads back a 0. 1h: Writing a 1 clears the CEVT3 flag condition.
2	CEVT2	R/W	0h	Capture Event 2 Status Flag 0h: Writing a 0 has no effect. Always reads back a 0. 1h: Writing a 1 clears the CEVT2 flag condition.
1	CEVT1	R/W	0h	Capture Event 1 Status Flag 0h: Writing a 0 has no effect. Always reads back a 0. 1h: Writing a 1 clears the CEVT1 flag condition.
0	INT	R/W	0h	Global Interrupt Clear Flag 0h: Writing a 0 has no effect. Always reads back a 0. 1h: Writing a 1 clears the INT flag and enable further interrupts to be generated if any of the event flags are set to 1.

Table 6-645. Register Call Summary for PRUSS_ECAP_ECCLR

PRU-ICSS eCAP Module
• PRUSS_ECAP_ECCLR Register (Offset = 30h) [reset = 0h]: [0] • PRU-ICSS_ECAP Registers: [0]

6.4.8.2.12 PRUSS_ECAP_ECFRC Register (Offset = 34h) [reset = 0h]

PRUSS_ECAP_ECFRC is shown in Figure 6-237 and described in Table 6-647.

Return to Summary Table.

ECAP Interrupt Forcing Register

Table 6-646. PRUSS_ECAP_ECFRC Instances

Instance	Physical Address
PRU_ICSS_0_ECAP	20AB 0034h
PRU_ICSS_1_ECAP	20AF 0034h

Figure 6-237. PRUSS_ECAP_ECFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CMPEQ	PRDEQ	CNTOVF	CEVT4	CEVT3	CEVT2	CEVT1	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-647. PRUSS_ECAP_ECFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	CMPEQ	R/W	0h	Force Counter Equal Compare Interrupt 0h: No effect. Always reads back a 0. 1h: Writing a 1 sets the CTR=CMP flag bit.
6	PRDEQ	R/W	0h	Force Counter Equal Period Interrupt 0h: No effect. Always reads back a 0. 1h: Writing a 1 sets the CTR=PRD flag bit.
5	CNTOVF	R/W	0h	Force Counter Overflow 0h: No effect. Always reads back a 0. 1h: Writing a 1 to this bit sets the CNTOVF flag bit.
4	CEVT4	R/W	0h	Force Capture Event 4 0h: No effect. Always reads back a 0. 1h: Writing a 1 sets the CEVT4 flag bit
3	CEVT3	R/W	0h	Force Capture Event 3 0h: No effect. Always reads back a 0. 1h: Writing a 1 sets the CEVT3 flag bit
2	CEVT2	R/W	0h	Force Capture Event 2 0h: No effect. Always reads back a 0. 1h: Writing a 1 sets the CEVT2 flag bit.
1	CEVT1	R/W	0h	Always reads back a 0. Force Capture Event 1 0h: No effect. 1h: Writing a 1 sets the CEVT1 flag bit.
0	RESERVED	R	0h	Reserved

Table 6-648. Register Call Summary for PRUSS_ECAP_ECFRC

PRU-ICSS eCAP Module
• PRU-ICSS_ECAP Registers: [0] • PRUSS_ECAP_ECFRC Register (Offset = 34h) [reset = 0h]: [0]

6.4.8.2.13 PRUSS_ECAP_PID Register (Offset = 5Ch) [reset = -h]

PRUSS_ECAP_PID is shown in Figure 6-238 and described in Table 6-650.

Return to Summary Table.

ECAP Revision ID

Table 6-649. PRUSS_ECAP_PID Instances

Instance	Physical Address
PRU_ICSS_0_ECAP	20AB 005Ch
PRU_ICSS_1_ECAP	20AF 005Ch

Figure 6-238. PRUSS_ECAP_PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REVISION																															
R--h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 6-650. PRUSS_ECAP_PID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REVISION	R	-h	IP Revision

Table 6-651. Register Call Summary for PRUSS_ECAP_PID

PRU-ICSS eCAP Module
- PRUSS_ECAP_PID Register (Offset = 5Ch) [reset = -h]: [0]
- PRU-ICSS_ECAP Registers: [0]

6.4.9 PRU-ICSS MII RT Module

6.4.9.1 Introduction

The Real-time Media Independent Interface (MII_RT) provides a programmable I/O interface for the PRUs to access and control up to two MII ports. The MII_RT module can also be configured to push and pull data independent of the PRU cores.

NOTE: In order to guarantee the MII_RT IO timing values published in the device data manual, the ICSS_i_VCLK_CLK (where i = 0 or 1) clock must be configured for 200MHz (default value) and PRUSS_MII_RT_TXCFG0/1 must be set to 6h (non-default value).

6.4.9.1.1 Features

The PRU-ICSS MII_RT module supports:

- Two MII ports
  - Each MII port has:
    - 32-byte RX L1 FIFO
    - 64-byte RX L2 buffer
    - 96-byte TX L1 FIFO
  - Rate decoupling on TX L1 FIFO
  - Configurable pre-amble removal on RX L1 FIFO and insertion on TX L1 FIFO
  - Configurable TX L1 FIFO trigger (10 bits with 40 ns ticks)
- MII port multiplexer per direction to support line/ring structure
  - Link detection through RX_ERR
- Cyclic redundancy check (CRC)
  - CRC32 generation on TX path
  - CRC32 checker on RX path

6.4.9.1.2 Unsupported Features

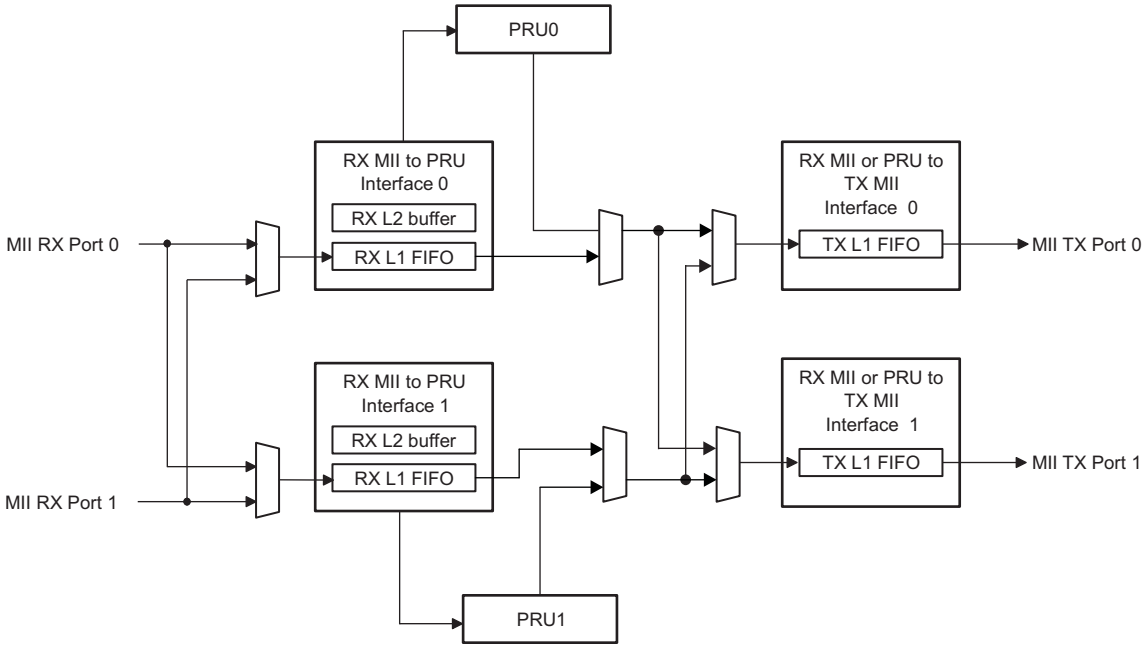
The PRU-ICSS MII_RT module does not support:

- Auto padding in TX L1 FIFO
- Dynamic TX multiplexer switching during packet handling
  - Can allow one PRU to handle both MII interfaces and a second PRU to manage the host and switch functions.

6.4.9.1.3 Block Diagram

Figure 6-239 shows the MII_RT in context of the PRU-ICSS. This diagram is a conceptual block diagram and does not necessarily reflect actual topologies.

Figure 6-239. MII_RT Block Diagram



6.4.9.2 Functional Description

6.4.9.2.1 Data Path Configuration

The MII_RT module supports three basic data path configurations. These configurations are compared in Table 6-652 and described in the following sections.

Table 6-652. Data Path Configuration Comparison

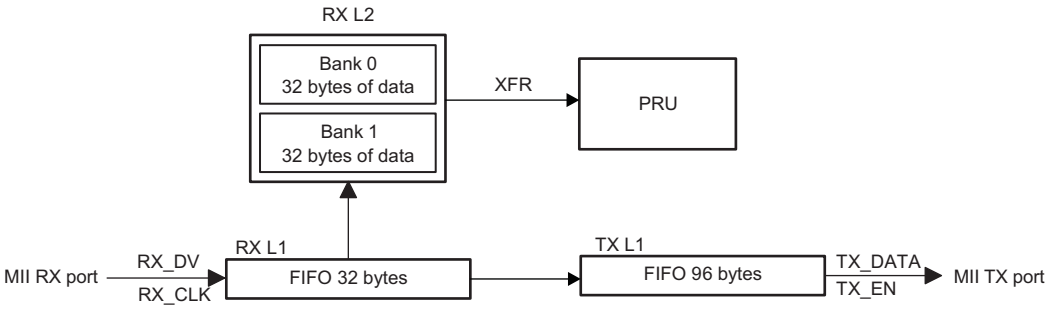
Configuration	PRU Dependency	Data Servicing	Port-to-Port Latency
Auto-forward	Snoop only	One word in flight	Low
8- or 16-bit processing with on-the-fly modifications (RX L1)	Yes	One word or byte in flight	Low
32-byte double buffer or ping-pong processing (RX L2)	Yes	Multi-words in flight	Medium (application-dependent)

6.4.9.2.1.1 Auto-forward with Optional PRU Snoop

Data is automatically forwarded from the MII RX port to the MII TX port without manipulations, as shown in Figure 6-240. This configuration does not depend on the PRU core. However, it does support an option for PRU to snoop or monitor the received data through the RX L2, shown in Figure 6-241. The PRU does not access data and status bits through R31, and it does not modify and push data.

Figure 6-240. Auto-forward

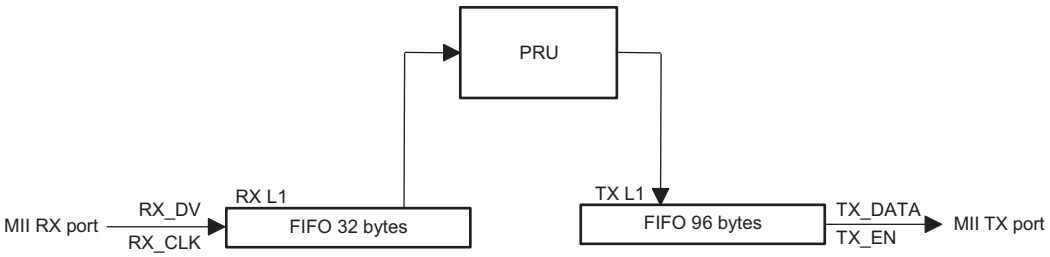
Figure 6-241. Auto-forward with PRU Snoop



6.4.9.2.1.2 8- or 16-bit Processing with On-the-Fly Modifications

This configuration services one byte or word in flight and has low latency. The PRU has the option to manipulate the received word and control popping data from the RX L1 FIFO and pushing it on the TX L1 FIFO.

Figure 6-242. 8- or 16-bit Processing with On-the-Fly Modifications

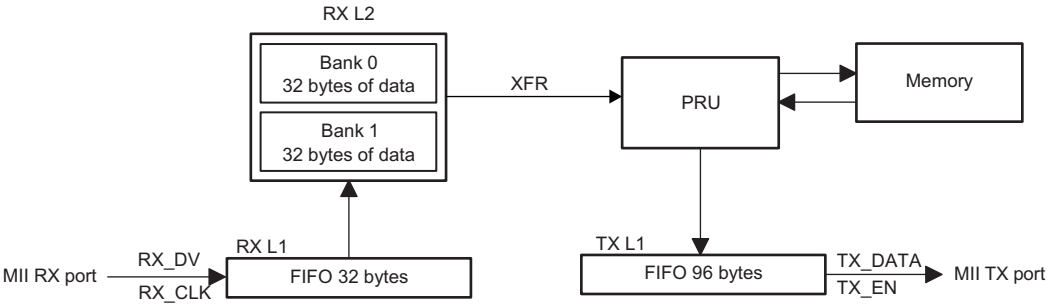


6.4.9.2.1.3 32-byte Double Buffer or Ping-Pong Processing

This configuration supports high bandwidth, high efficiency transactions. Often implementations using this mode permit relaxed servicing requirements allowing the PRU to manipulate the received data before transmitting.

Data received in this configuration is passed into the RX L2 buffer. The PRU reads multiple bytes of data from one of the RX L2 banks through the high bandwidth broadside interface and XFR instructions. The PRU can then store or manipulate data before pushing it to the TX L1 FIFO for transmission on the MII TX port.

Figure 6-243. 32-byte Double Buffer or Ping-Pong Processing



6.4.9.2.2 Definition and Terms

6.4.9.2.2.1 Data Frame Structure

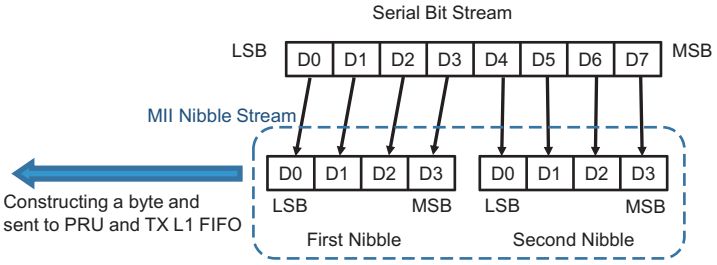
The data received and transmitted over MII conforms with the frame structure shown in Table 6-653.

Table 6-653. Frame Structure

Inter-frame	Preamble	Start of Frame Delimiter (SFD)	Data	Cyclic Redundancy Check (CRC)

The data following the SFD is formatted in a 4-bit nibble structure. Figure 6-244 illustrates the nibble order. The MSB arriving first is on the LSB side of a nibble. When receiving data, the MII_RT receive logic will wait for the next nibble to arrive before constructing a byte and delivering to the PRU.

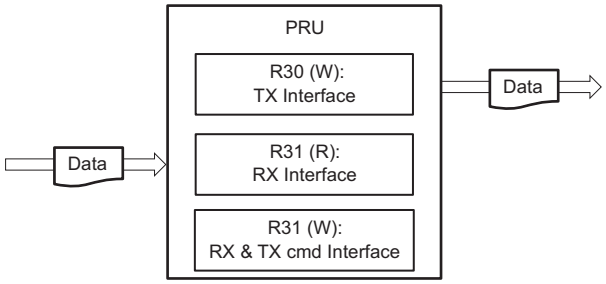
Figure 6-244. Data Nibble Structure



6.4.9.2.2.2 PRU R30 and R31

The PRU registers R30 and R31 are used to receive, transmit, and control the data for the PRU. As shown in Figure 6-245, the R31 is used to access data in the RX L1 FIFO, the R30 is used to transmit data from the PRU, and the R31 output is used the control the flow of receive and transmit. For more details about these registers, see the following sections.

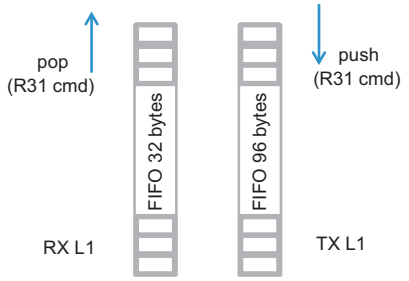
Figure 6-245. PRU R30, R31 Operations



6.4.9.2.2.3 RX and TX L1 FIFO Data Movement

To advance the next data byte seen by R31, the PRU must pop the data from the RX L1 FIFO. Likewise, the PRU can push the data from R30 to the TX L1 FIFO. These operations are illustrated in Figure 6-246.

Figure 6-246. Reading and Writing FIFO Data



6.4.9.2.2.4 CRC Computation

6.4.9.2.2.4.1 Receive CRC Computation

For the incoming data, the MII_RT calculates CRC32 and then compares against the value provided in the incoming frame. If there is a mismatch, the MII RT signals ERROR_CRC to the PRU. If a previous node or Ethernet device appended an error nibble, the CRC calculation of received packet will be wrong because the longer frame and the frame length will end at a 4-bit boundary instead of the usual 8-bit boundary. When RX_DV goes inactive on the 4-bit boundary, the interface will assert DATA_RDY and BYTE_RDY flag with the ERROR NIBBLE. The error event is also mapped into the PRU-ICSS INTC.

6.4.9.2.2.4.2 Transmit CRC Computation

For the outgoing data, the MII_RT calculates the CRC32 value and inserts it into outgoing packets. The CRC value computed on each MII transmit path is also available in memory map registers (MMRs) that can be read by the PRU and used primarily for debug and diagnostic purposes. The CRC is inserted into the outgoing packet based on the commands received through the R31 register of the PRU. The CRC will be inserted into the TX L1 FIFO, and there must be enough room to store the CRC value in the FIFO or else the FIFO will overflow. As Table 6-654 shows, the CRC programming model supports three sequences that provide more flexibility. Note “cmdR31” indicates write to the mentioned bits of the R31 command interface.

Table 6-654. TX CRC Programming Models

Option 1	Step 1: cmdR31 [TX_CRC_HIGH + TX_CRC_LOW + TX_EOF]
Option 2	Step 1: cmdR31 [TX_CRC_HIGH] Step 2: wait > 6 clocks (PRU cycles) Step 3: cmdR31 [TX_CRC_LOW + TX_EOF]
Option 3	Step 1: cmdR31 [TX_CRC_HIGH] Step 2: wait > 6 clocks (PRU cycles) Step 3: read PRUSS_MII_RT_TX_CRC0/1 Step 4: modify CRC[15:0] Step 5: cmdR31 [TX_PUSH16 + TX_EOF + TX_ERROR_NIBBLE]

6.4.9.2.3 RX MII Interface

6.4.9.2.3.1 RX MII Submodule Overview

The RX MII interface is composed of multiple submodules that process the incoming frames and pass receive data and status information into the PRU register R31. These submodules include:

- Latch received data
- Start of frame detection
- Start frame delimiter detection

- CRC calculation and error detection
- Enhanced link detection through RX error detection

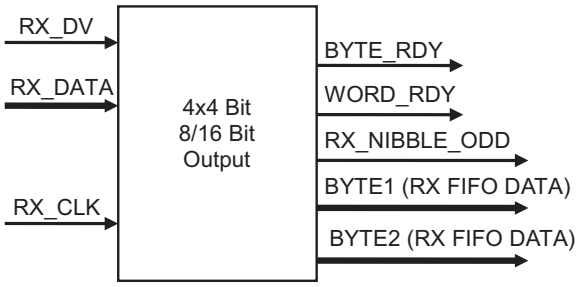
Table 6-655 includes more details about the internal signals and output of these submodules.

6.4.9.2.3.1.1 Receive Data Latch

The receive data from the MII interface is stored in the receive data FIFO which is 32 bytes. The PRU can access this data through the register R31. Depending on the configuration settings, the data can be latched on reception of one or two bytes. In each scheme, the configured number of nibbles is assembled before being copied into the PRU registers. Figure 6-247 shows the inputs and outputs of the data latch logic block.

The receiver logic in MII_RT can be programmed through the PRUSS_MII_RT_RXCFG0 and PRUSS_MII_RT_RXCFG1 registers to remove or retain the preamble + SFD from incoming frames.

Figure 6-247. RX Data Latch



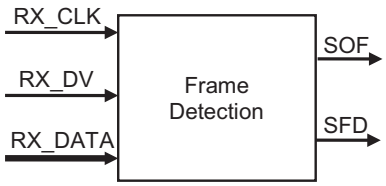
6.4.9.2.3.1.1.1 Start of Frame Detection

The start of frame detection logic tracks the frame boundaries and signals the beginning of a frame to other components of the PRU-ICSS. This logic detects two events:

- Start of Frame (SOF) event that occurs when Receive Data Valid MII signal is sampled high.
- Start of Frame Delimiter (SFD) event is seen on MII Receive Data bus.

These event triggers can be used to add timestamp to the frames. The notification for these events is available through R31 as well as through INTC which is integrated in the PRU-ICSS. Figure 6-248 shows the inputs and outputs of the start of frame detection logic block.

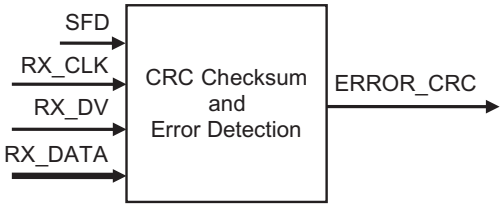
Figure 6-248. Start of Frame Detection



6.4.9.2.3.1.1.2 CRC Error Detection

For each incoming frame, the CRC is calculated by the MII_RT and compared against the CRC included in the frame. When the two values do not match, a CRC error is flagged. The ERROR_CRC indication is available in the register interface (PRU R31 Receive Interface) as well as in the FIFO interface (RX L2 Status Interface). It is also provided to the INTC which is integrated in the PRU-ICSS. Figure 6-249 shows the inputs and outputs of CRC error detection logic block.

Figure 6-249. CRC Error Detection

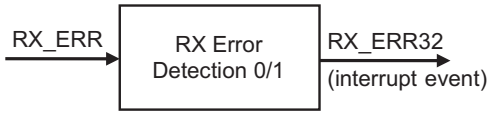


6.4.9.2.3.1.1.3 RX Error Detection and Action

The RX error detection logic tracks the receive error signaled by the physical layer and informs the PRU-ICSS INTC whenever an error is detected. Figure 6-250 shows the inputs and outputs of the RX error detection logic block. Note the following dependencies:

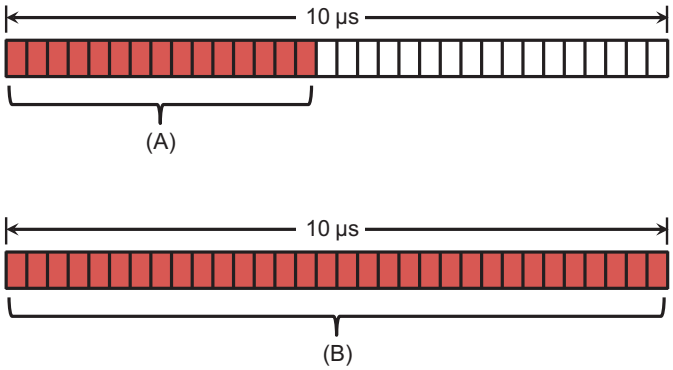
- RX_ERR signal is only sampled when RX_DV is asserted.
- All nibbles are discarded post RX_ERR event, including the nibble which had RX_ERR asserted. This state will remain until EOF occurs.
- Due to this fact, RX L1 FIFO and RX L2 FIFO will never receive any data with RX_ERR or post RX_ERR during that frame.

Figure 6-250. RX Error Detection



This submodule also keeps track of a running count of receive error events within a 10 µs error detection window, as shown in Figure 6-251. The INTC is notified when 32 or more events have occurred in a 10 µs error detection window. The error detection window is not a sliding window but a non-overlapping window with no specific initialization time with respect to incoming traffic. The timer starts its 10 µs counts immediately after de-assertion of reset to the MII_RT module.

Figure 6-251. Error Detection Window with Running Counter



A There are fewer than 32 consecutive error events in the 10 µs window. The detection module will not forward to the interrupt controller (INTC).

B There are more or equal to 32 error events in the 10 µs window. The detection module will notify the interrupt controller (INTC).

6.4.9.2.3.1.2 RX Data Path Options to PRU

There are two data path options for delivering received data to the PRU, described further in the subsequent sections:

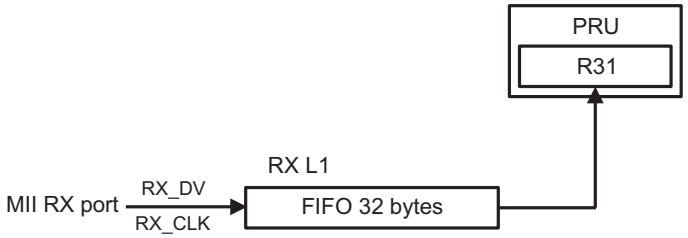
1. RX MII port → RX L1 FIFO → PRU (one word in flight)
2. RX MII port → RX L1 FIFO → RX L2 buffer → PRU (multi-word in flight)

Once the PRU has received RX data, the PRU can both manipulate received data or send data to the TX MII Interface.

6.4.9.2.3.1.2.1 RX MII Port → RX L1 FIFO → PRU

The RX L1 FIFO to PRU interface is depicted in Figure 6-252. In this mode, the data received from the MII interface is fed into the 32-byte RX L1 FIFO. The first data byte into the FIFO is automatically available in R31 of the PRU. Therefore, the PRU firmware can directly operate on this data without having to read it in a separate instruction. This allows the PRU to access receive data with low latency.

Figure 6-252. RX L1 to PRU Interface



When the new data is received, the PRU is provided with up to two bytes at a time in the R31 register, as shown in Figure 6-253. Once the PRU processes the incoming data, it instructs the MII_RT by writing to the R31 command interface bits to pop one or two bytes of data from the 32-byte RX FIFO. The pop operation causes current contents of R31 to be refreshed with new data from the incoming packet. Each time the data is popped, the status bits change to indicate so. If the pop is completed and there is no new data, the status bits immediately change to indicate no new data. Note the current R31 content, including data, will be lost after issuing the pop operation. If this information needs to be accessed later, the PRU should store the existing R31 content before popping new data.

Figure 6-253. MII RX Data to PRU R31 (R) and RX FIFO

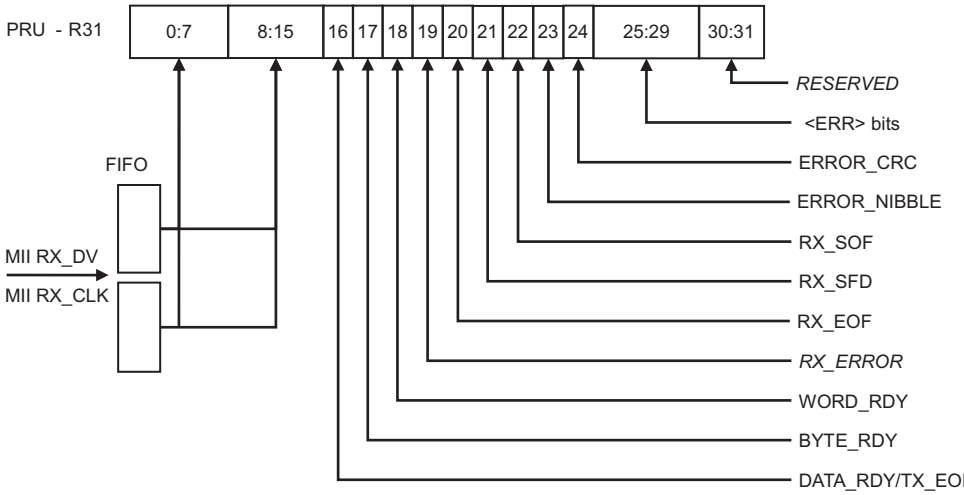


Table 6-655 describes the receive interface data and status contents provided by the R31 register. These contents are available when R31 is read. To configure this register, the PRU GPI mode should be set for MII_RT mode in the CFG register space. Note the following:

1. If the data from receive path is not read in time, it could cause an overflow event because the data is still continuously provided to the 32-byte receive FIFO. Due to the receive FIFO overflow, the data gets automatically discarded to avoid lack of space in the FIFO. At the same time, an interrupt is raised to the INTC through a system event (PRU<n>_RX_OVERFLOW). To detect an overflow condition, the PRU should poll for this system event condition and a RX RESET command through the R31 command interface is required to clear out from this condition. Note that the received Ethernet frame is corrupted and should not be used for further processing as bytes have been dropped due to the overflow condition. A FIFO reset is recommended.

2. The receive data in the R31 register is available following synchronization to the PRU clock domain. So, there is a finite delay (120 ns) when data is available from MII interface and it is accessible to the PRU.
3. The receive FIFO also has the capability to be reset through software. When reset, all contents of receive FIFO are purged and it may result in the current frame not being received as expected. When a frame is being received and the PRU resets the RX FIFO, the remaining frame is not placed into the RX FIFO. However, any new frame arriving on the receive MII port will be stored in the FIFO.

Table 6-655. PRU R31: Receive Interface Data and Status (Read Mode)

Bits	Field Name	Description
31:30	RESERVED	In case of register interface, these bits are provided to PRU by other modules in PRU-ICSS. From the MII_RT module point of view, these bits are always zero.
29	RX_MIN_FRM_CNT_ERR	RX_MIN_FRM_CNT_ERR is set to 1 when the count of total bytes of incoming frame is less than the value defined by RX_MIN_FRM_CNT. RX_MIN_FRM_CNT_ERR is cleared by RX_ERROR_CLR.
28	RX_MAX_FRM_CNT_ERR	RX_MAX_FRM_CNT_ERR is set to 1 when the count of total bytes of incoming frame is more than the value defined by RX_MAX_FRM_CNT_ERR. RX_MAX_FRM_CNT_ERR is cleared by RX_ERROR_CLR.
27	RX_EOF_ERROR	RX_EOF_ERROR is set to 1 when an RX_EOF event or RX_ERROR event occurs. RX_EOF_ERROR is cleared by RX_ EOF_CLR and/or RX_ ERROR_CLR.
26	RX_MAX_PRE_CNT_ERR	RX_MAX_PRE_CNT_ERR is set to 1 when the number of nibbles equaling 0x5 before SFD event (0xD5) is more than the value defined by PRUSS_MII_RT_RX_PCNT0/1 [RX_MAX_PCNT]. RX_MAX_PRE_CNT_ERR is cleared by RX_ERROR_CLR.
25	RX_ERR	RX_ERR is set to 1 when pr1_mii0/1_rxer is asserted while pr1_mii0/1_rxdv bit is set. RX_ERR is cleared by RX_ERROR_CLR.
24	ERROR_CRC	ERROR_CRC indicates that the frame has a CRC mismatch. This bit is valid when the RX_EOF bit is set. It should be noted that ERROR_CRC bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO. ERROR_CRC is cleared by RX_ERROR_CLR.
23	ERROR_NIBBLE	ERROR_NIBBLE indicates that the frame ended in odd nibble. It should be considered valid only when the RX_EOF bit and pr1_mii0/1_rxdv are set. Nibble counter is enabled post SFD event. It should be noted that ERROR_NIBBLE bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO. ERROR_NIBBLE is cleared by RX_ERROR_CLR.
22	RX_SOF	RX_SOF transitions from low to high when the frame data starts to arrive and pr1_mii0/1_rxdv is asserted. Note there will be a small sync delay of 0ns – 5ns. The recommended time to clear this bit via RX_SOF_CLR is at the end of frame (EOF). It should be noted that RX_SOF bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO.
21	RX_SFD	RX_SFD transitions from low to high when the SFD sequence (0xD5) post RX_SOF is observed on the receive MII data. The recommended time to clear this bit via RX_SFD_CLR is at the end of frame (EOF). It should be noted that RX_SFD bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO.
20	RX_EOF	RX_EOF indicates that the frame has ended and pr1_mii0/1_rxdv is de-asserted. It also validates the CRC match bit. Note there will be a small sync delay of 0ns – 5ns. It should be noted that RX_EOF bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO. Note also if RX_L2_EOF_SCLR_DIS is set, then this flag will remain asserted when RX_L2 is enabled until RX_EOF_CLR.
19	RX_ERROR	RX_ERROR indicates one or more of the following errors occurred: • RX_MAX/MIN_FRM_CNT_ERR • RX_MAX/MIN_PRE_CNT_ERR • RX_ERR RX_ERROR is cleared by RX_ERROR_CLR.
18	WORD_RDY	WORD_RDY indicates that all four nibbles in R31 have valid data. There is a 2 clock cycle latency from the command RX_POP16 to WORD_RDY update. Therefore, firmware needs to insure it does not read WORD_RDY until 2 clock cycles after RX_POP16.
17	BYTE_RDY	BYTE_RDY indicates that the lower two nibbles in R31 have valid data. There is a 2 clock cycle latency from the command RX_POP8 to BYTE_RDY update. Therefore, PRU firmware needs to insure it does not read BYTE_RDY until 2 clock cycles after RX_POP8.

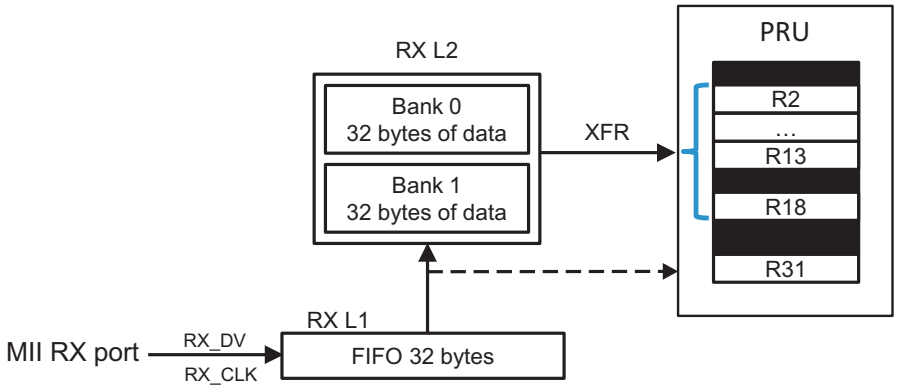
Table 6-655. PRU R31: Receive Interface Data and Status (Read Mode) (continued)

Bits	Field Name	Description
16	DATA_RDY/ TX_EOF	When RX_DATA_RDY_MODE_DIS = 0: DATA_RDY indicates there is valid data in R31 ready to be read. This bit goes to zero when the PRU does a POP8/16 and there is no new data left in the receive MII port. This bit is high if there is more receive data for PRU to read. There is a 2 clock cycle latency from the command RX_POP16/8 to WORD_RDY/BYTE_RDY update. Therefore, PRU firmware needs to insure it does not read BYTE_RDY/WORD_RDY until 2 clock cycles after RX_POP16/8. When RX_DATA_RDY_MODE_DIS = 1: TX_EOF indicates an TX EOF event (i.e. a 1 --> 0 transition on TX_EN) has occurred. This bit will clear when TX_RESET is set or when new data is first loaded. PRU firmware can wait until TX_EOF = 1, then start a new TX Frame by immediately loading new data.
15:8	BYTE1	Data Byte 1. This data is available such that it is safe to read by the PRU when the DATA_RDY/BYTE_RDY/WORD_RDY bits are asserted.
7:0	BYTE0	Data Byte 0. This data is available such that it is safe to read by the PRU when the DATA_RDY/BYTE_RDY/WORD_RDY bits are asserted.

6.4.9.2.3.1.2.2 RX MII Port → RX L1 FIFO → RX L2 Buffer → PRU

The RX L2 is an optional high performance buffer between the RX L1 FIFO and the PRU. Figure 6-254 illustrates the receive data path using RX L2 buffer. This data path is characterized by multi-word in flight transactions.

Figure 6-254. RX L2 to PRU Interface



The 64-byte RX L2 buffer is divided into two 32 byte banks, or ping/pong buffers. When the RX L2 is enabled, the incoming data from the MII RX port will transmit first to the 32 byte RX L1 FIFO. RX L1 pushes data into RX L2, starting when the first byte is ready until the final EOF marker. The RX L2 buffer does not apply any backpressure to the RX L1 FIFO. Therefore, it is the PRU firmware's responsibility to fetch the data in RX L2 before it is overwritten by the cyclic buffer. The RX L1 will remain near empty, with only one byte (nibble) stored.

Each RX L2 bank holds up to 32 bytes of data, and every four nibbles (or 16 bits) of data has a corresponding 8-bit status. The data and status information are stored in packed arrays. In each bank, R2 to R9 contains the data packed array and R10 to R13 contains the status packed array. Figure 6-255 shows the relationship of the data registers and status registers. The RX L2 status registers record status information about the received data, such as ERROR_CRC, RX_ERROR, STATUS_RDY, etc. The RX L2 status register details are described in Table 6-656. Note RX_RESET clears all Data and Status elements and resets R18.

Figure 6-255. Data and Status Register Dependency

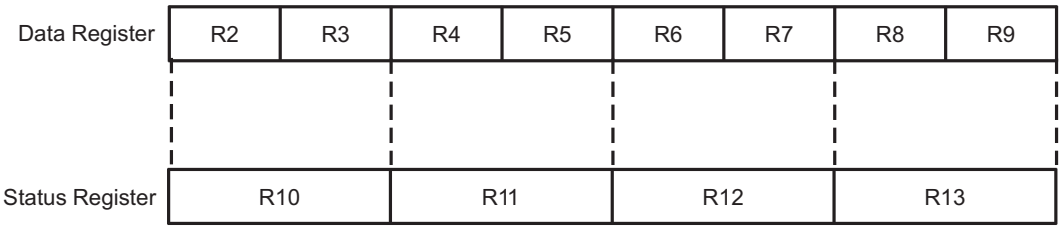


Table 6-656. RX L2 Status

Bit	Field Name	Description
7	ERROR_CRC	ERROR_CRC indicates that the frame has a CRC mismatch. This bit is valid when the RX_EOF bit is set. It should be noted that ERROR_CRC bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO. ERROR_CRC will only be set for one entry, self clear on next entry.
6	ERROR_NIBBLE	ERROR_NIBBLE indicates that the frame ended in odd nibble. It should be considered valid only when the RX_EOF bit and pr1_mii0/1_rxdv are set. Nibble counter is enabled post SFD event. It should be noted that ERROR_NIBBLE bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO. ERROR_NIBBLE will only be set for one entry, self clear on next entry.
5	RX_SOF	RX_SOF transitions from low to high when the frame data starts to arrive and pr1_mii0/1_rxdv is asserted. Note there will be a small sync delay of 0ns – 5ns. It should be noted that RX_SOF bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO. RX_SOF will only be set for one entry, self clear on next entry.
4	RX_SFD	RX_SFD transitions from low to high when the SFD sequence (0xD5) post RX_SOF is observed on the receive MII data. It should be noted that RX_SFD bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO. RX_SOF will only be set for one entry, self clear on next entry.
3	RX_EOF	RX_EOF indicates that the frame has ended and pr1_mii0/1_rxdv is de-asserted. It also validates the CRC match bit. Note there will be a small sync delay of 0ns – 5ns. It should be noted that RX_EOF bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO. If RX_L2_EOF_SCLR_DIS = 1, then RX_EOF will remain set until RX_EOF_CLR event. Otherwise, RX_ERROR is self-clearing on next entry.
2	RX_ERROR	RX_ERROR indicates one or more of the following errors occurred: • RX_MAX/MIN_FRM_CNT_ERR • RX_MAX/MIN_PRE_CNT_ERR • RX_ERR RX_ERROR is cleared by RX_ERROR_CLR.
1	STATUS_RDY	STATUS_RDY is set when RX_EOF or write pointer advanced by 2. This is a simple method for software to determine if RX_EOF event has occurred or new data is available. If RX_EOF is not set, all status bits are static.
0	RX_ERR	RX_ERR is set to 1 when pr1_mii0/1_rxer is asserted while pr1_mii0/1_rxdv bit is set. It will get set for first pr1_mii0/1_rxer event and self clear on SOF for the next FRAME.

Bank 0 and Bank 1 are used as ping/pong buffers. RX L2 supports the reading of a write pointer in R18 that allows software to determine which bank has active write transactions, as well as the specific write address within packed data arrays.

The PRU interacts with the RX L2 buffer using the high performance XFR read instructions and broadside interface. Table 6-657 shows the device XFR ID numbers for each bank.

Table 6-657. RX L2 XFR ID

Device ID	Function	Description
20	Selects RX L2 Bank0	R2:R9 Data packed array R10:R13 Status packed array
21	Selects RX L2 Bank1	R2:R9 Data packed array R10:R13 Status packed array
20/21	Byte pointer of current write	R18[5:0] Pointer indicating location of current write in data packed array. 0 = Bank0.R2.Byte0 (default and reset value) 1 = Bank0.R2.Byte1 2 = Bank0.R2.Byte2 3 = Bank0.R2.Byte3 4 = Bank0.R3.Byte0 … 63=Bank1.R9.Byte3

XFR read transactions are passive and have no effect on any status or other states in RX L2. The firmware can also read R18 to determine which Bank has active write transactions and the location of the transaction. With this information, the firmware can read multiple times the stable preserved data. Note when RX L1 data is written to RX L2, the next status byte gets cleared at the same time the current status byte gets updated. The rest of the status buffer is persistent. When software is accessing any register of the ping/ pong buffer, software needs to issue an XFER read transaction to fetch the latest/current state of the ping/pong buffer. The PRU registers will not reflect the current snapshot of L2 unless an XFER is issued by software.

6.4.9.2.4 TX MII Interface

Data to be transmitted is loaded into the TX L1 FIFO. The transmit FIFO (TX L1) stores up to 96 bytes of transmit data. From the FIFO, the data is sent to the MII TX port of the PHY by the MII_RT transmit logic.

The transmit FIFO also has the capability to be reset through software (TX_RESET). When reset, all contents of transmit FIFO are purged and this may result in a frame not getting transmitted as expected, if the transmission is already ongoing. Any new data written in the transmit FIFO results in a new frame being composed and transmitted. An overflow event will require a TX_RESET to recover from this condition.

There are four dependencies that must be true for TX_EN to assert.

1. TX L1 FIFO not empty
2. Interpacket gap (IPG) timer expiration
3. RX_DV to TX_EN timer expiration
4. TX_EN compare timer expiration

The transmit interface also provides an underflow error signal in case there was no data loaded when TX_EN triggered. The transmit underflow signal is mapped to the INTC in PRU-ICSS. The PRU firmware must track the FIFO fill level, such as by a timer or the PRU cycle count register (PRU_ICSS_CTRL_CYCLE).The current FIFO fill level cannot be accessed by PRU firmware. The firmware can issue an R31 command via R31 bit 29 (TX_EOF) to indicate that the last byte has been written into the TX FIFO.

6.4.9.2.4.1 TX Data Path Options to TX L1 FIFO

There are two data path options for delivering data to the TX L1 FIFO and transmit port, described further in the subsequent sections:

1. PRU → TX L1 FIFO → TX MII port
2. RX L1 FIFO → TX L1 FIFO → TX MII port

6.4.9.2.4.1.1 PRU → TX L1 FIFO → TX MII Port

The PRU can be used to feed data into the TX L1 FIFO using the R30 and R31 registers, shown in Figure 6-256. The PRU has the option to write up to two or four bytes of R30 and then pushes the data into the TX L1 FIFO by writing to the R31 command interface.

Figure 6-256. PRU to TX L1 Interface

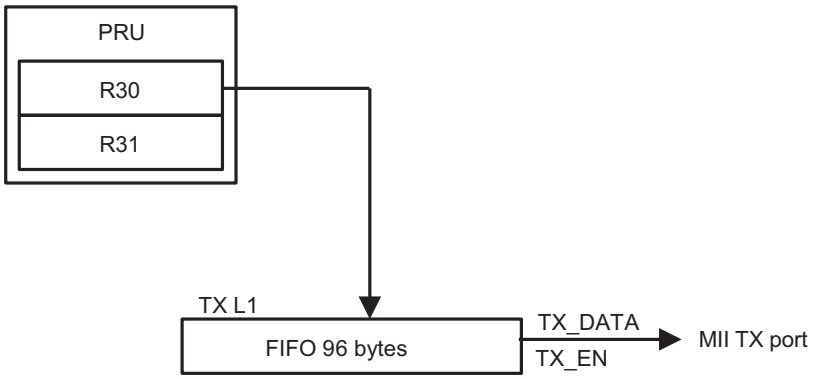


Figure 6-257 shows the R30 transmit interface. The lower 16 bits of the R30 (or FIFO transmit word) contain transmit data nibbles. When PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 0, then the upper 16 bits contain mask information. Alternatively, when PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 1, then the upper 16 bits contain transmit data nibbles. The operation to be performed on the transmit interface is controlled by PRU writes to the R31 command interface. Table 6-658 describes the supported configurations for 8, 16, and 32 bit TX push operations.

Figure 6-257. PRU to TX MII Interface

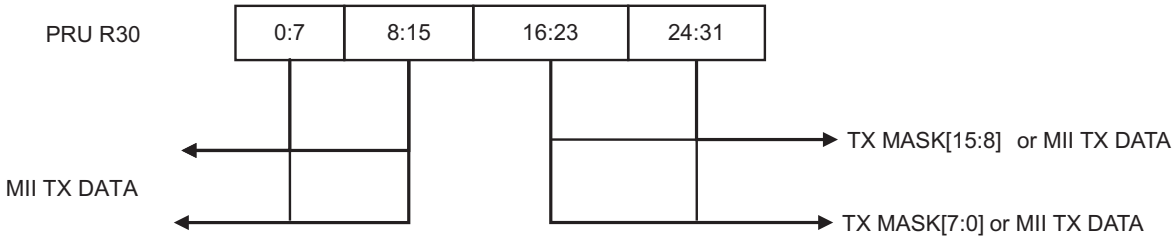


Table 6-658. TX Push

R31[25] TX_PUSH16/32	R31[24] TX_PUSH8/32	Supported R30 bits	TX_32_MODE_EN	TX_BYTE_SWAP	TX Push Action
0	1	X	0	X	8 bits of TXDATA (R30[7:0]) pushed post TX mask
1	0	X	0	X	16 bits of TXDATA (R30[15:0]) pushed post TX mask
1	1	X	0	X	Illegal
X	X	0x000000FF	1	0	8 bits of TXDATA (R30[7:0]) pushed
X	X	0x0000FFFF	1	0	16 bits of TXDATA (R30[15:0]) pushed
X	X	0xFFFFFFFF	1	X	32 bits of TXDATA (R30[31:0]) pushed
X	X	All other - reserved	1	X	Reserved

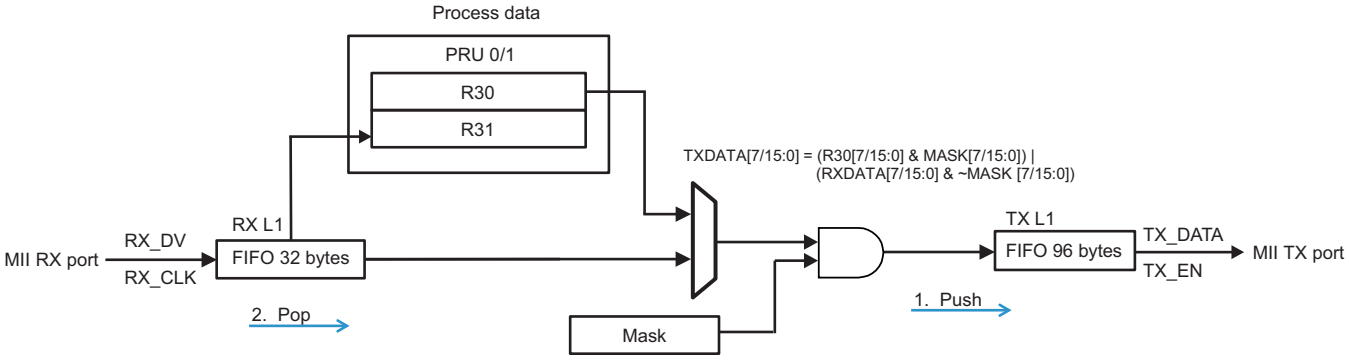
Using PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 0 and the TX mask, the PRU can send a mix of R30 and RX L1 FIFO data to the TX L1 FIFO. Note the TX mask is only available when the PRU is fed one word or byte at a time by the RX L1 FIFO. It is not applicable when the RX L2 buffer is enabled. To disable TX mask, set TXMASK to 0xFFFF.

As shown in Figure 6-258, the PRU drives the MII transmit interface through its R30 register. The contents of R30 and RX data from the receive interface are taken and fed into a 96 byte transmit FIFO.

If PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 0, then before transmission, a mask is applied to the data portion of the R30 register. By using the mask, the PRU firmware can control whether received data from the RX L1 FIFO is sent to transmit, R30 data is sent to transmit, or a mix of the two is sent. The Boolean equation that is used by MII_RT to compose TX data is:

TXDATA[7/15:0] = (R30[7/15:0] & MASK[7/15:0]) | (RXDATA[7/15:0] & ~MASK [7/15:0])

Figure 6-258. TX Mask Mode (PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 0)



6.4.9.2.4.1.2 RX L1 FIFO → TX L1 FIFO → TX MII Port

When PRUSS_MII_RT_TXCFG0/1[TX_AUTO_SEQUENCE] is set, the data frame is passed from the RX to TX FIFOs without the any interaction of the PRU. This mode of operations is shown in Figure 6-259. The RX L1 will push into TX L1 as long as it is enabled and not full.

There is no PRU dependency in this mode and no option for the PRU to perform any operation to the TX L1 FIFO. RX_RESET clears all data and status elements.

Figure 6-259. RX L1 to TX L1 Interface

6.4.9.2.5 PRU R31 Command Interface

The PRU uses writes to R31[31:16] to control the reception and transmission of packets in register mode. Table 6-659 lists the available commands. Each bit in the table is a single clock pulse output from the PRU. When more than one action is to be performed in the same instant, the PRU firmware must set those command bits in one instruction.

Table 6-659. PRU R31: Command Interface (Write Mode)

Bit	Command	Description
31	TX_CRC_ERR	TX_CRC_ERR command when set will add 0xa5 byte to the TX L1 FIFO if the current FCS is valid. This bit can only be set with the TX_EOF command and optionally with the TX_ERROR_NIBBLE command. It cannot get set with any other commands, and the PRU firmware must wait > 2 clocks from the last command. Note for proper operations auto-forward preamble must be enabled.
30	TX_RESET	TX_RESET command is used to reset the transmit FIFO and clear all its contents. This is required to recover from a TX FIFO overrun.

Table 6-659. PRU R31: Command Interface (Write Mode) (continued)

Bit	Command	Description
29	TX_EOF	TX_EOF command is used to indicate that the data loaded is considered last for the current frame
28	TX_ERROR_NIBBLE	TX_ERROR_NIBBLE command is used to insert an error nibble. This makes the frame invalid. Also, it will add 0x0 after the 32-bit CRC.
27	TX_CRC_HIGH	TX_CRC_HIGH command ends the CRC calculations and pushes CRC[31:16] to append to the outgoing frame in the TX L1 FIFO. Note PRUSS_MII_RT_TX_CRC0/1 will become valid after 6 clock cycles.
26	TX_CRC_LOW	TX_CRC_LOW command pushes CRC[15:0] to append to the outgoing frame in the TX L1 FIFO.
25	TX_PUSH16	TX_PUSH16 command pushes R30[15:0] when PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 0. See Table x, TX Push for more details. Note there are no restrictions on concurrent PUSH/POP nor R30 requirements to maintain data. Back to back PUSH is supported.
24	TX_PUSH8	TX_PUSH8 command pushes R30[7:0] when PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 0. See Table x, TX Push for more details. Note there are no restrictions on concurrent PUSH/POP nor R30 requirements to maintain data. Back to back PUSH is supported.
23	RX_ ERROR_CLR	RX_ERROR_CLR command is used to clear RX_ ERROR indicator bit by writing 1.
22	RX_EOF_CLR	RX_EOF_CLR command is used to clear RX_EOF status indicator bit by writing 1.
21	RX_SFD_CLR	RX_SFD_CLR command is used to clear RX_SFD indicator bit by writing 1.
20	RX_SOF_CLR	RX_SOF_CLR command is used to clear RX_SOF indicator bit by writing 1.
19	Reserved	Reserved
18	RX_RESET	RX_RESET is used to reset the receive FIFO and clear all contents. This is required to recover from a RX FIFO overrun, if software does not want to undrain. The typical use case is assertion after RX_EOF. If asserted during an active frame, the following actions will occur: 1. Terminate the current frame 2. Block/terminate all new data 3. Flush/clear all FIFO elements 4. Cause RX state machine into an idle state 5. Cause EOF event 6. Cause minimum frame error, if you abort before minimum size reached
17	RX_POP16	RX_POP16 command advances the receive traffic by two bytes. This is only required when you are using R31 to read the data. After R31[15:0] is ready to read by PRU, it will set 1 to WORD_RDY, and the next new data will be allowed to advance. RX_POP16 to WORD_RDY update has 2 clock cycles latency. Firmware needs to insure it does not read WORD_RDY/BYTE_RDY until 2 clock cycles after RX_POP16.
16	RX_POP8	RX_POP8 command advances the receive traffic by one bytes. This is only required when you are using R31 to read the data. After R31[7:0] is ready to read by PRU, it will set 1 to BYTE_RDY, and the next new data will be allowed to advance. RX_POP8 to BYTE_RDY update has 2 clock cycles latency. Firmware needs to insure it does not read WORD_RDY/BYTE_RDY until 2 clock cycles after RX_POP8.

6.4.9.2.6 Other Configuration Options

6.4.9.2.6.1 Nibble and Byte Order

The PRU core is little endian. To support big endian, the MII_RT supports optional nibble swapping on both the RX and TX side.

On the receive side, the order of the two data bytes in RX R31 and the RX L2 buffer are configurable through the RX_BYTE_SWAP bit in the PRUSS_MII_RT_RXCFG0/1 registers, as shown in Table 6-660. Note the Nibble0 is the first nibble received.

Table 6-660. RX Nibble and Byte Order

Configuration	Order
PRUSS_MII_RT_RXCFG0/1[RX_BYTE_SWAP] = 0 (default)	R31[15:8] / RXL2[15:8] = Byte1{Nibble3,Nibble2} R31[7:0] / RXL2[7:0] = Byte0{Nibble1,Nibble0}
PRUSS_MII_RT_RXCFG0/1[RX_BYTE_SWAP] = 1	R31[15:8] / RXL2[15:8] = Byte0{Nibble1,Nibble0} R31[7:0] / RXL2[7:0] = Byte1{Nibble3,Nibble2}

On the transmit side, the order of the two data bytes and mask bytes in TX R30 are configurable through the TX_BYTE_SWAP bit in the PRUSS_MII_RT_TXCFG0/1 registers, as shown in Table 6-661. Note the Nibble0 is the first nibble received.

Table 6-661. TX Nibble and Byte Order

Configuration	Order
PRUSS_MII_RT_TXCFG0/1[TX_BYTE_SWAP] = 0 (default)	If PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 0, R30[15:8] = Byte1{Nibble3,Nibble2} R30[7:0] = Byte0{Nibble1,Nibble0} R30[31:24] = TX_MASK[15:8] R30[23:16] = TX_MASK[7:0] If PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 1, R30[31:24] = Byte3{Nibble7,Nibble6} R30[23:16] = Byte2{Nibble5,Nibble4} R30[15:8] = Byte1{Nibble3,Nibble2} R30[7:0] = Byte0{Nibble1,Nibble0}
PRUSS_MII_RT_TXCFG0/1[TX_BYTE_SWAP] = 1	If PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 0, R30[15:8] = Byte0{Nibble1,Nibble0} R30[7:0] = Byte1{Nibble3,Nibble2} R30[31:24] = TX_MASK[7:0] R30[23:16] = TX_MASK[15:8] If PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 1, Only 32bit push is supported. R30[31:24] = Byte0{Nibble1,Nibble0} R30[23:16] = Byte1{Nibble3,Nibble2} R30[15:8] = Byte2{Nibble5,Nibble4} R30[7:0] = Byte3{Nibble7,Nibble6}

6.4.9.2.6.2 Preamble Source

The MII_RT module has the option to preserve and forward a received preamble in the TX data stream, use a preamble provided by the PRU, or auto-generate a preamble. These configurations are highlighted in Table 6-662.

Table 6-662. Preamble Configuration Options

RX_CUT_PREAMBLE	Determines whether RX preamble is passed to the RX L1/L2 FIFO
RX_AUTO_FWD_PRE	Determines whether RX preamble is automatically passed to TX L1 FIFO

Table 6-662. Preamble Configuration Options (continued)

TX_AUTO_PREAMBLE	TX interface logic auto-generates and appends preamble to TX data stream with the first push of data into the TX L1 FIFO. Note that enabling this option does fill the TX FIFO with the preamble length, hence software has to consider this to not overrun the TX FIFO.

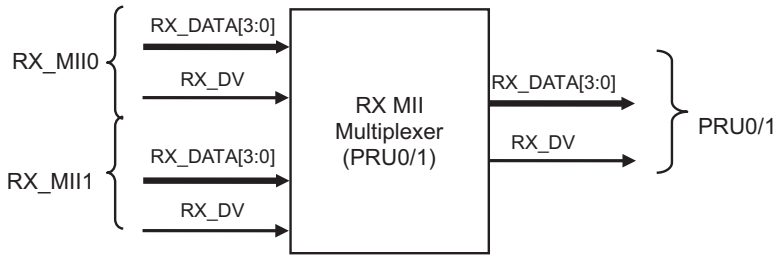
6.4.9.2.6.3 PRU and MII Port Multiplexer

The MII_RT module supports configurable PRU core to MII TXn / RXn port mapping. By default, PRU0 is mapped to TX1 and RX0 and PRU1 is mapped to TX0 and RX1. However, the system supports the flexibility to map any PRU core to any TX and RX port. Note the mapping options are destination fixed. For example, the input to PRU0 can be either RX_MII0 or RX_MII1. Similarly, the input to TX_MII0 can be either PRU0 or PRU1.

6.4.9.2.6.3.1 Receive Multiplexer

A multiplexer is provided to allow selecting either of the two MII interfaces for the receive data that is sent to PRU. Figure 6-260 shows the symbol of receive multiplexer of PRU.

Figure 6-260. MII Receive Multiplexer

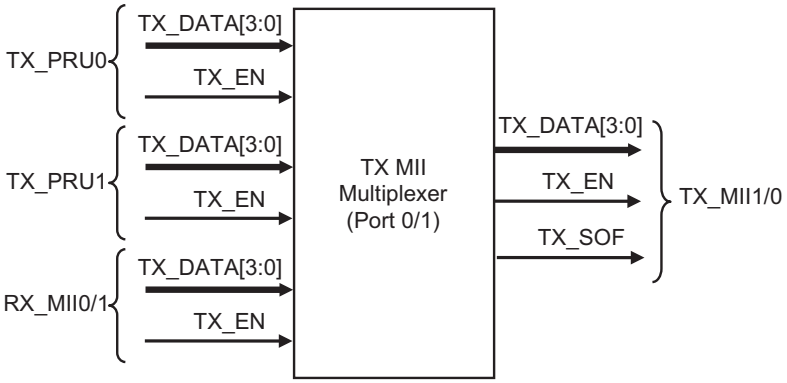


There are two receive multiplexer instances to enable selection of RX MII path for each PRU. The select lines of the RX multiplexers are driven from the PRU-ICSS programmable registers (PRUSS_MII_RT_RXCFG0/1).

6.4.9.2.6.3.2 Transmit Multiplexer

On the MII transmit ports, there is a multiplexer for each MII transmit port that enables selection of either the transmit data from the PRUs or from the RX MII interface of the other MII interface. Figure 6-261 shows the symbol of transmit multiplexer of PRU.

Figure 6-261. MII Transmit Multiplexer

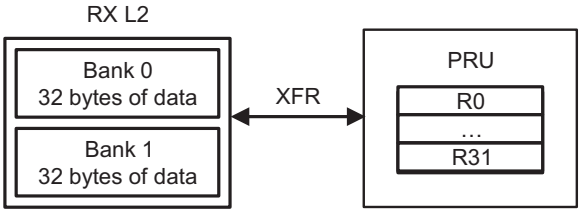


The transmit multiplexers enable the PRU-ICSS to either operate in a bypass mode where the PRU is not involved in processing MII traffic or use of one of the PRU cores for transmitting data into the MII interface. There are two instances of the TX MII multiplexer and the select lines for each TX multiplexer are provided by the PRU-ICSS. The select lines are common between register and FIFO interface. It is expected that the select lines will not change during the course of a frame so that can avoid data exchange error.

6.4.9.2.6.4 RX L2 Scratch Pad

When the RX L2 is disabled (PRUSS_MII_RT_RXCFG0/1[RX_L2_EN] = 0), the RX L2 banks can be used as a generic scratch pad. In scratch pad mode, RX L2 Bank0 and RX L2 Bank1 operate like simple write/read memory mapped registers (MMRs). All XFR size and start operations are supported. RX_RESET has no effect in this mode. This mode is shown in Figure 6-262.

Figure 6-262. Scratch Pad Mode



6.4.9.3 PRU-ICSS MII RT Registers

Table 6-664 lists the memory-mapped registers for the PRU-ICSS MII_RT module. All register offset addresses not listed in Table 6-664 should be considered as reserved locations and the register contents should not be modified.

Table 6-663. PRU-ICSS MII RT Instances

Instance	Base Address
PRU_ICSS_0_MII_RT	20AB 2000h
PRU_ICSS_1_MII_RT	20AF 2000h

Table 6-664. PRU-ICSS MII RT Registers

Offset	Acronym	Register Name	PRU_ICSS_0_MII_RT Physical Address	PRU_ICSS_1_MII_RT Physical Address	Section
0h	PRUSS_MII_RT_RXCFG0	MII RXCFG 0 Register	20AB 2000h	20AF 2000h	Section 6.4.9.3.1
4h	PRUSS_MII_RT_RXCFG1	MII RXCFG 1 Register	20AB 2004h	20AF 2004h	Section 6.4.9.3.2
10h	PRUSS_MII_RT_TXCFG0	MII TXCFG 0 Register	20AB 2010h	20AF 2010h	Section 6.4.9.3.3
14h	PRUSS_MII_RT_TXCFG1	MII TXCFG 1 Register	20AB 2014h	20AF 2014h	Section 6.4.9.3.4
20h	PRUSS_MII_RT_TX_CRC0	MII TXCRC 0 Register	20AB 2020h	20AF 2020h	Section 6.4.9.3.5
24h	PRUSS_MII_RT_TX_CRC1	MII TXCRC 1 Register	20AB 2024h	20AF 2024h	Section 6.4.9.3.6
30h	PRUSS_MII_RT_TX_IPG0	MII TXIPG 0 Register	20AB 2030h	20AF 2030h	Section 6.4.9.3.7
34h	PRUSS_MII_RT_TX_IPG1	MII TXIPG 1 Register	20AB 2034h	20AF 2034h	Section 6.4.9.3.8
38h	PRUSS_MII_RT_PRS0	MII PORT STATUS 0 Register	20AB 2038h	20AF 2038h	Section 6.4.9.3.9
3Ch	PRUSS_MII_RT_PRS1	MII PORT STATUS 1 Register	20AB 203Ch	20AF 203Ch	Section 6.4.9.3.10
40h	PRUSS_MII_RT_RX_FRMS0	MII RXFRMS 0 Register	20AB 2040h	20AF 2040h	Section 6.4.9.3.11
44h	PRUSS_MII_RT_RX_FRMS1	MII RXFRMS 1 Register	20AB 2044h	20AF 2044h	Section 6.4.9.3.12
48h	PRUSS_MII_RT_RX_PCNT0	MII RXPCNT 0 Register	20AB 2048h	20AF 2048h	Section 6.4.9.3.13
4Ch	PRUSS_MII_RT_RX_PCNT1	MII RXPCNT 1 Register	20AB 204Ch	20AF 204Ch	Section 6.4.9.3.14
50h	PRUSS_MII_RT_RX_ERR0	MII RXERR 0 Register	20AB 2050h	20AF 2050h	Section 6.4.9.3.15
54h	PRUSS_MII_RT_RX_ERR1	MII RXERR 1 Register	20AB 2054h	20AF 2054h	Section 6.4.9.3.16
60h	PRUSS_MII_RT_RXFLV0	MII RX FIFO Level 0 Register	20AB 2060h	20AF 2060h	Section 6.4.9.3.17
64h	PRUSS_MII_RT_RXFLV1	MII RX FIFO Level 1 Register	20AB 2064h	20AF 2064h	Section 6.4.9.3.18
68h	PRUSS_MII_RT_TXFLV0	MII TX FIFO Level 0 Register	20AB 2068h	20AF 2068h	Section 6.4.9.3.19
6Ch	PRUSS_MII_RT_TXFLV1	MII TX FIFO Level 1 Register	20AB 206Ch	20AF 206Ch	Section 6.4.9.3.20

6.4.9.3.1 PRUSS_MII_RT_RXCFG0 Register (Offset = 0h) [reset = 0h]

PRUSS_MII_RT_RXCFG0 is shown in Figure 6-263 and described in Table 6-666.

MII RXCFG 0 REGISTER
This register contains the PRU0 RXCFG configuration variables (RXCFG0) for the RX path.
PRUSS_MII_RT_RXCFG0 is attached to PRU0.
PRUSS_MII_RT_RXCFG0 controls which RX port is attached to PRU0.

Table 6-665. PRUSS_MII_RT_RXCFG0 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2000h
PRU_ICSS_1_MII_RT	20AF 2000h

Figure 6-263. PRUSS_MII_RT_RXCFG0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						RX_L2_EOF_SCLR_DIS	RX_ERR_RAW
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RX_SFD_RAW	RX_AUTO_FWD_PRE	RX_BYTE_SWAP	RX_L2_EN	RX_MUX_SEL	RX_CUT_PREAMBLE	RX_DATA_RDY_MODE_DIS	RX_ENABLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-666. PRUSS_MII_RT_RXCFG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	RX_L2_EOF_SCLR_DIS	R/W	0h	0h: RX_EOF flag in R31 and RXL2 is self cleared by hardware when RXL2 is enabled 1h: RX_EOF flag in R31 and RXL2 is not self cleared by hardware when RXL2 is enabled. To clear this flag, RX_EOF_CLR must be set.
8	RX_ERR_RAW	R/W	0h	0h: Error Raw Mode Disabled. RX_ERR is qualified with RX_DV, meaning RX_DV = 1 before RX_ERR action/event is generated. 1h: Error Raw Mode Enabled. RX_ERR is not qualified with RX_DV, meaning RX_ERR action/event is generated even if RX_DV = 0.
7	RX_SFD_RAW	R/W	0h	0h: SFD Raw Mode Disabled. RX_SFD requires a pattern of 5D. 1h: SFD Raw Mode Enable. The first byte of any pattern after RX_DV assertion will trigger RX_SFD event. The first nibble of the frame (RX_DV = 1) will be in the RX FIFO.

Table 6-666. PRUSS_MII_RT_RXCFG0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	RX_AUTO_FWD_PRE	R/W	0h	Enables auto-forward of received preamble. When enabled, this will forward the preamble nibbles including the SFD to the TX L1 FIFO that is attached to the PRU. First data byte seen by PRU R31 and/or RX L2 is destination address (DA). **Note: Odd number of preamble nibbles is supported in this mode. For example, 0x55D Note that new RX should only occur after the current TX completes** 0h: Disable 1h: Enable, it must disable RX_CUT_PREAMBLE and TX_AUTO_PREAMBLE
5	RX_BYTE_SWAP	R/W	0h	Defines the order of Byte0/1 placement for RX R31 and RX L2. **Note: that if TX_AUTO_SEQUENCE enabled, this bit cannot get enable since TX_BYTE_SWAP on swaps the PRU output. This bit must be selected/updated when the port is disabled or there is no traffic.** 0h: R31 [15:8]/RXL2 [15:8] = Byte1{Nibble3,Nibble2} R31[7:0]/RXL2 [7:0] = Byte0{Nibble1,Nibble0} 1h: R31 [15:8]/RXL2 [15:8] = Byte0{Nibble1,Nibble0} R31[7:0]/RXL2 [7:0] = Byte1{Nibble3,Nibble2} Nibble0 is the first nibble received.
4	RX_L2_EN	R/W	0h	Enables RX L2 buffer. 0h: Disable (RX L2 can function as generic scratch pad) 1h: Enable
3	RX_MUX_SEL	R/W	0h	Selects receive data source. Typically, the setting for this will not be identical for the two MII receive configuration registers. 0h: MII RX Data from Port 0 (default for PRUSS_MII_RT_RXCFG0) 1h: MII RX Data from Port 1 (default for PRUSS_MII_RT_RXCFG1)
2	RX_CUT_PREAMBLE	R/W	0h	Removes received preamble. 0h: All data from Ethernet PHY are passed on to PRU register. This assumes Ethernet PHY which does not shorten the preamble. 1h: MII interface suppresses preamble and sync frame delimiter. First data byte seen by PRU register is destination address.
1	RX_DATA_RDY_MODE_DIS	R/W	0h	0h: R31, Bit 16 is configured for DATA_RDY mode. 1h: R31, Bit 16 is configured for TX_EOF mode.
0	RX_ENABLE	R/W	0h	Enables the receive traffic currently selected by RX_MUX_SELECT. 0h: Disable 1h: Enable

Table 6-667. Register Call Summary for PRUSS_MII_RT_RXCFG0

PRU-ICSS MII RT Module

- Nibble and Byte Order: [0][1][2]
- PRUSS_MII_RT_RXCFG1 Register (Offset = 4h) [reset = 8h]: [0]
- RX MII Submodule Overview: [0]
- PRU and MII Port Multiplexer: [0]
- PRU-ICSS MII RT Registers: [0]
- PRUSS_MII_RT_RXCFG0 Register (Offset = 0h) [reset = 0h]: [0][1][2][3]
- RX L2 Scratch Pad: [0]

6.4.9.3.2 PRUSS_MII_RT_RXCFG1 Register (Offset = 4h) [reset = 8h]

PRUSS_MII_RT_RXCFG1 is shown in Figure 6-264 and described in Table 6-669.

MII RXCFG 1 REGISTER
This register contains the PRU1 RXCFG configuration variables (PRUSS_MII_RT_RXCFG1) for the RX path.
PRUSS_MII_RT_RXCFG1 is attached to PRU1.
PRUSS_MII_RT_RXCFG1 controls which RX port is attached to PRU1.

Table 6-668. PRUSS_MII_RT_RXCFG1 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2004h
PRU_ICSS_1_MII_RT	20AF 2004h

Figure 6-264. PRUSS_MII_RT_RXCFG1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						RX_L2_EOF_SCLR_DIS	RX_ERR_RAW
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RX_SFD_RAW	RX_AUTO_FWD_PRE	RX_BYTE_SWAP	RX_L2_EN	RX_MUX_SEL	RX_CUT_PREAMBLE	RX_DATA_RDY_MODE_DIS	RX_ENABLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-669. PRUSS_MII_RT_RXCFG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	RX_L2_EOF_SCLR_DIS	R/W	0h	0h: RX_EOF flag in R31 and RXL2 is self cleared by hardware when RXL2 is enabled 1h: RX_EOF flag in R31 and RXL2 is not self cleared by hardware when RXL2 is enabled. To clear this flag, RX_EOF_CLR must be set.
8	RX_ERR_RAW	R/W	0h	0h: Error Raw Mode Disabled. RX_ERR is qualified with RX_DV, meaning RX_DV = 1 before RX_ERR action/event is generated. 1h: Error Raw Mode Enabled. RX_ERR is not qualified with RX_DV, meaning RX_ERR action/event is generated even if RX_DV = 0.
7	RX_SFD_RAW	R/W	0h	0h: SFD Raw Mode Disabled. RX_SFD requires a pattern of 5D. 1h: SFD Raw Mode Enable. The first byte of any pattern after RX_DV assertion will trigger RX_SFD event. The first nibble of the frame (RX_DV = 1) will be in the RX FIFO.

Table 6-669. PRUSS_MII_RT_RXCFG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	RX_AUTO_FWD_PRE	R/W	0h	Enables auto-forward of received preamble. When enabled, this will forward the preamble nibbles including the SFD to the TX L1 FIFO that is attached to the PRU. First data byte seen by PRU R31 and/or RX L2 is destination address (DA). **Note: Odd number of preamble nibbles is supported in this mode. For example, 0x55D Note that new RX should only occur after the current TX completes** 0h: Disable 1h: Enable, it must disable RX_CUT_PREAMBLE and TX_AUTO_PREAMBLE
5	RX_BYTE_SWAP	R/W	0h	Defines the order of Byte0/1 placement for RX R31 and RX L2. **Note: that if TX_AUTO_SEQUENCE enabled, this bit cannot get enable since TX_BYTE_SWAP on swaps the PRU output. This bit must be selected/updated when the port is disabled or there is no traffic.** 0h: R31 [15:8]/RXL2 [15:8] = Byte1{Nibble3,Nibble2} R31[7:0]/RXL2 [7:0] = Byte0{Nibble1,Nibble0} 1h: R31 [15:8]/RXL2 [15:8] = Byte0{Nibble1,Nibble0} R31[7:0]/RXL2 [7:0] = Byte1{Nibble3,Nibble2} Nibble0 is the first nibble received.
4	RX_L2_EN	R/W	0h	Enables RX L2 buffer. 0h: Disable (RX L2 can function as generic scratch pad) 1h: Enable
3	RX_MUX_SEL	R/W	1h	Selects receive data source. Typically, the setting for this will not be identical for the two MII receive configuration registers. 0h: MII RX Data from Port 0 (default for PRUSS_MII_RT_RXCFG0) 1h: MII RX Data from Port 1 (default for PRUSS_MII_RT_RXCFG1)
2	RX_CUT_PREAMBLE	R/W	0h	Removes received preamble. 0h: All data from Ethernet PHY are passed on to PRU register. This assumes Ethernet PHY which does not shorten the preamble. 1h: MII interface suppresses preamble and sync frame delimiter. First data byte seen by PRU register is destination address.
1	RX_DATA_RDY_MODE_DIS	R/W	0h	0h: R31, Bit 16 is configured for DATA_RDY mode. 1h: R31, Bit 16 is configured for TX_EOF mode.
0	RX_ENABLE	R/W	0h	Enables the receive traffic currently selected by RX_MUX_SELECT. 0h: Disable 1h: Enable

Table 6-670. Register Call Summary for PRUSS_MII_RT_RXCFG1

PRU-ICSS MII RT Module

- PRU-ICSS MII RT Registers: [0]
- PRUSS_MII_RT_RXCFG0 Register (Offset = 0h) [reset = 0h]: [0]
- PRUSS_MII_RT_RXCFG1 Register (Offset = 4h) [reset = 8h]: [0][1][2][3][4]
- RX MII Submodule Overview: [0]

6.4.9.3.3 PRUSS_MII_RT_TXCFG0 Register (Offset = 10h) [reset = 00400100h]

PRUSS_MII_RT_TXCFG0 is shown in Figure 6-265 and described in Table 6-672.

MII TXCFG 0 REGISTER
This register contains the configuration variables for the transmit path on the MII interface port 0.
PRUSS_MII_RT_TXCFG0 is attached to Port TX0.
PRUSS_MII_RT_TXCFG0 controls which PRU is selected for TX0

Table 6-671. PRUSS_MII_RT_TXCFG0 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2010h
PRU_ICSS_1_MII_RT	20AF 2010h

Figure 6-265. PRUSS_MII_RT_TXCFG0 Register

31	30	29	28	27	26	25	24
RESERVED	TX_CLK_DELAY			RESERVED		TX_START_DELAY	
R-0h	R/W-0h			R-0h		R/W-40h	
23	22	21	20	19	18	17	16
TX_START_DELAY							
R/W-40h							
15	14	13	12	11	10	9	8
RESERVED				TX_32_MODE_EN	PRE_TX_AUTO_ESC_ERR	PRE_TX_AUTO_SEQUENCE	TX_MUX_SEL
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-1h
7	6	5	4	3	2	1	0
RESERVED				TX_BYTE_SWAP	TX_EN_MODE	TX_AUTO_PREAMBLE	TX_ENABLE
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-672. PRUSS_MII_RT_TXCFG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30-28	TX_CLK_DELAY	R/W	0h	In order to guarantee the MII_RT IO timing values published in the device data manual, the ICSS_i_VCLK_CLK (where i = 0 or 1) clock must be configured for 200MHz and TX_CLK_DELAY must be set to 6h.
27-26	RESERVED	R	0h	Reserved

Table 6-672. PRUSS_MII_RT_TXCFG0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-16	TX_START_DELAY	R/W	40h	Defines the minimum time interval (delay) between receiving the RXDV for the current frame and the start of the transmit interface sending data to the MII interface. Delay value is in units of MII_RT clock cycles, which uses the ICSS_i_VCLK_CLK (default is 200MHz, or 5ns). Default TX_START_DELAY value is 320ns, which is optimized for minimum latency at 16 bit processing. Counter is started with RX_DV signal going active. Transmit interface stops sending data when no more data is written into transmit interface by PRU along with TX_EOF marker bit set. If the TX FIFO has data when the delay expires, then TX will start sending data. But if the TX FIFO is empty, it will not start until the TX FIFO is not empty. It is possible to overflow the TX FIFO with the max delay setting when auto-forwarding is enabled since the time delay is larger than the amount of data it needs to store. As long as TX L1 FIFO overflows, software will need to issue a TX_RESET to reset the TX FIFO. The total delay is 96-byte times (size of TX FIFO), but you need to allow delays for synchronization. Do to this fact, the maximum delay should be 80ns less when auto forwarding is enabled. Therefore, 0x3F0 is the maximum in this configuration.
15-12	RESERVED	R	0h	Reserved
11	TX_32_MODE_EN	R/W	0h	0h: Disable 32-bit Data Push mode. 1h: Enable 32-bit, 16-bit, and 8-bit Data Push mode with TX_MASK disabled. In this mode, the internal PRU R30 byte write strobes are used and not the R31 CMD TX_PUSH mode. Any update to R30 will trigger an TX PUSH. See Table 6-658.
10	RESERVED	R	0h	Reserved
9	TX_AUTO_SEQUENCE	R/W	0h	Enables transmit auto-sequence. Note the transmit data source is determined by TX_MUX_SEL setting. 0h: Disable 1h: Enable, transmit state machine based on events on receiver path that is connected to the respective transmitter. Also, the masking logic is disabled and only the MII data is used.
8	TX_MUX_SEL	R/W	1h	Selects transmit data source. The default/reset setting for TX Port 0 is 1. This setting permits MII TX Port 0 to receive data from PRU1 and the MII TX Port 1 which is connected to PRU0 by default. 0h: Data from PRU0 (default for PRUSS_MII_RT_TXCFG1) 1h: Data from PRU1 (default for PRUSS_MII_RT_TXCFG0)
7-4	RESERVED	R	0h	Reserved

Table 6-672. PRUSS_MII_RT_TXCFG0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	TX_BYTE_SWAP	R/W	0h	Defines the order of Byte0/1 placement for TX R30. This bit must be selected/updated when the port is disabled or there is no traffic. **0h: If PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 0,** R30[15:8] = Byte1{Nibble3,Nibble2} R30[7:0] = Byte0{Nibble1,Nibble0} R30[31:24] = TX_MASK[15:8] R30[23:16] = TX_MASK[7:0] **If PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 1,** R30[31:24] = Byte3{Nibble7,Nibble6} R30[23:16] = Byte2{Nibble5,Nibble4} R30[15:8] = Byte1{Nibble3,Nibble2} R30[7:0] = Byte0{Nibble1,Nibble0} **1h: If PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 0,** R30[15:8] = Byte0{Nibble1,Nibble0} R30[7:0] = Byte1{Nibble3,Nibble2} R30[31:24] = TX_MASK[7:0] R30[23:16] = TX_MASK[15:8] **If PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 1,** (ONLY SUPPORT 32bit push) R30[31:24] = Byte0{Nibble1,Nibble0} R30[23:16] = Byte1{Nibble3,Nibble2} R30[15:8] = Byte2{Nibble5,Nibble4} R30[7:0] = Byte3{Nibble7,Nibble6 **Note Nibble0 is the first nibble received.**
2	TX_EN_MODE	R/W	0h	Enables transmit self clear on TX_EOF event. **Note that iep.cmp[3] must be set before transmission will start for TX0, and iep_cmp[4] for TX1. This is a new dependency, in addition to TX L1 FIFO not empty and TX_START_DELAY expiration, to start transmission.** 0h: Disable 1h: Enable, TX_ENABLE will be clear for a TX_EOF event by itself.
1	TX_AUTO_PREAMBLE	R/W	0h	Transmit data auto-preamble. 0h: PRU will provide full preamble 1h: TX FIFO will insert pre-amble automatically **Note: the TX FIFO does not get preloaded with the preamble until the first write occurs. This can cause the latency to be larger the min latency.**
0	TX_ENABLE	R/W	0h	Enables transmit traffic on TX PORT. If TX_EN_MODE is set, then TX_ENABLE will self clear during a TX_EOF event. Note Software can use this to pre-fill the TX FIFO and then start the TX frame during non-ECS operations. 0h: TX PORT is disabled/stopped immediately 1h: TX PORT is enabled and the frame will start once the IPG counter expired and TX Start Delay counter has expired

Table 6-673. Register Call Summary for PRUSS_MII_RT_TXCFG0

PRU-ICSS Industrial Ethernet Peripheral (IEP) • Industrial Ethernet Mapping: [0][1]
PRU-ICSS MII RT Module • PRU R31 Command Interface: [0][1] • TX Data Path Options to TX L1 FIFO: [0][1][2][3][4][5] • Introduction: [0] • PRU-ICSS MII RT Registers: [0] • PRUSS_MII_RT_TXCFG0 Register (Offset = 10h) [reset = 00400100h]: [0][1][2][3][4][5][6][7] • Nibble and Byte Order: [0][1][2][3][4][5][6] • PRUSS_MII_RT_TXCFG1 Register (Offset = 14h) [reset = 00400000h]: [0][1][2][3][4]

6.4.9.3.4 PRUSS_MII_RT_TXCFG1 Register (Offset = 14h) [reset = 00400000h]

PRUSS_MII_RT_TXCFG1 is shown in Figure 6-266 and described in Table 6-675.

MII TXCFG 1 REGISTER
This register contains the configuration variables for the transmit path on the MII interface port 1.
PRUSS_MII_RT_TXCFG1 is attached to Port TX1.
PRUSS_MII_RT_TXCFG1 controls which PRU is selected for TX1

Table 6-674. PRUSS_MII_RT_TXCFG1 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2014h
PRU_ICSS_1_MII_RT	20AF 2014h

Figure 6-266. PRUSS_MII_RT_TXCFG1 Register

31	30	29	28	27	26	25	24
RESERVED	TX_CLK_DELAY			RESERVED		TX_START_DELAY	
R-0h	R/W-0h			R-0h		R/W-40h	
23	22	21	20	19	18	17	16
TX_START_DELAY							
R/W-40h							
15	14	13	12	11	10	9	8
RESERVED				TX_32_MODE_EN	PRE_TX_AUTO_ESC_ERR	PRE_TX_AUTO_SEQUENCE	TX_MUX_SEL
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED				TX_BYTE_SWAP	TX_EN_MODE	TX_AUTO_PREAMBLE	TX_ENABLE
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-675. PRUSS_MII_RT_TXCFG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30-28	TX_CLK_DELAY	R/W	0h	In order to guarantee the MII_RT IO timing values published in the device data manual, the ICSS_i_VCLK_CLK (where i = 0 or 1) clock must be configured for 200MHz and TX_CLK_DELAY must be set to 6h.
27-26	RESERVED	R	0h	Reserved

Table 6-675. PRUSS_MII_RT_TXCFG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-16	TX_START_DELAY	R/W	40h	Defines the minimum time interval (delay) between receiving the RXDV for the current frame and the start of the transmit interface sending data to the MII interface. Delay value is in units of MII_RT clock cycles, which uses the ICSS_i_VCLK_CLK, where i = 0 or 1 (default is 200MHz, or 5ns). Default TX_START_DELAY value is 320ns, which is optimized for minimum latency at 16 bit processing. Counter is started with RX_DV signal going active. Transmit interface stops sending data when no more data is written into transmit interface by PRU along with TX_EOF marker bit set. If the TX FIFO has data when the delay expires, then TX will start sending data. But if the TX FIFO is empty, it will not start until the TX FIFO is not empty. It is possible to overflow the TX FIFO with the max delay setting when auto-forwarding is enabled since the time delay is larger than the amount of data it needs to store. As long as TX L1 FIFO overflows, software will need to issue a TX_RESET to reset the TX FIFO. The total delay is 96-byte times (size of TX FIFO), but you need to allow delays for synchronization. Do to this fact, the maximum delay should be 80ns less when auto forwarding is enabled. Therefore, 0x3F0 is the maximum in this configuration.
15-12	RESERVED	R	0h	Reserved
11	TX_32_MODE_EN	R/W	0h	0h: Disable 32-bit Data Push mode. 1h: Enable 32-bit, 16-bit, and 8-bit Data Push mode with TX_MASK disabled. In this mode, the internal PRU R30 byte write strobes are used and not the R31 CMD TX_PUSH mode. Any update to R30 will trigger an TX PUSH.
10	RESERVED	R	0h	Reserved
9	TX_AUTO_SEQUENCE	R/W	0h	Enables transmit auto-sequence. Note the transmit data source is determined by TX_MUX_SEL setting. 0h: Disable 1h: Enable, transmit state machine based on events on receiver path that is connected to the respective transmitter. Also, the masking logic is disabled and only the MII data is used.
8	TX_MUX_SEL	R/W	0h	Selects transmit data source. The default/reset setting for TX Port 0 is 1. This setting permits MII TX Port 0 to receive data from PRU1 and the MII TX Port 1 which is connected to PRU0 by default. 0h: Data from PRU0 (default for PRUSS_MII_RT_TXCFG1) 1h: Data from PRU1 (default for PRUSS_MII_RT_TXCFG0)
7-4	RESERVED	R	0h	Reserved

Table 6-675. PRUSS_MII_RT_TXCFG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	TX_BYTE_SWAP	R/W	0h	Defines the order of Byte0/1 placement for TX R30. This bit must be selected/updated when the port is disabled or there is no traffic. 0h: If PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 0, R30[15:8] = Byte1{Nibble3,Nibble2} R30[7:0] = Byte0{Nibble1,Nibble0} R30[31:24] = TX_MASK[15:8] R30[23:16] = TX_MASK[7:0] If PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 1, R30[31:24] = Byte3{Nibble7,Nibble6} R30[23:16] = Byte2{Nibble5,Nibble4} R30[15:8] = Byte1{Nibble3,Nibble2} R30[7:0] = Byte0{Nibble1,Nibble0} 1h: If PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 0, R30[15:8] = Byte0{Nibble1,Nibble0} R30[7:0] = Byte1{Nibble3,Nibble2} R30[31:24] = TX_MASK[7:0] R30[23:16] = TX_MASK[15:8] If PRUSS_MII_RT_TXCFG0/1 [TX_32_MODE_EN] = 1, (ONLY SUPPORT 32bit push) R30[31:24] = Byte0{Nibble1,Nibble0} R30[23:16] = Byte1{Nibble3,Nibble2} R30[15:8] = Byte2{Nibble5,Nibble4} R30[7:0] = Byte3{Nibble7,Nibble6 **Note Nibble0 is the first nibble received.**
2	TX_EN_MODE	R/W	0h	Enables transmit self clear on TX_EOF event. **Note that iep.cmp[3] must be set before transmission will start for TX0, and iep_cmp[4] for TX1. This is a new dependency, in addition to TX L1 FIFO not empty and TX_START_DELAY expiration, to start transmission.** 0h: Disable 1h: Enable, TX_ENABLE will be clear for a TX_EOF event by itself.
1	TX_AUTO_PREAMBLE	R/W	0h	Transmit data auto-preamble. 0h: PRU will provide full preamble 1h: TX FIFO will insert pre-amble automatically **Note: the TX FIFO does not get preloaded with the preamble until the first write occurs. This can cause the latency to be larger the min latency.**
0	TX_ENABLE	R/W	0h	Enables transmit traffic on TX PORT. If TX_EN_MODE is set, then TX_ENABLE will self clear during a TX_EOF event. Note Software can use this to pre-fill the TX FIFO and then start the TX frame during non-ECS operations. 0h: TX PORT is disabled/stopped immediately 1h: TX PORT is enabled and the frame will start once the IPG counter expired and TX Start Delay counter has expired

Table 6-676. Register Call Summary for PRUSS_MII_RT_TXCFG1

PRU-ICSS MII RT Module
• PRU-ICSS MII RT Registers: [0] • PRUSS_MII_RT_TXCFG0 Register (Offset = 10h) [reset = 00400100h]: [0] • PRUSS_MII_RT_TXCFG1 Register (Offset = 14h) [reset = 00400000h]: [0][1][2][3]

6.4.9.3.5 PRUSS_MII_RT_TX_CRC0 Register (Offset = 20h) [reset = 0h]

PRUSS_MII_RT_TX_CRC0 is shown in Figure 6-267 and described in Table 6-678.

MII TXCRC 0 REGISTER
It contains CRC32 which PRU0 reads

Table 6-677. PRUSS_MII_RT_TX_CRC0 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2020h
PRU_ICSS_1_MII_RT	20AF 2020h

Figure 6-267. PRUSS_MII_RT_TX_CRC0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX_CRC																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-678. PRUSS_MII_RT_TX_CRC0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TX_CRC	R	0h	FCS (CRC32) data can be read by PRU for diagnostics. It is only valid after 6 clocks after a TX_CRC_HIGH command is given.

Table 6-679. Register Call Summary for PRUSS_MII_RT_TX_CRC0

PRU-ICSS MII RT Module
- CRC Computation: [0]
- PRU R31 Command Interface: [0]
- PRU-ICSS MII RT Registers: [0]
- PRUSS_MII_RT_TX_CRC0 Register (Offset = 20h) [reset = 0h]: [0]

6.4.9.3.6 PRUSS_MII_RT_TX_CRC1 Register (Offset = 24h) [reset = 0h]

PRUSS_MII_RT_TX_CRC1 is shown in Figure 6-268 and described in Table 6-681.

MII TXCRC 1 REGISTER
It contains CRC32 which PRU1 reads

Table 6-680. PRUSS_MII_RT_TX_CRC1 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2024h
PRU_ICSS_1_MII_RT	20AF 2024h

Figure 6-268. PRUSS_MII_RT_TX_CRC1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX_CRC																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-681. PRUSS_MII_RT_TX_CRC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TX_CRC	R	0h	FCS (CRC32) data can be read by PRU for diagnostics. It is only valid after 6 clocks after a TX_CRC_HIGH command is given.

Table 6-682. Register Call Summary for PRUSS_MII_RT_TX_CRC1

PRU-ICSS MII RT Module
- PRU-ICSS MII RT Registers: [0]
- PRUSS_MII_RT_TX_CRC1 Register (Offset = 24h) [reset = 0h]: [0]

6.4.9.3.7 PRUSS_MII_RT_TX_IPG0 Register (Offset = 30h) [reset = 28h]

PRUSS_MII_RT_TX_IPG0 is shown in Figure 6-269 and described in Table 6-684.

MII TXIPG 0 REGISTER

Table 6-683. PRUSS_MII_RT_TX_IPG0 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2030h
PRU_ICSS_1_MII_RT	20AF 2030h

Figure 6-269. PRUSS_MII_RT_TX_IPG0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						TX_IPG									
R-0h																						R/W-28h									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-684. PRUSS_MII_RT_TX_IPG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9-0	TX_IPG	R/W	28h	Defines the minimum of transmit Inter Packet Gap (IPG) which is the number of ICSS_i_VCLK_CLK (where i = 0 or 1) cycles between the de-assertion of TX_EN and the assertion of TX_EN. The start of the TX will get delayed when the incoming packet IPG is less than defined minimum value. In general, software should program in increments of 8, 40ns to insure the extra delays takes effect.

Table 6-685. Register Call Summary for PRUSS_MII_RT_TX_IPG0

PRU-ICSS MII RT Module
- PRU-ICSS MII RT Registers: [0]
- PRUSS_MII_RT_TX_IPG0 Register (Offset = 30h) [reset = 28h]: [0]

6.4.9.3.8 PRUSS_MII_RT_TX_IPG1 Register (Offset = 34h) [reset = 28h]

PRUSS_MII_RT_TX_IPG1 is shown in Figure 6-270 and described in Table 6-687.

MII TXIPG 1 REGISTER

Table 6-686. PRUSS_MII_RT_TX_IPG1 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2034h
PRU_ICSS_1_MII_RT	20AF 2034h

Figure 6-270. PRUSS_MII_RT_TX_IPG1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						TX_IPG									
R-0h																						R/W-28h									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-687. PRUSS_MII_RT_TX_IPG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9-0	TX_IPG	R/W	28h	Defines the minimum of transmit Inter Packet Gap (IPG) which is the number of ICSS_i_VCLK_CLK (where i = 0 or 1) cycles between the de-assertion of TX_EN and the assertion of TX_EN. The start of the TX will get delayed when the incoming packet IPG is less than defined minimum value. In general, software should program in increments of 8, 40ns to insure the extra delays takes effect.

Table 6-688. Register Call Summary for PRUSS_MII_RT_TX_IPG1

PRU-ICSS MII RT Module
- PRU-ICSS MII RT Registers: [0]
- PRUSS_MII_RT_TX_IPG1 Register (Offset = 34h) [reset = 28h]: [0]

6.4.9.3.9 PRUSS_MII_RT_PRS0 Register (Offset = 38h) [reset = 0h]

PRUSS_MII_RT_PRS0 is shown in Figure 6-271 and described in Table 6-690.

MII PORT STATUS 0 REGISTER

Table 6-689. PRUSS_MII_RT_PRS0 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2038h
PRU_ICSS_1_MII_RT	20AF 2038h

Figure 6-271. PRUSS_MII_RT_PRS0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						MII_CRS	MII_COL
R-0h						R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 6-690. PRUSS_MII_RT_PRS0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	MII_CRS	R	0h	Read the current state of pr1_mii0_crs
0	MII_COL	R	0h	Read the current state of pr1_mii0_col

Table 6-691. Register Call Summary for PRUSS_MII_RT_PRS0

PRU-ICSS MII RT Module
- PRU-ICSS MII RT Registers: [0]
- PRUSS_MII_RT_PRS0 Register (Offset = 38h) [reset = 0h]: [0]

6.4.9.3.10 PRUSS_MII_RT_PRS1 Register (Offset = 3Ch) [reset = 0h]

PRUSS_MII_RT_PRS1 is shown in Figure 6-272 and described in Table 6-693.

MII PORT STATUS 1 REGISTER

Table 6-692. PRUSS_MII_RT_PRS1 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 203Ch
PRU_ICSS_1_MII_RT	20AF 203Ch

Figure 6-272. PRUSS_MII_RT_PRS1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						MII_CRS	MII_COL
R-0h						R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 6-693. PRUSS_MII_RT_PRS1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	MII_CRS	R	0h	Read the current state of pr1_mii1_crs
0	MII_COL	R	0h	Read the current state of pr1_mii1_col

Table 6-694. Register Call Summary for PRUSS_MII_RT_PRS1

PRU-ICSS MII RT Module

- PRU-ICSS MII RT Registers: [0]
- PRUSS_MII_RT_PRS1 Register (Offset = 3Ch) [reset = 0h]: [0]

6.4.9.3.11 PRUSS_MII_RT_RX_FRMS0 Register (Offset = 40h) [reset = 05F1003Fh]

PRUSS_MII_RT_RX_FRMS0 is shown in Figure 6-273 and described in Table 6-696.

MII RXFRMS 0 REGISTER

Table 6-695. PRUSS_MII_RT_RX_FRMS0 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2040h
PRU_ICSS_1_MII_RT	20AF 2040h

Figure 6-273. PRUSS_MII_RT_RX_FRMS0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX_MAX_FRM																RX_MIN_FRM															
R/W-5F1h																R/W-3Fh															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-696. PRUSS_MII_RT_RX_FRMS0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RX_MAX_FRM	R/W	5F1h	Defines the maximum received frame count. If the total byte count of received frame is more than defined value, RX_MAX_FRM_ERR will get set. 0h = 1 byte after SFD and including CRC N= N+1 bytes after SFD and including CRC. Note if the incoming frame is truncated at the marker, RX_CRC and RX_NIBBLE_ODD will not get asserted.
15-0	RX_MIN_FRM	R/W	3Fh	Defines the minimum received frame count. If the total byte count of received frame is less than defined value, RX_MIN_FRM_ERR will get set. 0h = 1 byte after SFD and including CRC N=N+1 bytes after SFD and including CRC

Table 6-697. Register Call Summary for PRUSS_MII_RT_RX_FRMS0

PRU-ICSS MII RT Module
- PRU-ICSS MII RT Registers: [0]
- PRUSS_MII_RT_RX_FRMS0 Register (Offset = 40h) [reset = 05F1003Fh]: [0]

6.4.9.3.12 PRUSS_MII_RT_RX_FRMS1 Register (Offset = 44h) [reset = 05F1003Fh]

PRUSS_MII_RT_RX_FRMS1 is shown in Figure 6-274 and described in Table 6-699.

MII RXFRMS 1 REGISTER

Table 6-698. PRUSS_MII_RT_RX_FRMS1 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2044h
PRU_ICSS_1_MII_RT	20AF 2044h

Figure 6-274. PRUSS_MII_RT_RX_FRMS1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX_MAX_FRM																RX_MIN_FRM															
R/W-5F1h																R/W-3Fh															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-699. PRUSS_MII_RT_RX_FRMS1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RX_MAX_FRM	R/W	5F1h	Defines the maximum received frame count. If the total byte count of the received frame is more than defined value, RX_MAX_FRM_ERR will get set. 0h = 1 byte after SFD and including CRC N= N+1 bytes after SFD and including CRC Note if the incoming frame is truncated at the marker, RX_CRC and RX_NIBBLE_ODD will not get asserted.
15-0	RX_MIN_FRM	R/W	3Fh	Defines the minimum received frame count. If the total byte count of received frame is less than defined value, RX_MIN_FRM_ERR will get set. 0h = 1 byte after SFD and including CRC N=N+1 bytes after SFD and including CRC

Table 6-700. Register Call Summary for PRUSS_MII_RT_RX_FRMS1

PRU-ICSS MII RT Module
- PRU-ICSS MII RT Registers: [0]
- PRUSS_MII_RT_RX_FRMS1 Register (Offset = 44h) [reset = 05F1003Fh]: [0]

6.4.9.3.13 PRUSS_MII_RT_RX_PCNT0 Register (Offset = 48h) [reset = E1h]

PRUSS_MII_RT_RX_PCNT0 is shown in Figure 6-275 and described in Table 6-702.

MII RXPCNT 0 REGISTER

Table 6-701. PRUSS_MII_RT_RX_PCNT0 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2048h
PRU_ICSS_1_MII_RT	20AF 2048h

Figure 6-275. PRUSS_MII_RT_RX_PCNT0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RX_MAX_PCNT				RX_MIN_PCNT			
R-0h								R/W-Eh				R/W-1h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-702. PRUSS_MII_RT_RX_PCNT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-4	RX_MAX_PCNT	R/W	Eh	Defines the maximum number of nibbles until the start of frame delimiter (SFD) event occurred (i.e. matches 0xD5). RX_MAX_PRE_COUNT_ERR will be set if the preamble counts more than the value of RX_MAX_PCNT. If the SFD does not occur within 16 nibbles, the error will assert and the incoming frame will be truncated. 0h: Disabled 1h: Reserved 2h: 4th nibble needs to have built 0xD5 Eh: 16th nibble needs to have built 0xD5 **Note the 16th nibble is transmitted.** **Note for firmware enabling preamble error detection, it is recommended to keep RX_MAX_PCNT disabled (0x0). Otherwise, hardware can truncate a valid frame with too long of a preamble.**
3-0	RX_MIN_PCNT	R/W	1h	Defines the minimum number of nibbles until the start of frame delimiter (SFD) event occurred, which is matched the value 0xD5. RX_MIN_PRE_COUNT_ERR will be set if the preamble counts less than the value of RX_MIN_PCNT. 0h: Disabled 1h: 1 0x5 before 0xD5 2h: 2 0x5 before 0xD5 N min of N 0x5 before 0xD5 **Note it does not need to be "0x5"**

Table 6-703. Register Call Summary for PRUSS_MII_RT_RX_PCNT0

PRU-ICSS MII RT Module
- PRU-ICSS MII RT Registers: [0]
- PRUSS_MII_RT_RX_PCNT0 Register (Offset = 48h) [reset = E1h]: [0]
- RX MII Submodule Overview: [0]

6.4.9.3.14 PRUSS_MII_RT_RX_PCNT1 Register (Offset = 4Ch) [reset = E1h]

PRUSS_MII_RT_RX_PCNT1 is shown in Figure 6-276 and described in Table 6-705.

MII RXPCNT 1 REGISTER

Table 6-704. PRUSS_MII_RT_RX_PCNT1 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 204Ch
PRU_ICSS_1_MII_RT	20AF 204Ch

Figure 6-276. PRUSS_MII_RT_RX_PCNT1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RX_MAX_PCNT				RX_MIN_PCNT			
R-0h								R/W-Eh				R/W-1h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-705. PRUSS_MII_RT_RX_PCNT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-4	RX_MAX_PCNT	R/W	Eh	Defines the maximum number of nibbles until the start of frame delimiter (SFD) event occurred (i.e. matches 0xD5). RX_MAX_PRE_COUNT_ERR will be set if the preamble counts more than the value of RX_MAX_PCNT. If the SFD does not occur within 16 nibbles, the error will assert and the incoming frame will be truncated. 0h: Disabled 1h: Reserved 2h: 4th nibble needs to have built 0xD5 Eh: 16th nibble needs to have built 0xD5 **Note the 16th nibble is transmitted** Note for firmware enabling preamble error detection, it is recommended to keep RX_MAX_PCNT disabled (0x0). Otherwise, hardware can truncate a valid frame with too long of a preamble.
3-0	RX_MIN_PCNT	R/W	1h	Defines the minimum number of nibbles until the start of frame delimiter (SFD) event occurred, which is matched the value 0xD5. RX_MIN_PRE_COUNT_ERR will be set if the preamble counts less than the value of RX_MIN_PCNT. 0h Disabled 1h: 1 0x5 before 0xD5 2h: 2 0x5 before 0xD5 N: N 0x5 before 0xD5 **Note it does not need to be "0x5"**

Table 6-706. Register Call Summary for PRUSS_MII_RT_RX_PCNT1

PRU-ICSS MII RT Module
- PRU-ICSS MII RT Registers: [0]
- PRUSS_MII_RT_RX_PCNT1 Register (Offset = 4Ch) [reset = E1h]: [0]

6.4.9.3.15 PRUSS_MII_RT_RX_ERR0 Register (Offset = 50h) [reset = 0h]

PRUSS_MII_RT_RX_ERR0 is shown in Figure 6-277 and described in Table 6-708.

MII RXERR 0 REGISTER

Table 6-707. PRUSS_MII_RT_RX_ERR0 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2050h
PRU_ICSS_1_MII_RT	20AF 2050h

Figure 6-277. PRUSS_MII_RT_RX_ERR0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RX_MAX_FRM_ERR	RX_MIN_FRM_ERR	RX_MAX_PCNT_ERR	RX_MIN_PCNT_ERR
R-0h				RWr1Clr-0h	RWr1Clr-0h	RWr1Clr-0h	RWr1Clr-0h

LEGEND: R = Read Only; RWr1Clr = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 6-708. PRUSS_MII_RT_RX_ERR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	RX_MAX_FRM_ERR	RWr1Clr	0h	Error status of received frame is more than the value of RX_MAX_FRM. 0h: No error occurred 1h: Error occurred Write 1 to Clear
2	RX_MIN_FRM_ERR	RWr1Clr	0h	Error status of received frame is less than the value of RX_MIN_FRM. 0h: No error occurred 1h: Error occurred Write 1 to Clear
1	RX_MAX_PCNT_ERR	RWr1Clr	0h	Error status of received preamble nibble is more than the value of RX_MAX_PCNT. 0h: No error occurred 1h: Error occurred Write 1 to Clear
0	RX_MIN_PCNT_ERR	RWr1Clr	0h	Error status of received preamble nibble is less than the value of RX_MIN_PCNT. 0h: No error occurred 1h: Error occurred Write 1 to Clear

Table 6-709. Register Call Summary for PRUSS_MII_RT_RX_ERR0

PRU-ICSS MII RT Module
• PRU-ICSS MII RT Registers: [0] • PRUSS_MII_RT_RX_ERR0 Register (Offset = 50h) [reset = 0h]: [0]

6.4.9.3.16 PRUSS_MII_RT_RX_ERR1 Register (Offset = 54h) [reset = 0h]

PRUSS_MII_RT_RX_ERR1 is shown in Figure 6-278 and described in Table 6-711.

MII RXERR 1 REGISTER

Table 6-710. PRUSS_MII_RT_RX_ERR1 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2054h
PRU_ICSS_1_MII_RT	20AF 2054h

Figure 6-278. PRUSS_MII_RT_RX_ERR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RX_MAX_FRM_ERR	RX_MIN_FRM_ERR	RX_MAX_PCNT_ERR	RX_MIN_PCNT_ERR
R-0h				RWr1Clr-0h	RWr1Clr-0h	RWr1Clr-0h	RWr1Clr-0h

LEGEND: R = Read Only; RWr1Clr = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 6-711. PRUSS_MII_RT_RX_ERR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	RX_MAX_FRM_ERR	RWr1Clr	0h	Error status of received frame is more than the value of RX_MAX_FRM_CNT. 0h: No error occurred 1h: Error occurred Write 1 to Clear
2	RX_MIN_FRM_ERR	RWr1Clr	0h	Error status of received frame is less than the value of RX_MIN_FRM_CNT. 0h: No error occurred 1h: Error occurred Write 1 to Clear
1	RX_MAX_PCNT_ERR	RWr1Clr	0h	Error status of received preamble nibble is more than the value of RX_MAX_PCNT. 0h: No error occurred 1h: Error occurred Write 1 to Clear
0	RX_MIN_PCNT_ERR	RWr1Clr	0h	Error status of received preamble nibble is less than the value of RX_MIN_PCNT. 0h: No error occurred 1h: Error occurred Write 1 to Clear

Table 6-712. Register Call Summary for PRUSS_MII_RT_RX_ERR1

PRU-ICSS MII RT Module
• PRU-ICSS MII RT Registers: [0] • PRUSS_MII_RT_RX_ERR1 Register (Offset = 54h) [reset = 0h]: [0]

6.4.9.3.17 PRUSS_MII_RT_RXFLV0 Register (Offset = 60h) [reset = 0h]

PRUSS_MII_RT_RXFLV0 is shown in Figure 6-279 and described in Table 6-714.

MII PRUSS_MII_RT_RXFLV0 REGISTER
This register defines the number of valid bytes in the RX FIFO MII interface port 0.
PRUSS_MII_RT_RXFLV0 is attached to Port RX0.
PRUSS_MII_RT_RXFLV0 controls which PRU is selected for RX0

Table 6-713. PRUSS_MII_RT_RXFLV0 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2060h
PRU_ICSS_1_MII_RT	20AF 2060h

Figure 6-279. PRUSS_MII_RT_RXFLV0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RX_FIFO_LEVEL							
R-0h								RWr1Clr-0h							

LEGEND: R = Read Only; RWr1Clr = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 6-714. PRUSS_MII_RT_RXFLV0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	RX_FIFO_LEVEL	RWr1Clr	0h	Define the number of valid bytes in the RX FIFO. 0h: Empty 1h: 1 Byte/ 2 Nibbles 2h: 2 Byte/ 4 Nibbles .... 32h: 32 Bytes/ 64 Nibbles

Table 6-715. Register Call Summary for PRUSS_MII_RT_RXFLV0

PRU-ICSS MII RT Module
- PRU-ICSS MII RT Registers: [0]
- PRUSS_MII_RT_RXFLV0 Register (Offset = 60h) [reset = 0h]: [0][1][2][3]

6.4.9.3.18 PRUSS_MII_RT_RXFLV1 Register (Offset = 64h) [reset = 0h]

PRUSS_MII_RT_RXFLV1 is shown in Figure 6-280 and described in Table 6-717.

MII PRUSS_MII_RT_RXFLV1 REGISTER
This register defines the number of valid bytes in the RX FIFO MII interface port 1.
PRUSS_MII_RT_RXFLV1 is attached to Port RX1.
PRUSS_MII_RT_RXFLV1 controls which PRU is selected for RX1

Table 6-716. PRUSS_MII_RT_RXFLV1 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2064h
PRU_ICSS_1_MII_RT	20AF 2064h

Figure 6-280. PRUSS_MII_RT_RXFLV1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RX_FIFO_LEVEL							
R-0h								RWr1Clr-0h							

LEGEND: R = Read Only; RWr1Clr = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 6-717. PRUSS_MII_RT_RXFLV1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	RX_FIFO_LEVEL	RWr1Clr	0h	Define the number of valid bytes in the RX FIFO. 0h: Empty 1h: 1 Byte/ 2 Nibbles 2h: 2 Byte/ 4 Nibbles .... 32h: 32 Bytes/ 64 Nibbles

Table 6-718. Register Call Summary for PRUSS_MII_RT_RXFLV1

PRU-ICSS MII RT Module
- PRU-ICSS MII RT Registers: [0]
- PRUSS_MII_RT_RXFLV1 Register (Offset = 64h) [reset = 0h]: [0][1][2][3]

6.4.9.3.19 PRUSS_MII_RT_TXFLV0 Register (Offset = 68h) [reset = 0h]

PRUSS_MII_RT_TXFLV0 is shown in Figure 6-281 and described in Table 6-720.

MII PRUSS_MII_RT_TXFLV0 REGISTER
This register defines the number of valid bytes in the TX FIFO MII interface port 0.
PRUSS_MII_RT_TXFLV0 is attached to Port TX0.
PRUSS_MII_RT_TXFLV0 controls which PRU is selected for TX0.

Table 6-719. PRUSS_MII_RT_TXFLV0 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 2068h
PRU_ICSS_1_MII_RT	20AF 2068h

Figure 6-281. PRUSS_MII_RT_TXFLV0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								TX_FIFO_LEVEL							
R-0h								RWr1Clr-0h							

LEGEND: R = Read Only; RWr1Clr = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 6-720. PRUSS_MII_RT_TXFLV0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	TX_FIFO_LEVEL	RWr1Clr	0h	Define the number of valid nibbles in the TX FIFO. 0h: Empty 1h: 1 Nibble 2h: 1 Byte/ 2 Nibbles .... 128h: 64 Bytes/ 128 Nibbles .... 192h: 96 Bytes/ 192 Nibbles

Table 6-721. Register Call Summary for PRUSS_MII_RT_TXFLV0

PRU-ICSS MII RT Module
- PRU-ICSS MII RT Registers: [0]
- PRUSS_MII_RT_TXFLV0 Register (Offset = 68h) [reset = 0h]: [0][1][2][3]

6.4.9.3.20 PRUSS_MII_RT_TXFLV1 Register (Offset = 6Ch) [reset = 0h]

PRUSS_MII_RT_TXFLV1 is shown in Figure 6-282 and described in Table 6-723.

MII PRUSS_MII_RT_TXFLV1 REGISTER
This register defines the number of valid bytes in the TX FIFO MII interface port 1.
PRUSS_MII_RT_TXFLV1 is attached to Port TX1.
PRUSS_MII_RT_TXFLV1 controls which PRU is selected for TX1.

Table 6-722. PRUSS_MII_RT_TXFLV1 Instances

Instance	Physical Address
PRU_ICSS_0_MII_RT	20AB 206Ch
PRU_ICSS_1_MII_RT	20AF 206Ch

Figure 6-282. PRUSS_MII_RT_TXFLV1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								TX_FIFO_LEVEL							
R-0h								RWr1Clr-0h							

LEGEND: R = Read Only; RWr1Clr = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 6-723. PRUSS_MII_RT_TXFLV1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	TX_FIFO_LEVEL	RWr1Clr	0h	Define the number of valid nibbles in the TX FIFO. 0h: Empty 1h: 1 Nibble 2h: 1 Byte/ 2 Nibble .... 128h: 64 Bytes/ 128 Nibbles .... 192h: 96 Bytes/ 192 Nibbles

Table 6-724. Register Call Summary for PRUSS_MII_RT_TXFLV1

PRU-ICSS MII RT Module
- PRU-ICSS MII RT Registers: [0]
- PRUSS_MII_RT_TXFLV1 Register (Offset = 6Ch) [reset = 0h]: [0][1][2][3]

6.4.10 PRU-ICSS MII MDIO Module

This section describes the PRU-ICSS_0 and PRU-ICSS_1 integrated **MII management interface module - MII_MDIO** module (ICSS_0_MII_MDIO / ICSS_1_MII_MDIO, respectively).

6.4.10.1 PRU-ICSS MII MDIO Overview

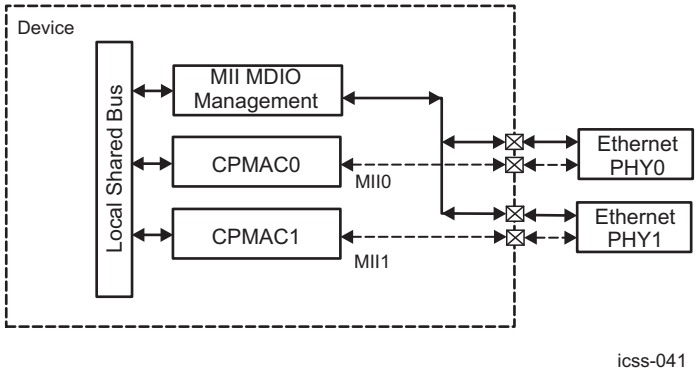
The following features are supported:

- Supports up to 32 PHY addresses.
- Two user access registers to control and monitor up to two PHYs simultaneously.
- Slave interface for configuration and control (MII RT MDIO CFG)

The PRU-ICSS MII MDIO management I/F module implements the **802.3 serial management interface** to interrogate and control two Ethernet PHYs simultaneously using a shared two-wire bus. Figure 1 shows a device with two MACs, each connected to an Ethernet PHY, being managed by the MII interface module using a shared bus.

The Figure 6-283 gives an overview of the MII MDIO management interface.

Figure 6-283. Device PRU-ICSS MII MDIO Management Interface Overview



6.4.10.2 PRU-ICSS MII MDIO Functional Description

The MII Management interface incorporates:

- **MDIO Registers** - Host interaction with this module is facilitated through the registers in this block.
- **Control and Schedule** - The control and register logic in the MII Management Interface module contain the state machine and scheduling logic which control the wire side operation.
- **MDIO Interface** - The MDIO interface block provides the serial interface to the MDIO interface.

The MDIO logic is fully synchronous to the PRU-ICSS local shared bus clock.

6.4.10.2.1 MII MDIO Management Interface Frame Formats

The below Table 6-725 shows the read and write format of the 32-bit MII Management interface frames, respectively.

Table 6-725. MII MDIO Frame Formats

Pre-amble	Start Delimiter	Operation Code	PHY Address	Register Address	Turnaround	Data
MDIO Read Frame Format						
FFFFFFFFh	01	10	AAAAA	RRRRR	Z0	DDDD.DDDD.DDDD.DDDD
MDIO Write Frame Format						
FFFFFFFFh	01	00	AAAAA	RRRRR	10	DDDD.DDDD.DDDD.DDDD

The default or idle state of the two wire serial interface is a logic one. All tri-state drivers should be disabled and the PHY's pull-up resistor will pull the **MDIO** line to a logic one. Prior to initiating any other transaction, the station management entity shall send a preamble sequence of 32 contiguous logic one bits on the **MDIO** line with 32 corresponding cycles on **MDCLK** to provide the PHY with a pattern that it can use to establish synchronization. A PHY shall observe a sequence of 32 contiguous logic one bits on **MDIO** with 32 corresponding **MDCLK** cycles before it responds to any other transaction.

Preamble
The start of a frame is indicated by a preamble, which consists of a sequence of 32 contiguous bits all of which are a "1". This sequence provides the Ethernet PHY a pattern to use to establish synchronization.

Start Delimiter
The preamble is followed by the start delimiter which is indicated by a "01" pattern. The pattern assures transitions from the default logic one state to zero and back to one.

Operation Code
The operation code for a read is "10", while the operation code for a write is a "00".

Ethernet PHY Address
The PHY address is 5 bits allowing 32 unique values. The first bit transmitted is the MSB of the PHY address.

Register Address
The Register address is 5 bits allowing 32 registers to be addressed within each PHY. Refer to the 10/100 PHY address map for addresses of individual registers.

Turnaround
An idle bit time during which no device actively drives the MDIO signal shall be inserted between the register address field and the data field of a read frame in order to avoid contention. During a read frame, the PHY shall drive a zero bit onto MDIO for the first bit time following the idle bit and preceding the Data field. During a write frame, this field shall consist of a one bit followed by a zero bit.

Data
The Data field is 16 bits. The first bit transmitted and received is the MSB of the data word.

The Table 6-726 shows the PRU-ICSS_0 / PRU-ICSS_1 MII MDIO signals and their availability at the device boundary.

Table 6-726. PRU-ICSS MII MDIO Control and Interface Signals

MDIO Control Signals			
Pin Name	**Type**	**Available as device I/O**	**Function**
MDIO_LINKINT[1:0]	O	N.A.	Serial interface link change interrupt. Indicates a change in the state of the PHY link.
MDIO_USERINT[1:0]	O	N.A.	Serial interface user command event complete interrupt.
MDIO Interface Signals			
Pin Name	**Type**	**Available as device I/O**	**Function**
MDIO_I	I	device bidirectioanal **pr0_mdio_data** and pr1_mdio_mdclk pin in input mode	Serial data input
MDIO_O	O	device bidirectional **pr0_mdio_data** and pr1_mdio_mdclk pin in output mode	Serial data output
MDIO_OE_N	O	N.A.	Serial data output enable. Asserted "0" when data output is valid
MDCLK_O	O	device output - **pr0_mdio_mdclk** pr1_mdio_mdclk	Serial clock output
MLINK_I[1:0]	I	N.A.	Optional link status inputs from PHY. Each input is connected to a single PHY. Unused inputs are tied '0'.

6.4.10.2.2 PRU-ICSS MII MDIO Interractions

The MII Management I/F will remain idle until enabled by setting the **[30] ENABLE** bit in the PRUSS_MII_MDIO_CONTROL register. The MII Management I/F will then continuously poll the link status from within the Generic Status Register of all possible 32 PHY addresses in turn recording the results in the PRUSS_MII_MDIO_LINK register. The link status of two of the 32 possible PHY addresses can also be determined using the **MLINK** pin inputs. The bit **[7] LINKSEL** in the PRUSS_MII_MDIO_USERPHYSEL0/1 register determines the status input that is used. A change in the link status of the two PHYs being monitored will set the appropriate bit in the PRUSS_MII_MDIO_LINKINTRAW register and the PRUSS_MII_MDIO_LINKINTMASKED register, if enabled by the **[6]LINKINT_ENABLE**bit in the PRUSS_MII_MDIO_USERPHYSEL0/1 register.

The PRUSS_MII_MDIO_ALIVE register is updated by the MII Management I/F module if the PHY acknowledged the read of the generic status register. In addition, any PHY register read transactions initiated by the host also cause the PRUSS_MII_MDIO_ALIVE register to be updated.

At any time, the host can define a transaction for the MII Management interface module to undertake using the **[15:0] DATA**, **[20:16] PHYADR**, **[25:21] REGADR**, and **[30] WRITE** fields in a PRUSS_MII_MDIO_USERACCESS0/1 register. When the host sets the **[31] GO** bit in this register, the MII Management interface module will begin the transaction without any further intervention from the host. Upon completion, the MII Management interface will clear the **[31] GO** bit and set the **[1:0] USERINTRAW** bit field in the PRUSS_MII_MDIO_USERINTRAW register corresponding to the PRUSS_MII_MDIO_USERACCESS0/1 register being used. The corresponding bit in the PRUSS_MII_MDIO_USERINTMASKED register may also be set depending on the mask setting in the PRUSS_MII_MDIO_USERINTMASKSET and PRUSS_MII_MDIO_USERINTMASKCLR registers. A round-robin arbitration scheme is used to schedule transactions which may queued by the host in different PRUSS_MII_MDIO_USERACCESS0/1 registers. The host should check the status of the **[31] GO** bit in the PRUSS_MII_MDIO_USERACCESS0/1 register before initiating a new transaction to ensure that the previous transaction has completed. The host can use the**[29] ACK** bit in the PRUSS_MII_MDIO_USERACCESS0/1 register to determine the status of a read transaction.

It is necessary for software to use the MII Management interface module to set up the autonegotiation parameters of each PHY attached to a MAC port, retrieve the negotiation results, and set up the **MAC Control** register in the corresponding MAC.

6.4.10.2.3 PRU-ICSS MII MDIO Interrupts

The MII Management interface state machine will assert the **MDIO_LINKINT** signals if there is a change in the link state of the Ethernet PHY corresponding to the address in the **[4-0] PHYADR_MON** field of the PRUSS_MII_MDIO_USERPHYSEL0/1 registers and the corresponding **[6] LINKINT_ENABLE** bit is set. The **MDIO_LINKINT** event is also captured in the PRUSS_MII_MDIO_LINKINTMASKED register. **MDIO_LINKINT[0]** and **MDIO_LINKINT[1]** correspond to the PRUSS_MII_MDIO_USERPHYSEL0 and PRUSS_MII_MDIO_USERPHYSEL1 registers, respectively.

When the **[31] GO** bit in the PRUSS_MII_MDIO_USERACCESS0/1 registers transitions from ‘1’ to ‘0’, indicating the completion of a user access, and the corresponding **[1:0] USERINTMASKEDSET** field in the

PRUSS_MII_MDIO_USERINTMASKSET register is set, the **MDIO_USERINT** signal is asserted ‘1’. The **MDIO_USERINT** event is also captured in the PRUSS_MII_MDIO_USERINTMASKSET register. **MDIO_USERINT[0]** and **MDIO_USERINT[1]** correspond to the PRUSS_MII_MDIO_USERACCESS0 and PRUSS_MII_MDIO_USERACCESS1 registers, respectively.

6.4.10.3 PRU-ICSS MII MDIO Receive/Transmit Frame Host Software Interface

To facilitate transmission and reception of serial management frames, the host has to perform the following operations:

- Configure the **[20] PREAMBLE** and **[15:0] CLKDIV** fields in the PRUSS_MII_MDIO_CONTROL register.
- Enable the VBUS MII management module by setting the **[30] ENABLE** bit in the PRUSS_MII_MDIO_CONTROL register. If Byte access is being used, the **[30] ENABLE** bit should be written last.

- The PRUSS_MII_MDIO_ALIVE register can be read after a delay to determine which Ethernet PHYs responded.
- Set up the appropriate PHY addresses in the PRUSS_MII_MDIO_USERPHYSEL0/1[4-0] PHYADR_MON bits.
- Set up the appropriate **[6] LINKINT_ENABLE** bit in the PRUSS_MII_MDIO_USERPHYSEL0/1 registers.
- Set up the appropriate **[7] LINKSEL** bits in the PRUSS_MII_MDIO_USERPHYSEL0/1 registers.
- Set up the appropriate **[1:0] USERINTMASKEDSET** field in the PRUSS_MII_MDIO_USERINTMASKSET register.
- To write to an Ethernet PHY register the host should first check to ensure that the **[31] GO** bit in a PRUSS_MII_MDIO_USERACCESS0/1 register is cleared. The **[31] GO**, **[30] WRITE**, **[25:21] REGADR**, **[20:16] PHYADR** and data fields in that PRUSS_MII_MDIO_USERACCESS0/1 registers can then be updated to be appropriate value. If byte access is being used, the **[31] GO** bit should be written last. The write operation to the PHY will be scheduled and completed by the module. Completion of the write operation can be determined by examining the **[31] GO** bit in the PRUSS_MII_MDIO_USERACCESS0/1 registers. It also results in a transition on the appropriate **MDIO_INT** signal and the corresponding bit in the PRUSS_MII_MDIO_USERINTMASKED register based on the setting of the PRUSS_MII_MDIO_USERINTMASKSET register.
- To read from an Ethernet PHY register the host should first check to ensure that the **[31] GO** bit in a PRUSS_MII_MDIO_USERACCESS0/1 register is cleared. The **[31] GO**, **[25:21] REGADR**, and **[20:16] PHYADR** fields in that PRUSS_MII_MDIO_USERACCESS0/1 register can then be updated to the appropriate value. The read data value will be available in the **[15:0] DATA** field of the PRUSS_MII_MDIO_USERACCESS0/1 register after the module completes the read operation on the serial bus. The completion of the read operation can be determined by examining the **[31] GO** and **[29] ACK** bits in the PRUSS_MII_MDIO_USERACCESS0/1 register. It also results in a transition on the appropriate **MDIO_INT** signal and the corresponding bit in the PRUSS_MII_MDIO_USERINTMASKED register based on the setting of the PRUSS_MII_MDIO_USERINTMASKSET register.
- The module de-asserts the **MDIO_USERINT** signal when the host writes to the appropriate **[1:0] USERINTMASKED** bit in the PRUSS_MII_MDIO_USERINTMASKED register or the **[1:0] USERINTRAW** bit in the PRUSS_MII_MDIO_USERINTRAW register.
- The host can poll the PRUSS_MII_MDIO_LINK register periodically or use the **MDIO_LINKINT** signals to determine the state of the serial interface to a particular Ethernet PHY.
- The module de-asserts the **MDIO_LINKINT** when the host writes to the appropriate **[1:0] LINKINTRAW** bit in the PRUSS_MII_MDIO_LINKINTRAW register or the **[1:0] LINKINTMASKED** bit in the PRUSS_MII_MDIO_LINKINTMASKED register.

Table 6-727. Summary of the PRU-ICSS MII MDIO Functional Registers

Address Offset	Register Mnemonic	Register Name	Register Purpose
00h	**MDIOVer**	PRUSS_MII_MDIO_VER	Module version register
04h	**MDIOControl**	PRUSS_MII_MDIO_CONTROL	Module control register
08h	**MDIOAlive**	PRUSS_MII_MDIO_ALIVE	Ethernet PHY acknowledge status register
0Ch	**MDIOLink**	PRUSS_MII_MDIO_LINK	Ethernet PHY link status register
10h	**MDIOLinkIntRaw**	PRUSS_MII_MDIO_LINKINTRAW	Link status change interrupt register (raw value)
14h	**MDIOLinkIntMasked**	PRUSS_MII_MDIO_LINKINTMASKED	Link status change interrupt register (masked value)
18h-1Ch	**Reserved**	-	Reserved
20h	**MDIOUserIntRaw**	PRUSS_MII_MDIO_USERINTRAW	User command complete interrupt register (raw value)

Table 6-727. Summary of the PRU-ICSS MII MDIO Functional Registers (continued)

Address Offset	Register Mnemonic	Register Name	Register Purpose
24h	**MDIOUserIntMasked**	PRUSS_MII_MDIO_USERINTMASKED	User command complete interrupt register (masked value)
28h	**MDIOUserIntMaskSet**	PRUSS_MII_MDIO_USERINTMASKSET	User interrupt mask set register
2Ch	**MDIOUserIntMaskClr**	PRUSS_MII_MDIO_USERINTMASKCLR	User interrupt mask clear register
30h – 7Ch	**Reserved**	-	Reserved
80h	**MDIOUserAccess0**	PRUSS_MII_MDIO_USERACCESS0	User access register 0
84h	**MDIOUserPhySel0**	PRUSS_MII_MDIO_USERPHYSEL0	User PHY select register 0
88h	**MDIOUserAccess1**	PRUSS_MII_MDIO_USERACCESS1	User access register 1
8Ch	**MDIOUserPhySel1**	PRUSS_MII_MDIO_USERPHYSEL1	User PHY select register 1
90h – FFh	**Reserved**	-	Reserved

6.4.10.4 PRU-ICSS MII MDIO Registers

Table 6-729 lists the memory-mapped registers for the PRU-ICSS MII MDIO. All register offset addresses not listed in Table 6-729 should be considered as reserved locations and the register contents should not be modified.

Table 6-728. PRU-ICSS MII MDIO Instances

Instance	Base Address
PRU_ICSS_0_MII_MDIO	20AB 2400h
PRU_ICSS_1_MII_MDIO	20AF 2400h

Table 6-729. PRU-ICSS MII MDIO Registers

Offset	Acronym	Register Name	PRU_ICSS_0_MII_MDIO Physical Address	PRU_ICSS_1_MII_MDIO Physical Address	Section
0h	PRUSS_MII_MDIO_VER	MDIO MODULE VERSION REGISTER	20AB 2400h	20AF 2400h	Section 6.4.10.4.1
4h	PRUSS_MII_MDIO_CONTROL	MDIO MODULE CONTROL REGISTER	20AB 2404h	20AF 2404h	Section 6.4.10.4.2
8h	PRUSS_MII_MDIO_ALIVE	PHY ACKNOWLEDGE STATUS REGISTER	20AB 2408h	20AF 2408h	Section 6.4.10.4.3
Ch	PRUSS_MII_MDIO_LINK	PHY LINK STATUS REGISTER	20AB 240Ch	20AF 240Ch	Section 6.4.10.4.4
10h	PRUSS_MII_MDIO_LINKINTRAW	LINK STATUS CHANGE INTERRUPT REGISTER (RAW VALUE)	20AB 2410h	20AF 2410h	Section 6.4.10.4.5
14h	PRUSS_MII_MDIO_LINKINTMASKED	LINK STATUS CHANGE INTERRUPT REGISTER (MASKED VALUE)	20AB 2414h	20AF 2414h	Section 6.4.10.4.6
20h	PRUSS_MII_MDIO_USERINTRAW	USER COMMAND COMPLETE INTERRUPT REGISTER (RAW VALUE)	20AB 2420h	20AF 2420h	Section 6.4.10.4.7
24h	PRUSS_MII_MDIO_USERINTMASKED	USER COMMAND COMPLETE INTERRUPT REGISTER (MASKED VALUE)	20AB 2424h	20AF 2424h	Section 6.4.10.4.8
28h	PRUSS_MII_MDIO_USERINTMASKSET	USER INTERRUPT MASK SET REGISTER	20AB 2428h	20AF 2428h	Section 6.4.10.4.9
2Ch	PRUSS_MII_MDIO_USERINTMASKCLR	USER INTERRUPT MASK CLEAR REGISTER	20AB 242Ch	20AF 242Ch	Section 6.4.10.4.10
80h	PRUSS_MII_MDIO_USERACCESS0	USER ACCESS REGISTER0	20AB 2480h	20AF 2480h	Section 6.4.10.4.11
84h	PRUSS_MII_MDIO_USERPHYSEL0	USER PHY SELECT REGISTER0	20AB 2484h	20AF 2484h	Section 6.4.10.4.12
88h	PRUSS_MII_MDIO_USERACCESS1	USER ACCESS REGISTER1	20AB 2488h	20AF 2488h	Section 6.4.10.4.13
8Ch	PRUSS_MII_MDIO_USERPHYSEL1	USER PHY SELECT REGISTER1	20AB 248Ch	20AF 248Ch	Section 6.4.10.4.14

6.4.10.4.1 PRUSS_MII_MDIO_VER Register (Offset = 0h) [reset = -h]

PRUSS_MII_MDIO_VER is shown in Figure 6-284 and described in Table 6-731.

MDIO MODULE VERSION REGISTER

Table 6-730. PRUSS_MII_MDIO_VER Instances

Instance	Physical Address
PRU_ICSS_0_MII_MDIO	20AB 2400h
PRU_ICSS_1_MII_MDIO	20AF 2400h

Figure 6-284. PRUSS_MII_MDIO_VER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REVISION																															
R--h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-731. PRUSS_MII_MDIO_VER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REVISION	R	-h	IP Revision.

Table 6-732. Register Call Summary for PRUSS_MII_MDIO_VER

PRU-ICSS MII MDIO Module
- PRU-ICSS MII MDIO Receive/Transmit Frame Host Software Interface: [0]
- PRUSS_MII_MDIO_VER Register (Offset = 0h) [reset = -h]: [0]
- PRU-ICSS MII MDIO Registers: [0]

6.4.10.4.2 PRUSS_MII_MDIO_CONTROL Register (Offset = 4h) [reset = 810000FFh]

PRUSS_MII_MDIO_CONTROL is shown in Figure 6-285 and described in Table 6-734.

MDIO MODULE CONTROL REGISTER

Table 6-733. PRUSS_MII_MDIO_CONTROL Instances

Instance	Physical Address
PRU_ICSS_0_MII_MDIO	20AB 2404h
PRU_ICSS_1_MII_MDIO	20AF 2404h

Figure 6-285. PRUSS_MII_MDIO_CONTROL Register

31	30	29	28	27	26	25	24
IDLE	ENABLE	RESERVED	HIGHEST_USER_CHANNEL				
R-1h	R/W-0h	R-0h	R-1h				
23	22	21	20	19	18	17	16
RESERVED			PREAMBLE	FAULT	FAULT_DETECT_ENABLE	INT_TEST_ENABLE	RESERVED
R-0h			R/W-0h	RWr1Clr-0h	R/W-0h	R/W-0h	R-0h
15	14	13	12	11	10	9	8
CLKDIV							
R/W-FFh							
7	6	5	4	3	2	1	0
CLKDIV							
R/W-FFh							

LEGEND: R = Read Only; R/W = Read/Write; RWr1Clr = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 6-734. PRUSS_MII_MDIO_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IDLE	R	1h	MDIO state machine IDLE. Set to 1 when the state machine is in the idle state.
30	ENABLE	R/W	0h	Enable control. Writing a 1 to this bit enables the MDIO state machine, writing a 0 disables it. If the MDIO state machine is active at the time it is disabled, it will complete the current operation before halting and setting the idle bit. If using byte access, the enable bit has to be the last bit written in this register.
29	RESERVED	R	0h	Reserved
28-24	HIGHEST_USER_CHANNEL	R	1h	Highest user channel. This field specifies the highest useraccess channel that is available in the module and is currently set to 1. This implies that **MDIOUserAccess1** is the highest available user access channel.
23-21	RESERVED	R	0h	Reserved
20	PREAMBLE	R/W	0h	Preamble disable. Writing a 1 to this bit disables this device from sending MDIO frame preambles.
19	FAULT	RWr1Clr	0h	Fault indicator. This bit is set to 1 if the MDIO pins fail to read back what the device is driving onto them. This indicates a physical layer fault and the module state machine is reset. Writing a 1 to it clears this bit.
18	FAULT_DETECT_ENABLE	R/W	0h	Fault detect enable. This bit has to be set to 1 to enable the physical layer fault detection.
17	INT_TEST_ENABLE	R/W	0h	Interrupt test enable. This bit can be set to 1 to enable the host to set the userint and linkint bits for test purposes.
16	RESERVED	R	0h	Reserved

Table 6-734. PRUSS_MII_MDIO_CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-0	CLKDIV	R/W	FFh	Clock Divider. This field specifies the division ratio between CLK and the frequency of MDCLK. MDCLK is disabled when clkdiv is set to 0. MDCLK frequency = clk frequency/(clkdiv+1).

Table 6-735. Register Call Summary for PRUSS_MII_MDIO_CONTROL

PRU-ICSS MII MDIO Module
- PRU-ICSS MII MDIO Interractions: [0]
- PRU-ICSS MII MDIO Receive/Transmit Frame Host Software Interface: [0][1][2]
- PRUSS_MII_MDIO_CONTROL Register (Offset = 4h) [reset = 810000FFh]: [0]
- PRU-ICSS MII MDIO Registers: [0]

6.4.10.4.3 PRUSS_MII_MDIO_ALIVE Register (Offset = 8h) [reset = 0h]

PRUSS_MII_MDIO_ALIVE is shown in Figure 6-286 and described in Table 6-737.

PHY ACKNOWLEDGE STATUS REGISTER

Table 6-736. PRUSS_MII_MDIO_ALIVE Instances

Instance	Physical Address
PRU_ICSS_0_MII_MDIO	20AB 2408h
PRU_ICSS_1_MII_MDIO	20AF 2408h

Figure 6-286. PRUSS_MII_MDIO_ALIVE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALIVE																															
RWr1Clr-0h																															

LEGEND: RWr1Clr = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 6-737. PRUSS_MII_MDIO_ALIVE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ALIVE	RWr1Clr	0h	MDIO Alive bitfield. Each of the 32 bits of this register is set if the most recent access to the PHY with address corresponding to the register bit number was acknowledged by the PHY, the bit is reset if the PHY fails to acknowledge the access. Both the user and polling accesses to a PHY will cause the corresponding alive bit to be updated. The alive bits are only meant to be used to give an indication of the presence or not of a PHY with the corresponding address. Writing a 1 to any bit will clear it, writing a 0 has no effect.

Table 6-738. Register Call Summary for PRUSS_MII_MDIO_ALIVE

PRU-ICSS MII MDIO Module

- PRU-ICSS MII MDIO Interractions: [0][1]
- PRU-ICSS MII MDIO Receive/Transmit Frame Host Software Interface: [0][1]
- PRU-ICSS MII MDIO Registers: [0]
- PRUSS_MII_MDIO_ALIVE Register (Offset = 8h) [reset = 0h]: [0]

6.4.10.4.4 PRUSS_MII_MDIO_LINK Register (Offset = Ch) [reset = 0h]

PRUSS_MII_MDIO_LINK is shown in Figure 6-287 and described in Table 6-740.

PHY LINK STATUS REGISTER

Table 6-739. PRUSS_MII_MDIO_LINK Instances

Instance	Physical Address
PRU_ICSS_0_MII_MDIO	20AB 240Ch
PRU_ICSS_1_MII_MDIO	20AF 240Ch

Figure 6-287. PRUSS_MII_MDIO_LINK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LINK																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-740. PRUSS_MII_MDIO_LINK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LINK	R	0h	MDIO Link state. This register is updated after a read of the Generic Status Register of a PHY. The bit is set if the PHY with the corresponding address has link and the PHY acknowledges the read transaction. The bit is reset if the PHY indicates it does not have link or fails to acknowledge the read transaction. Writes to the register have no effect. In addition, the status of the two PHYs specified in the **MDIOUserPhySel** registers can be determined using the **MLINK** input pins. This is determined by the **linksel** bit in the **MDIOUserPhySel** register.

Table 6-741. Register Call Summary for PRUSS_MII_MDIO_LINK

PRU-ICSS MII MDIO Module
- PRU-ICSS MII MDIO Interractions: [0]
- PRU-ICSS MII MDIO Receive/Transmit Frame Host Software Interface: [0][1]
- PRU-ICSS MII MDIO Registers: [0]
- PRUSS_MII_MDIO_LINK Register (Offset = Ch) [reset = 0h]: [0]

6.4.10.4.5 PRUSS_MII_MDIO_LINKINTRAW Register (Offset = 10h) [reset = 0h]

PRUSS_MII_MDIO_LINKINTRAW is shown in Figure 6-288 and described in Table 6-743.

LINK STATUS CHANGE INTERRUPT REGISTER (RAW VALUE)

Table 6-742. PRUSS_MII_MDIO_LINKINTRAW Instances

Instance	Physical Address
PRU_ICSS_0_MII_MDIO	20AB 2410h
PRU_ICSS_1_MII_MDIO	20AF 2410h

Figure 6-288. PRUSS_MII_MDIO_LINKINTRAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						LINKINTRAW	
R-0h						RWr1Clr-0h	

LEGEND: R = Read Only; RWr1Clr = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 6-743. PRUSS_MII_MDIO_LINKINTRAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	LINKINTRAW	RWr1Clr	0h	MDIO link change event, raw value. When asserted 1 , a bit indicates that there was an **MDIO link** change event (i.e. change in the MDIOLink register) corresponding to the PHY address in the **MDIOUserPhySel** register. **linkintraw[0] and linkintraw[1]** correspond to **MDIOUserPhySel0 and MDIOUserPhysel1**, respectively. Writing a 1 will clear the event and writing 0 has no effect.**If the int_test bit in the MDIOControl register is set, the host may set the linkintraw bits to a 1 .**This mode may be used for test purposes.

Table 6-744. Register Call Summary for PRUSS_MII_MDIO_LINKINTRAW

PRU-ICSS MII MDIO Module

- PRU-ICSS MII MDIO Interractions: [0]
- PRUSS_MII_MDIO_LINKINTRAW Register (Offset = 10h) [reset = 0h]: [0]
- PRU-ICSS MII MDIO Receive/Transmit Frame Host Software Interface: [0][1]
- PRU-ICSS MII MDIO Registers: [0]

6.4.10.4.6 PRUSS_MII_MDIO_LINKINTMASKED Register (Offset = 14h) [reset = 0h]

PRUSS_MII_MDIO_LINKINTMASKED is shown in Figure 6-289 and described in Table 6-746.

LINK STATUS CHANGE INTERRUPT REGISTER (MASKED VALUE)

Table 6-745. PRUSS_MII_MDIO_LINKINTMASKED Instances

Instance	Physical Address
PRU_ICSS_0_MII_MDIO	20AB 2414h
PRU_ICSS_1_MII_MDIO	20AF 2414h

Figure 6-289. PRUSS_MII_MDIO_LINKINTMASKED Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						LINKINTMASKED	
R-0h						RWr1Clr-0h	

LEGEND: R = Read Only; RWr1Clr = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 6-746. PRUSS_MII_MDIO_LINKINTMASKED Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	LINKINTMASKED	RWr1Clr	0h	MDIO link change interrupt, masked value. When asserted 1 , a bit indicates that there was an MDIO link change event (i.e. change in the MDIOLink register) corresponding to the PHY address in the MDIOUserPhySel register and the corresponding linkint_enable bit was set. linkintmasked[0] and linkintmasked[1] correspond to MDIOUserPhySel0 and MDIOUserPhysel1, respectively. Writing a 1 will clear the interrupt and writing 0 has no effect. If the int_test bit in the MDIOControl register is set, the host may set the linkint bits to a 1 . This mode may be used for test purposes.

Table 6-747. Register Call Summary for PRUSS_MII_MDIO_LINKINTMASKED

PRU-ICSS MII MDIO Module

- PRU-ICSS MII MDIO Interractions: [0]
- PRU-ICSS MII MDIO Interrupts: [0]
- PRU-ICSS MII MDIO Receive/Transmit Frame Host Software Interface: [0][1]
- PRU-ICSS MII MDIO Registers: [0]
- PRUSS_MII_MDIO_LINKINTMASKED Register (Offset = 14h) [reset = 0h]: [0]

6.4.10.4.7 PRUSS_MII_MDIO_USERINTRAW Register (Offset = 20h) [reset = 0h]

PRUSS_MII_MDIO_USERINTRAW is shown in Figure 6-290 and described in Table 6-749.

USER COMMAND COMPLETE INTERRUPT REGISTER (RAW VALUE)

Table 6-748. PRUSS_MII_MDIO_USERINTRAW Instances

Instance	Physical Address
PRU_ICSS_0_MII_MDIO	20AB 2420h
PRU_ICSS_1_MII_MDIO	20AF 2420h

Figure 6-290. PRUSS_MII_MDIO_USERINTRAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						USERINTRAW	
R-0h						RWr1Clr-0h	

LEGEND: R = Read Only; RWr1Clr = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 6-749. PRUSS_MII_MDIO_USERINTRAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	USERINTRAW	RWr1Clr	0h	Raw value of MDIO user command complete event for MDIOUserAccess1 through MDIOUserAccess0, respectively. When asserted 1 , a bit indicates that the previously scheduled PHY read or write command using that particular MDIOUserAccess register has completed. Writing a 1 will clear the event and writing 0 has no effect. . If the int_test bit in the MDIOControl register is set, the host may set the userintraw bits to a 1 . This mode may be used for test purposes.

Table 6-750. Register Call Summary for PRUSS_MII_MDIO_USERINTRAW

PRU-ICSS MII MDIO Module

- PRU-ICSS MII MDIO Interractions: [0]
- PRUSS_MII_MDIO_USERINTRAW Register (Offset = 20h) [reset = 0h]: [0]
- PRU-ICSS MII MDIO Receive/Transmit Frame Host Software Interface: [0][1]
- PRU-ICSS MII MDIO Registers: [0]

6.4.10.4.8 PRUSS_MII_MDIO_USERINTMASKED Register (Offset = 24h) [reset = 0h]

PRUSS_MII_MDIO_USERINTMASKED is shown in Figure 6-291 and described in Table 6-752.

USER COMMAND COMPLETE INTERRUPT REGISTER (MASKED VALUE)

Table 6-751. PRUSS_MII_MDIO_USERINTMASKED Instances

Instance	Physical Address
PRU_ICSS_0_MII_MDIO	20AB 2424h
PRU_ICSS_1_MII_MDIO	20AF 2424h

Figure 6-291. PRUSS_MII_MDIO_USERINTMASKED Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						USERINTMASKED	
R-0h						RWr1Clr-0h	

LEGEND: R = Read Only; RWr1Clr = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 6-752. PRUSS_MII_MDIO_USERINTMASKED Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	USERINTMASKED	RWr1Clr	0h	Masked value of MDIO user command complete interrupt for MDIOUserAccess1 through MDIOUserAccess0, respectively. When asserted 1 , a bit indicates that the previously scheduled PHY read or write command using that particular MDIOUserAccess register has completed and the corresponding userintmaskset bit is set to 1 . . Writing a 1 will clear the interrupt and writing 0 has no effect. . If the int_test bit in the MDIOControl register is set, the host may set the userintmasked bits to a 1 . This mode may be used for test purposes.

Table 6-753. Register Call Summary for PRUSS_MII_MDIO_USERINTMASKED

PRU-ICSS MII MDIO Module

- PRU-ICSS MII MDIO Interractions: [0]
- PRUSS_MII_MDIO_USERINTMASKED Register (Offset = 24h) [reset = 0h]: [0]
- PRU-ICSS MII MDIO Receive/Transmit Frame Host Software Interface: [0][1][2][3]
- PRU-ICSS MII MDIO Registers: [0]

6.4.10.4.9 PRUSS_MII_MDIO_USERINTMASKSET Register (Offset = 28h) [reset = 0h]

PRUSS_MII_MDIO_USERINTMASKSET is shown in Figure 6-292 and described in Table 6-755.

USER INTERRUPT MASK SET REGISTER

Table 6-754. PRUSS_MII_MDIO_USERINTMASKSET Instances

Instance	Physical Address
PRU_ICSS_0_MII_MDIO	20AB 2428h
PRU_ICSS_1_MII_MDIO	20AF 2428h

Figure 6-292. PRUSS_MII_MDIO_USERINTMASKSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						USERINTMASKEDSET	
R-0h						R/W1S-0h	

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 6-755. PRUSS_MII_MDIO_USERINTMASKSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	USERINTMASKEDSET	R/W1S	0h	MDIO user interrupt mask set for userintmasked[1:0], respectively. Writing a bit to 1 will enable MDIO user command complete interrupts for that particular MDIOUserAccess register. MDIO user interrupt for a particular MDIOUserAccess register is disabled if the corresponding bit is 0 . Writing a 0 to this register has no effect.

Table 6-756. Register Call Summary for PRUSS_MII_MDIO_USERINTMASKSET

PRU-ICSS MII MDIO Module

- PRU-ICSS MII MDIO Interractions: [0]
- PRU-ICSS MII MDIO Interrupts: [0][1]
- PRU-ICSS MII MDIO Receive/Transmit Frame Host Software Interface: [0][1][2][3]
- PRU-ICSS MII MDIO Registers: [0]
- PRUSS_MII_MDIO_USERINTMASKSET Register (Offset = 28h) [reset = 0h]: [0]

6.4.10.4.10 PRUSS_MII_MDIO_USERINTMASKCLR Register (Offset = 2Ch) [reset = 0h]

PRUSS_MII_MDIO_USERINTMASKCLR is shown in Figure 6-293 and described in Table 6-758.

USER INTERRUPT MASK CLEAR REGISTER

Table 6-757. PRUSS_MII_MDIO_USERINTMASKCLR Instances

Instance	Physical Address
PRU_ICSS_0_MII_MDIO	20AB 242Ch
PRU_ICSS_1_MII_MDIO	20AF 242Ch

Figure 6-293. PRUSS_MII_MDIO_USERINTMASKCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						USERINTMASKEDCLR	
R-0h						RWr1Clr-0h	

LEGEND: R = Read Only; RWr1Clr = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 6-758. PRUSS_MII_MDIO_USERINTMASKCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	USERINTMASKEDCLR	RWr1Clr	0h	MDIO user command complete interrupt mask clear for userintmasked[1:0], respectively. Writing a bit to 1 will disable further user command complete interrupts for that particular MDIOUserAccess register. Writing a 0 to this register has no effect.

Table 6-759. Register Call Summary for PRUSS_MII_MDIO_USERINTMASKCLR

PRU-ICSS MII MDIO Module

- PRU-ICSS MII MDIO Interractions: [0]
- PRU-ICSS MII MDIO Receive/Transmit Frame Host Software Interface: [0]
- PRU-ICSS MII MDIO Registers: [0]
- PRUSS_MII_MDIO_USERINTMASKCLR Register (Offset = 2Ch) [reset = 0h]: [0]

6.4.10.4.11 PRUSS_MII_MDIO_USERACCESS0 Register (Offset = 80h) [reset = 0h]

PRUSS_MII_MDIO_USERACCESS0 is shown in Figure 6-294 and described in Table 6-761.

USER ACCESS REGISTER0

Table 6-760. PRUSS_MII_MDIO_USERACCESS0 Instances

Instance	Physical Address
PRU_ICSS_0_MII_MDIO	20AB 2480h
PRU_ICSS_1_MII_MDIO	20AF 2480h

Figure 6-294. PRUSS_MII_MDIO_USERACCESS0 Register

31	30	29	28	27	26	25	24
GO	WRITE	ACK	RESERVED			REGADR	
R/W1S-0h	R/W-0h	R/W-0h	R-0h			R/W-0h	
23	22	21	20	19	18	17	16
REGADR			PHYADR				
R/W-0h			R/W-0h				
15	14	13	12	11	10	9	8
DATA							
R/W-0h							
7	6	5	4	3	2	1	0
DATA							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; R/W1S = Read/Write 1 to Set Bit; -*n* = value after reset

Table 6-761. PRUSS_MII_MDIO_USERACCESS0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GO	R/W1S	0h	Go. Writing a 1 to this bit causes the MDIO state machine to perform an MDIO access when it is convenient for it to do so, this is not an instantaneous process. Writing a 0 to this bit has no effect. This bit is write able only if the MDIO state machine is enabled. This bit will self clear when the requested access has been completed. Any writes to the MDIOUserAccess0 register are blocked when the go bit is 1 . If byte access is being used, the go bit should be written last.
30	WRITE	R/W	0h	Write enable. Setting this bit to a 1 causes the MDIO transaction to be a register write, otherwise it is a register read.
29	ACK	R/W	0h	Acknowledge. This bit is set if the PHY acknowledged the read transaction.
28-26	RESERVED	R	0h	Reserved
25-21	REGADR	R/W	0h	Register address. This field specifies the PHY register to be accessed for this transaction.
20-16	PHYADR	R/W	0h	PHY address. This field specifies the PHY to be accessed for this transaction.
15-0	DATA	R/W	0h	User data. The data value read from or to be written to the specified PHY register.

Table 6-762. Register Call Summary for PRUSS_MII_MDIO_USERACCESS0

PRU-ICSS MII MDIO Module

- PRU-ICSS MII MDIO Interractions: [0][1][2][3][4]
- PRU-ICSS MII MDIO Interrupts: [0][1]
- PRU-ICSS MII MDIO Receive/Transmit Frame Host Software Interface: [0][1][2][3][4][5][6][7]
- PRU-ICSS MII MDIO Registers: [0]
- PRUSS_MII_MDIO_USERACCESS0 Register (Offset = 80h) [reset = 0h]: [0]

6.4.10.4.12 PRUSS_MII_MDIO_USERPHYSEL0 Register (Offset = 84h) [reset = 0h]

PRUSS_MII_MDIO_USERPHYSEL0 is shown in Figure 6-295 and described in Table 6-764.

USER PHY SELECT REGISTER0

Table 6-763. PRUSS_MII_MDIO_USERPHYSEL0 Instances

Instance	Physical Address
PRU_ICSS_0_MII_MDIO	20AB 2484h
PRU_ICSS_1_MII_MDIO	20AF 2484h

Figure 6-295. PRUSS_MII_MDIO_USERPHYSEL0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
LINKSEL	LINKINT_ENABLE	RESERVED	PHYADR_MON				
R/W-0h	R/W-0h	R-0h	R/W-0h				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-764. PRUSS_MII_MDIO_USERPHYSEL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	LINKSEL	R/W	0h	Link status determination select. Set to 1 to determine link status using the MLINK pin. Default value is 0 which implies that the link status is determined by the MDIO state machine.
6	LINKINT_ENABLE	R/W	0h	Link change interrupt enable. Set to 1 to enable link change status interrupts for PHY address specified in phyadr_mon. Link change interrupts are disabled if this bit is set to 0 .
5	RESERVED	R	0h	Reserved
4-0	PHYADR_MON	R/W	0h	PHY address whose link status is to be monitored.

Table 6-765. Register Call Summary for PRUSS_MII_MDIO_USERPHYSEL0

PRU-ICSS MII MDIO Module

- PRU-ICSS MII MDIO Interractions: [0][1]
- PRU-ICSS MII MDIO Interrupts: [0][1]
- PRU-ICSS MII MDIO Receive/Transmit Frame Host Software Interface: [0][1][2][3]
- PRU-ICSS MII MDIO Registers: [0]
- PRUSS_MII_MDIO_USERPHYSEL0 Register (Offset = 84h) [reset = 0h]: [0]

6.4.10.4.13 PRUSS_MII_MDIO_USERACCESS1 Register (Offset = 88h) [reset = 0h]

PRUSS_MII_MDIO_USERACCESS1 is shown in Figure 6-296 and described in Table 6-767.

USER ACCESS REGISTER1

Table 6-766. PRUSS_MII_MDIO_USERACCESS1 Instances

Instance	Physical Address
PRU_ICSS_0_MII_MDIO	20AB 2488h
PRU_ICSS_1_MII_MDIO	20AF 2488h

Figure 6-296. PRUSS_MII_MDIO_USERACCESS1 Register

31	30	29	28	27	26	25	24
GO	WRITE	ACK	RESERVED			REGADR	
R/W-0h	R/W-0h	R/W-0h	R-0h			R/W-0h	
23	22	21	20	19	18	17	16
REGADR			PHYADR				
R/W-0h			R/W-0h				
15	14	13	12	11	10	9	8
DATA							
R/W-0h							
7	6	5	4	3	2	1	0
DATA							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-767. PRUSS_MII_MDIO_USERACCESS1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GO	R/W	0h	Go. Writing a 1 to this bit causes the MDIO state machine to perform an MDIO access when it is convenient for it to do so, this is not an instantaneous process. Writing a 0 to this bit has no effect. This bit is write able only if the MDIO state machine is enabled. This bit will self clear when the requested access has been completed. Any writes to the MDIOUserAccess0 register are blocked when the go bit is 1 . If byte access is being used, the go bit should be written last.
30	WRITE	R/W	0h	Write enable. Setting this bit to a 1 causes the MDIO transaction to be a register write, otherwise it is a register read.
29	ACK	R/W	0h	Acknowledge. This bit is set if the PHY acknowledged the read transaction.
28-26	RESERVED	R	0h	Reserved
25-21	REGADR	R/W	0h	Register address. This field specifies the PHY register to be accessed for this transaction.
20-16	PHYADR	R/W	0h	PHY address. This field specifies the PHY to be accessed for this transaction.
15-0	DATA	R/W	0h	User data. The data value read from or to be written to the specified PHY register.

Table 6-768. Register Call Summary for PRUSS_MII_MDIO_USERACCESS1

PRU-ICSS MII MDIO Module

- PRUSS_MII_MDIO_USERACCESS1 Register (Offset = 88h) [reset = 0h]: [0]
- PRU-ICSS MII MDIO Interrupts: [0]
- PRU-ICSS MII MDIO Receive/Transmit Frame Host Software Interface: [0]
- PRU-ICSS MII MDIO Registers: [0]

6.4.10.4.14 PRUSS_MII_MDIO_USERPHYSEL1 Register (Offset = 8Ch) [reset = 0h]

PRUSS_MII_MDIO_USERPHYSEL1 is shown in Figure 6-297 and described in Table 6-770.

USER PHY SELECT REGISTER1

Table 6-769. PRUSS_MII_MDIO_USERPHYSEL1 Instances

Instance	Physical Address
PRU_ICSS_0_MII_MDIO	20AB 248Ch
PRU_ICSS_1_MII_MDIO	20AF 248Ch

Figure 6-297. PRUSS_MII_MDIO_USERPHYSEL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
LINKSEL	LINKINT_ENABLE	RESERVED	PHYADR_MON				
R/W-0h	R/W-0h	R-0h	R/W-0h				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-770. PRUSS_MII_MDIO_USERPHYSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	LINKSEL	R/W	0h	Link status determination select. Set to 1 to determine link status using the MLINK pin. Default value is 0 which implies that the link status is determined by the MDIO state machine.
6	LINKINT_ENABLE	R/W	0h	Link change interrupt enable. Set to 1 to enable link change status interrupts for PHY address specified in phyadr_mon. Link change interrupts are disabled if this bit is set to 0 .
5	RESERVED	R	0h	Reserved
4-0	PHYADR_MON	R/W	0h	PHY address whose link status is to be monitored.

Table 6-771. Register Call Summary for PRUSS_MII_MDIO_USERPHYSEL1

PRU-ICSS MII MDIO Module

- PRU-ICSS MII MDIO Interrupts: [0]
- PRU-ICSS MII MDIO Receive/Transmit Frame Host Software Interface: [0]
- PRU-ICSS MII MDIO Registers: [0]
- PRUSS_MII_MDIO_USERPHYSEL1 Register (Offset = 8Ch) [reset = 0h]: [0]

6.4.11 PRU-ICSS Industrial Ethernet Peripheral (IEP)

This section describes the Industrial Ethernet Peripheral (IEP) module which is part of the PRU-ICSS.

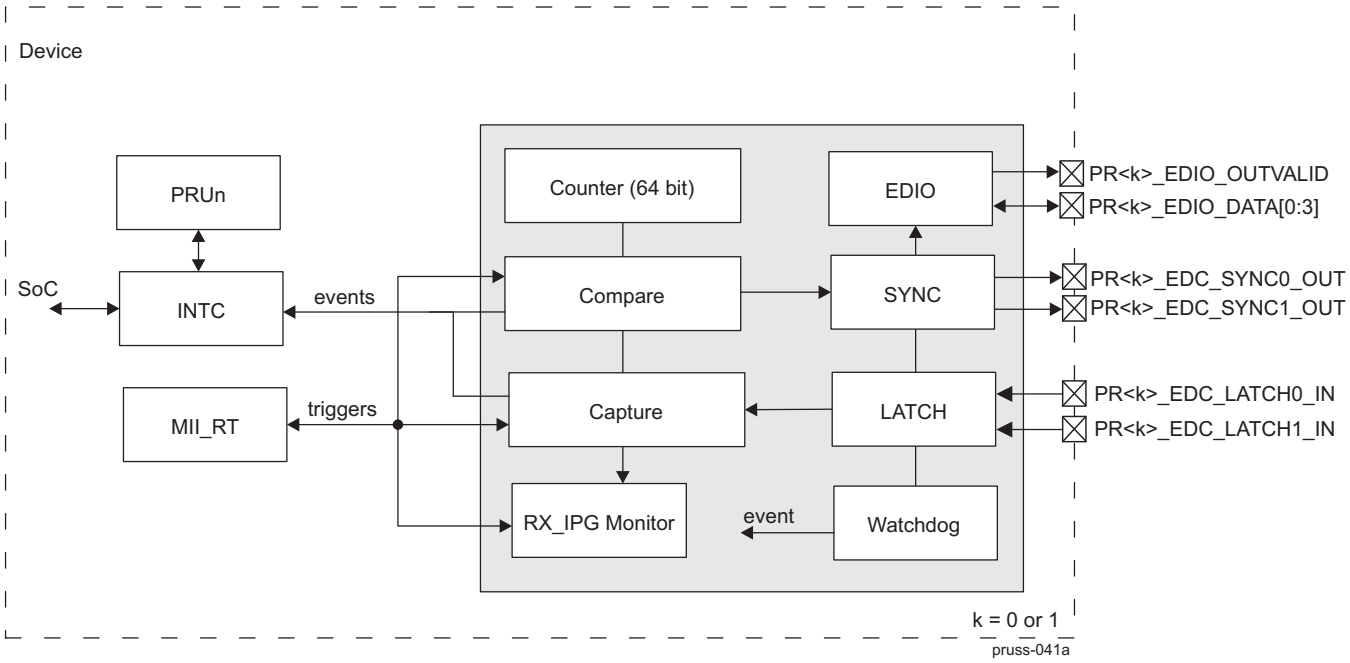
6.4.11.1 PRU-ICSS IEP Overview

The Industrial Ethernet Peripheral (IEP) performs hardware work required for Industrial Ethernet functions. The IEP module features an Industrial Ethernet Timer with 16 compare events, Industrial Ethernet Sync generator and Latch capture, Industrial Ethernet Watchdog Timer, and a digital I/O port (DIGIO).

6.4.11.2 PRU-ICSS IEP Functional Description

This section provides the functional description of the IEP components. The IEP functional block diagram is shown in Figure 6-298.

Figure 6-298. IEP Functional Block Diagram



6.4.11.2.1 PRU-ICSS IEP Clock Generation

The IEP has a selectable module input clock (ICSS_IEP_CLK, see also Section 6.4.3). The clock source is selected by the state of the PRUSS_IEPCLK[0] OCP_EN bit within the PRU-ICSS CFG register space. Two clock sources are supported for the IEP input clock:

- ICSS_IEP_CLK: The default functional clock for IEP is derived as divided version of the IEP/ NSS PLL output clock (IEP_NSS_PLL_CLK/ 5). The IEP functional clock (ICSS_IEP_CLK) runs at 200 MHz.
- ICSS_VCLK_CLK: The PRUSS_CFG interface clock (ICSS_VCLK_CLK) is derived as divided version of the device PLL Controller output clock (CHIP_CLK1/3).

Switching from ICSS_IEP_CLK to ICSS_VCLK_CLK is done by writing 1h to the PRUSS_IEPCLK[0] OCP_EN bit. This is a one time configuration step before enabling the IEP function. Switching back from ICSS_VCLK_CLK to ICSS_IEP_CLK is only supported through a hardware reset of the PRU-ICSS.

CAUTION

When software enables the clock (at PRU-ICSS level) to the IEP module clock input via setting bit PRUSS_IEPCLK[0] OCP_EN to 0b1 in the PRUSS_CFG space, there must be NO in-flight transactions to the IEP block.

CAUTION

ONLY switching from ICSS_IEP_CLK (the IEP specific functional clock source) to the ICSS_VCLK_CLK (top level interface clock) source is supported in software by device integrated PRU-ICSS. Switching back from ICSS_VCLK_CLK to ICSS_IEP_CLK is ONLY supported via assertion of a hardware reset to the PRU-ICSS.

6.4.11.2.2 PRU-ICSS Industrial Ethernet Timer

The industrial ethernet timer is a simple 64-bit timer. This timer is intended for use by industrial ethernet functions but can also be leveraged as a generic timer in other applications.

6.4.11.2.2.1 PRU-ICSS Industrial Ethernet Timer Features

The industrial ethernet timer supports the following features:

- One master 64-bit count-up counter with an overflow status bit.
  - Runs on ICSS_IEP_CLK or ICSS_VCLK_CLK clock.
  - Write 1 to clear status.
  - Supports a programmable increment value from 1 to 16 (default 5).
  - An optional compensation method allows the increment value to apply compensation increment value from 1 to 16 count up to 2^{24} ICSS_IEP_CLK/ ICSS_VCLK_CLK events with additional slow compensation mode.
- 10× 64-bit capture registers:
  - 8 capture inputs, with optional synchronous or asynchronous mode:
    - 6× rise capture, PRUSS_IEP_CAPTURE_RISE0i/PRUSS_IEP_CAPTURE_RISE1i (where i=0 to 5)
    - 2× rise capture- PRUSS_IEP_CAPTURE_RISE06/PRUSS_IEP_CAPTURE_RISE16 and PRUSS_IEP_CAPTURE_RISE07/PRUSS_IEP_CAPTURE_RISE17, each combined with a fall capture- PRUSS_IEP_CAPTURE_FALL06/PRUSS_IEP_CAPTURE_FALL16 and PRUSS_IEP_CAPTURE_FALL07/PRUSS_IEP_CAPTURE_FALL17, respectively
    - One global event (any capture event) output for interrupt
- 16× 64-bit compare registers: PRUSS_IEP_COMPARE0j/PRUSS_IEP_COMPARE1j (where j=0 to 15) and PRUSS_IEP_COMPARE_STATUS
  - 16 status bits, write 1 to clear
  - 16 individual event outputs
  - One global event output for interrupt generation triggered by any compare event
- 32 outputs, one high-level and one high-pulse for each compare hit event
- PRUSS_IEP_COMPARE_CFG[0] CMP0_RST_CNT_EN, if enabled, will reset the master counter on the next ICSS_IEP_CLK/ ICSS_VCLK_CLK cycle
- eHRPWM0_SYNCO/ eHRPWM3_SYNCO, if enabled, will reset the master counter on the next ICSS_IEP_CLK/ ICSS_VCLK_CLK cycle
- master counter reset-state is programmable

6.4.11.2.2.2 PRU-ICSS Industrial Ethernet Timer Basic Programming Sequence

Follow these basic steps to configure the IEP Timer.

Counter maintains/function:

1. Once enabled, the counter will count every PRUSS_IEP_CLK cycle, default rate of 200 MHz.
2. It is a free running counter with a sticky over flag status bit.
3. The counter over flow flag will get set when the counter switches/rollsover from 0XFFFF_FFFF to 0x0000_0000.
4. The counter will continue to count up. The software will need to read/clear the counter over flow flag and increment the MSB in software variable.

Compare function:

1. Initialize timer to known state (default values)
   - Disable counter (PRUSS_IEP_GLOBAL_CFG[0] CNT_ENABLE = 0)
   - Reset Count Register (PRUSS_IEP_LOW_COUNTER, PRUSS_IEP_HIGH_COUNTER) by writing FFFFFFFFh to clear
   - Clear overflow status register (PRUSS_IEP_STATUS[0] CNT_OVF = 1)
   - Clear compare status (PRUSS_IEP_COMPARE_STATUS) by writing FFFFFFFFh to clear
2. Set compare values PRUSS_IEP_COMPARE0j,PRUSS_IEP_COMPARE1j
3. Enable compare events (PRUSS_IEP_COMPARE_CFG[8:1] CMP_EN).
4. Set increment value (PRUSS_IEP_GLOBAL_CFG[7:4] DEFAULT_INC).
5. Set compensation value (PRUSS_IEP_COMPENSATION[23:0] COMPEN_CNT)
6. Enable counter (PRUSS_IEP_GLOBAL_CFG[0] CNT_ENABLE = 1)

Capture function:

1. Update/Enable the counter if required
2. Program the enable the desired capture event
3. Wait for global capture event
4. Read/Clear the capture status to determine which capture event occurred
5. Read the capture count

6.4.11.2.2.3 Industrial Ethernet Mapping

Some of the capture inputs and compare registers are mapped to specific industrial Ethernet functions in hardware, shown in Table 6-772. All capture inputs are mapped to industrial Ethernet functions, and these inputs are not available for any other application. The cmp1 and cmp2 compare registers also function as the start time triggers for SYNC0 and SYNC1, respectively.

Table 6-772. Industrial Ethernet Timer Mode Mapping

Capture Input	IEP Line/Function
cap0, rise only	PRU0_RX_SOF
cap1, rise only	PRU0_RX_SFD
cap2, rise only	PRU1_RX_SOF
cap3, rise only	PRU1_RX_SFD
cap4, rise only	PORT0_TX_SOF
cap5, rise only	PORT1_TX_SOF
cap6, rise and fall	latch0_in, SOC IO inputs
cap7, rise and fall	latch1_in, SOC IO inputs
cmp1	For SYNC0 trigger of start time
cmp2	For SYNC1 trigger of start time; only valid in the independent mode
cmp3	For MII TX0 start trigger, if MII register PRUSS_MII_RT_TXCFG0/1[2] TX_EN_MODE is enabled.

Table 6-772. Industrial Ethernet Timer Mode Mapping (continued)

Capture Input	IEP Line/Function
cmp4	For MII TX1 start trigger, if MII register PRUSS_MII_RT_TXCFG0/1[2] TX_EN_MODE is enabled.

6.4.11.2.3 PRU-ICSS IEP Sync0/Sync1 Signals Generation

The industrial ethernet sync block supports the generation of two synchronization signals: SYNC0 and SYNC1. SYNC0 and SYNC1 can be directly mapped to output signals (pr<k>_edc_sync0_out and pr<k>_edc_sync1_out) for external devices to use. They can also be used for internal synchronization within the PRU-ICSS. These signals are also mapped as system events and can therefore be mapped to the Arm core's Host interrupts.

6.4.11.2.3.1 PRU-ICSS IEP Sync0/Sync1 Features

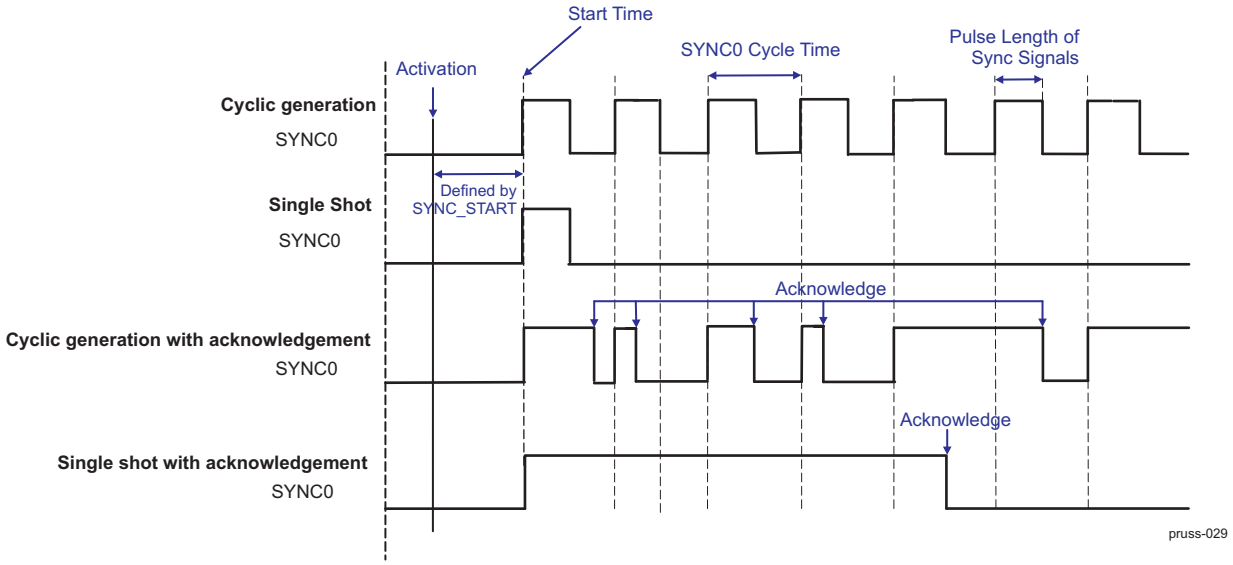
The industrial ethernet sync block supports the following features:

- Two synchronize generation signals (SYNC0, SYNC1)
  - Activation time synchronized with IEP Timer
- CMP[1] triggers SYNC0 activation time
- CMP[2] triggers SYNC1 activation time (only valid in the independent mode)
  - Pulse width defined by registers or ack mode (remain asserted until software acknowledged)
  - Cyclic or single-shot operation
  - Option to enable or disable sync generation
- Programmable number of clock cycles between the start of SYNC0 to the start of SYNC1

6.4.11.2.3.2 PRU-ICSS IEP Sync0/Sync1 Generation Modes

There are four modes of operation for the sync signals: cyclic mode, single shot mode, cyclic with acknowledge mode, and single shot with acknowledge mode. Figure 6-299 shows examples of these modes. The start time is set by the SYNC_START bit field in the PRUSS_IEP_SYNC_START register. The cycle time is configured by the SYNC0_PERIOD bit field in the PRUSS_IEP_SYNC0_PERIOD register. The pulse length is defined by SYNC_HPW bit field in the PRUSS_IEP_SYNC_PWIDTH register.

Figure 6-299. PRU-ICSS IEP SYNC0 Signal Generation Modes

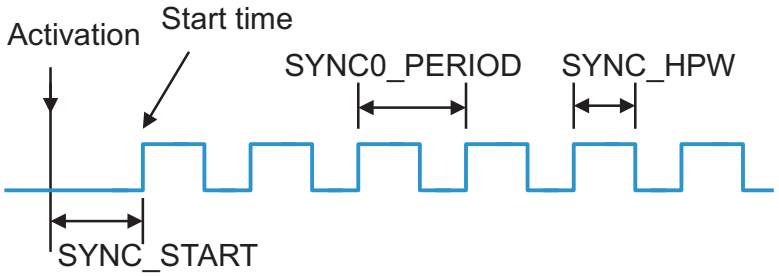


In SYNC1 dependent mode (PRUSS_IEP_SYNC_CTRL[8] SYNC1_IND_EN = 0b0), SYNC1 depends on SYNC0 and the start time of the SYNC1 can be defined by the PRUSS_IEP_SYNC1_DELAY register. Figure 6-300 shows different examples when changing the value in the PRUSS_IEP_SYNC1_DELAY register. Note if the SYNC1 delay time is 0, SYNC1 reflects SYNC0. Cyclic generation cannot be used for network time synchronized applications because only the CMP1/CMP2 hit occurs in the compensated time domain.

Figure 6-300. Examples of the Dependent Mode of SYNC1

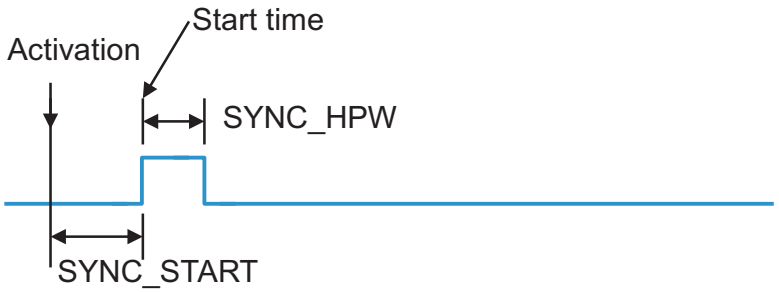
SYNC0 : **Cyclic generation**

Field	Value
SYNC0_CYCLIC_EN	1
SYNC0_ACK_EN	0



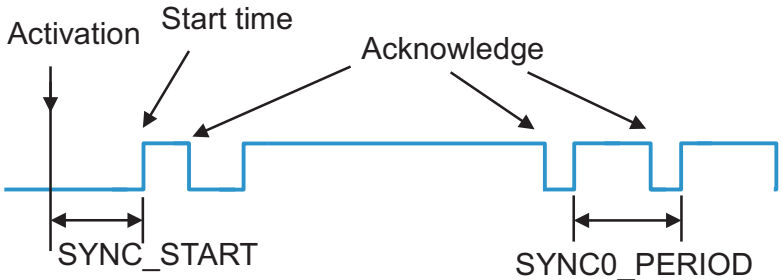
SYNC0 : **Single shot**

Field	Value
SYNC0_CYCLIC_EN	0
SYNC0_ACK_EN	0



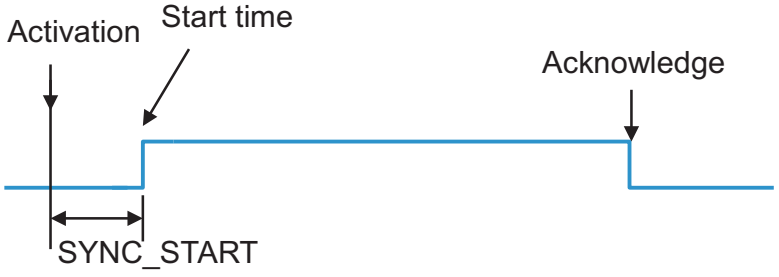
SYNC0 : **Cyclic generation with Acknowledge**

Field	Value
SYNC0_CYCLIC_EN	1
SYNC0_ACK_EN	1



SYNC0 : **Single shot with Acknowledge**

Field	Value
SYNC0_CYCLIC_EN	0
SYNC0_ACK_EN	1



pruss-031

6.4.11.2.4 PRU-ICSS Industrial Ethernet WatchDog

In industrial ethernet applications, the watchdog timer (WD) is used as a safety feature to monitor process data communication and to turn off the outputs of the digital input/output (DIGIO) functional block after a set time. The WD will thereby protect the system from errors or faults by timeout or expiration. The expiration is used to initiate corrective action in order to keep the system in a safe state and restore normal operation based on configuration. Therefore, if the system is stable, the watchdog timer should be regularly reset or cleared to avoid timeout or expiration.

The industrial ethernet watchdog timer supports the following features:

- One 16-bit pre-divider for generating a WD clock (default 100µs) based on ICSS_IEP_CLK input
- Two 16-bit Watchdog Timers:

  - PDI_WD for Sync Managers WD, used in conjunction with digital input/output (DIGIO)

  - PD_WD for data link layer user WD, used in conjunction with data link layer or application layer interface actions

NOTE: For more details on the PRU-ICSS Industrial Ethernet Watchdog timer, refer also to the PRU-ICSS_IEP_WD_x register descriptions covered in the .

6.4.11.2.5 PRU-ICSS Industrial Ethernet Digital IOs

The IEP digital I/O (DIGIO) block provides dedicated I/Os for industrial ethernet protocols. The digital inputs can be sampled when specific events occur or continuously as a raw input. Likewise, driving the digital outputs can be triggered by specific events or controlled by software. The timing, delay cycle clocks, data sources, and data valid of the digital input and outputs are controlled by the PRUSS_IEP_DIGIO_CTRL and PRUSS_IEP_DIGIO_EXP registers. Additionally, the IEP DIGIO block can be used as generic I/Os in other applications.

6.4.11.2.5.1 Features

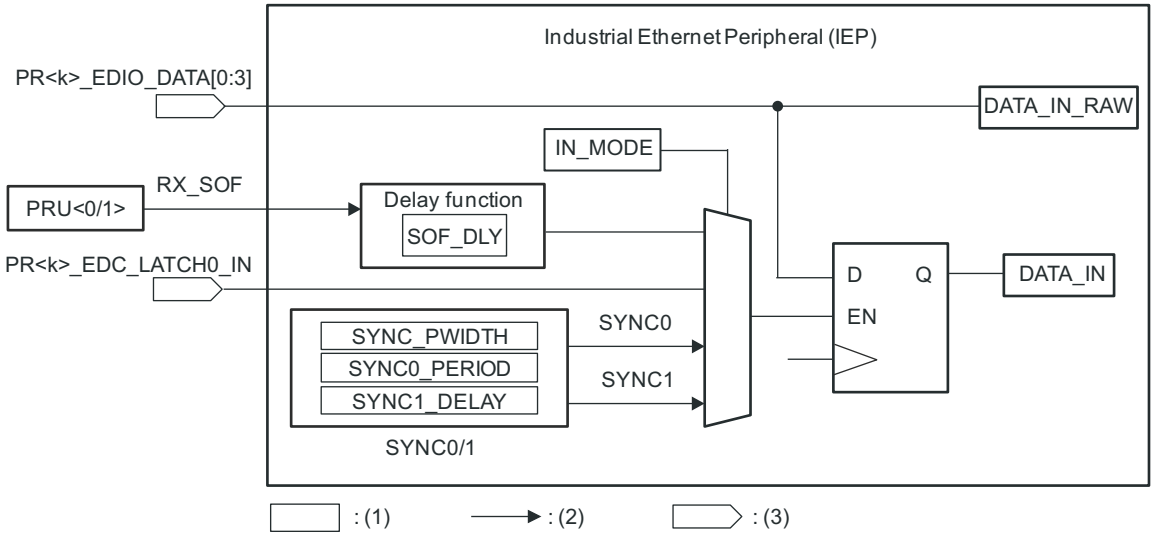
The industrial ethernet digital I/O supports the following features:

- Digital data output
  - 4 channels (PR<k>_EDIO_DATA[0:3])
  - Five event options for driving output data output:
    - End of frame event (PRU0/1_RX_EOF)
    - SYNC0 events
    - SYNC1 events
    - Watchdog trigger
    - Software enable
- Digital data out enable (optional tri-state control)
- Digital data input
  - 4 channels (PR<k>_EDIO_DATA[0:3])
  - PRUSS_IEP_DIGIO_DATA_IN_RAW supports direct sampling of PR<k>_EDIO_DATA[0:3]
  - DIGIO_DATA_IN supports four event options to trigger sampling of pr<k>_edio_data_in:
    - Start of frame event in start of frame (SOF) mode
    - pr<k>_edio_latch_in event
    - SYNC0 events
    - SYNC1 events

6.4.11.2.5.2 DIGIO Block Diagrams

Figure 6-301 shows the signals and registers for capturing the DIGIO data in. Note that IN_MODE in the PRUSS_IEP_DIGIO_CTRL register must be set to 1 for data to be latched on the external PR<k>_EDC_LATCH0_IN signal. In PRU0/1_RX_SOF mode, the delay time of capturing pr1_edio_data_in is programmable through the SOF_DLY bit of the PRUSS_IEP_DIGIO_EXP register.

Figure 6-301. IEP DIGIO Data In



(1) Register

(2) Internal signal wire

(3) External pin input/output

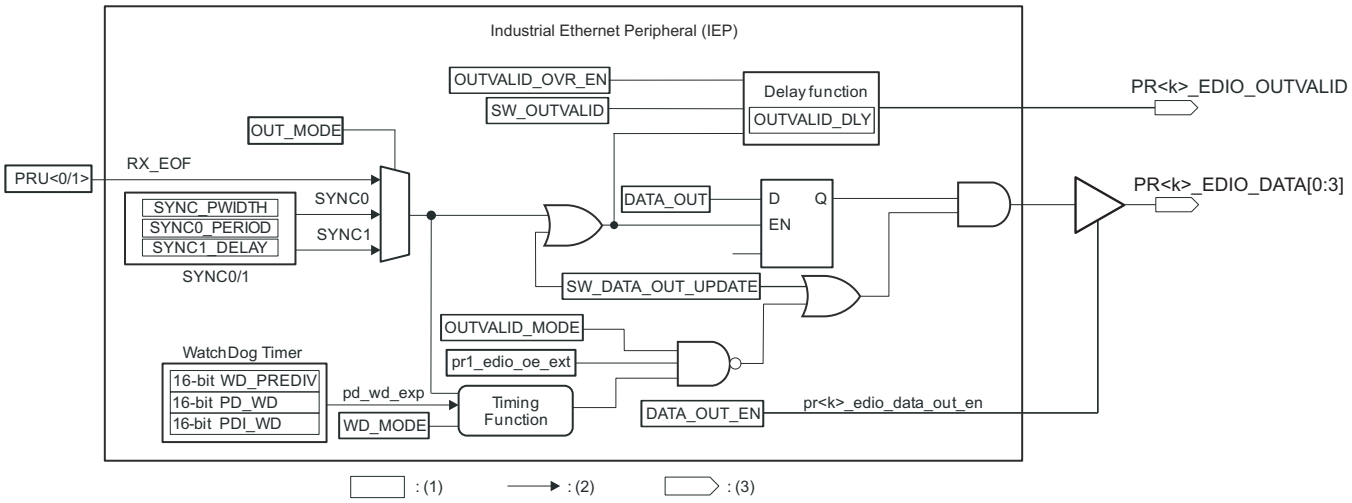
(1) Register

(2) External pin input/output

Figure 6-302 shows the signals and registers for driving the DIGIO data out.

The PR<k>_EDIO_DATA is immediately forced to zero when OUTVALID_MODE = 0b1, pr1_edio_oe_ext = 0b1, and PD_WD_EXP = 1, or the next update hardware post PD_WD_EXP. Delay assertion of PR<k>_EDIO_OUTVALID from PR<k>_EDIO_DATA update events are controlled by software (SW_OUTVALID).

Figure 6-302. IEP DIGIO Data Out



(1) Register
(2) Internal signal wire
(3) External pin input/output

(1) Register
(2) External pin input/output

6.4.11.2.5.3 Basic Programming Model

Follow these steps to configure and read the DIGIO Data Inpfut.

1. Read PRUSS_IEP_DIGIO_DATA_IN_RAW for raw input data

or

1. Enable sampling of PR<k>_EDIO_DATA[0:3] by setting PRUSS_IEP_DIGIO_CTRL[5:4] IN_MODE = 1h.
2. Read PRUSS_IEP_DIGIO_DATA_IN for data sampled upon PR<k>_EDC_LATCH0_IN posedge

Follow these steps to configure and write to the DIGIO Data Output.

1. Pre-configure DIGIO by setting PRUSS_IEP_DIGIO_EXP[1] OUTVALID_OVR_EN and PRUSS_IEP_DIGIO_EXP[0] SW_DATA_OUT_UPDATE
2. Write to PRUSS_IEP_DIGIO_DATA_OUT to configure output data.
3. To HiZ output, set corresponding PRUSS_IEP_DIGIO_DATA_OUT_EN bits to 1 (clear to 0 to drive value stored in PRUSS_IEP_DIGIO_DATA_OUT).

6.4.11.3 PRU-ICSS_IEP Registers

Table 6-774 lists the memory-mapped registers for the PRU-ICSS_IEP module. All register offset addresses not listed in Table 6-774 should be considered as reserved locations and the register contents should not be modified.

Table 6-773. PRU-ICSS_IEP Instances

Instance	Base Address
PRU_ICSS_0_IEP	20AA E000h
PRU_ICSS_1_IEP	20AE E000h

Table 6-774. PRU-ICSS_IEP Registers

Offset	Acronym	Register Name	PRU_ICSS_0_ IEP Physical Address	PRU_ICSS_1_ IEP Physical Address	Section
0h	PRUSS_IEP_GLOBAL_CFG	GLOBAL CFG	20AA E000h	20AE E000h	Section 6.4.11.3.1
4h	PRUSS_IEP_STATUS	STATUS	20AA E004h	20AE E004h	Section 6.4.11.3.2
8h	PRUSS_IEP_COMPENSATION	COMPENSATION	20AA E008h	20AE E008h	Section 6.4.11.3.3
Ch	PRUSS_IEP_SLOW_COMPENSATION	SLOW COMPENSATION	20AA E00Ch	20AE E00Ch	Section 6.4.11.3.4
10h	PRUSS_IEP_LOW_COUNTER	64 bit count value low	20AA E010h	20AE E010h	Section 6.4.11.3.5
14h	PRUSS_IEP_HIGH_COUNTER	64 bit count value high	20AA E014h	20AE E014h	Section 6.4.11.3.6
18h	PRUSS_IEP_CAPTURE_CFG	CAPTURE CFG	20AA E018h	20AE E018h	Section 6.4.11.3.7
1Ch	PRUSS_IEP_CAPTURE_STATUS	CAPTURE STATUS	20AA E01Ch	20AE E01Ch	Section 6.4.11.3.8
20h	PRUSS_IEP_CAPTURE_RISE00	CAPTURE RISE0 low	20AA E020h	20AE E020h	Section 6.4.11.3.9
24h	PRUSS_IEP_CAPTURE_RISE10	CAPTURE RISE0 high	20AA E024h	20AE E024h	Section 6.4.11.3.10
28h	PRUSS_IEP_CAPTURE_RISE01	CAPTURE RISE1 low	20AA E028h	20AE E028h	Section 6.4.11.3.11
2Ch	PRUSS_IEP_CAPTURE_RISE11	CAPTURE RISE1 high	20AA E02Ch	20AE E02Ch	Section 6.4.11.3.12
30h	PRUSS_IEP_CAPTURE_RISE02	CAPTURE RISE2 low	20AA E030h	20AE E030h	Section 6.4.11.3.13
34h	PRUSS_IEP_CAPTURE_RISE12	CAPTURE RISE2 high	20AA E034h	20AE E034h	Section 6.4.11.3.14
38h	PRUSS_IEP_CAPTURE_RISE03	CAPTURE RISE3 low	20AA E038h	20AE E038h	Section 6.4.11.3.15
3Ch	PRUSS_IEP_CAPTURE_RISE13	CAPTURE RISE3 high	20AA E03Ch	20AE E03Ch	Section 6.4.11.3.16
40h	PRUSS_IEP_CAPTURE_RISE04	CAPTURE RISE4 low	20AA E040h	20AE E040h	Section 6.4.11.3.17
44h	PRUSS_IEP_CAPTURE_RISE14	CAPTURE RISE4 high	20AA E044h	20AE E044h	Section 6.4.11.3.18
48h	PRUSS_IEP_CAPTURE_RISE05	CAPTURE RISE5 low	20AA E048h	20AE E048h	Section 6.4.11.3.19
4Ch	PRUSS_IEP_CAPTURE_RISE15	CAPTURE RISE5 high	20AA E04Ch	20AE E04Ch	Section 6.4.11.3.20
50h	PRUSS_IEP_CAPTURE_RISE06	CAPTURE RISE6 low	20AA E050h	20AE E050h	Section 6.4.11.3.21

Table 6-774. PRU-ICSS_IEP Registers (continued)

Offset	Acronym	Register Name	PRU_ICSS_0_ IEP Physical Address	PRU_ICSS_1_ IEP Physical Address	Section
54h	PRUSS_IEP_CAPTURE_RISE16	CAPTURE RISE6 high	20AA E054h	20AE E054h	Section 6.4.11.3.22
58h	PRUSS_IEP_CAPTURE_FALL06	CAPTURE FALL6 low	20AA E058h	20AE E058h	Section 6.4.11.3.23
5Ch	PRUSS_IEP_CAPTURE_FALL16	CAPTURE FALL6 high	20AA E05Ch	20AE E05Ch	Section 6.4.11.3.24
60h	PRUSS_IEP_CAPTURE_RISE07	CAPTURE RISE7 low	20AA E060h	20AE E060h	Section 6.4.11.3.25
64h	PRUSS_IEP_CAPTURE_RISE17	CAPTURE RISE7 high	20AA E064h	20AE E064h	Section 6.4.11.3.26
68h	PRUSS_IEP_CAPTURE_FALL07	CAPTURE FALL7 low	20AA E068h	20AE E068h	Section 6.4.11.3.27
6Ch	PRUSS_IEP_CAPTURE_FALL17	CAPTURE FALL7 high	20AA E06Ch	20AE E06Ch	Section 6.4.11.3.28
70h	PRUSS_IEP_COMPARE_CFG	COMPARE CFG	20AA E070h	20AE E070h	Section 6.4.11.3.29
74h	PRUSS_IEP_COMPARE_STATUS	COMPARE STATUS	20AA E074h	20AE E074h	Section 6.4.11.3.30
78h	PRUSS_IEP_COMPARE00	COMPARE0 low	20AA E078h	20AE E078h	Section 6.4.11.3.31
7Ch	PRUSS_IEP_COMPARE10	COMPARE0 high	20AA E07Ch	20AE E07Ch	Section 6.4.11.3.32
80h	PRUSS_IEP_COMPARE01	COMPARE1 low	20AA E080h	20AE E080h	Section 6.4.11.3.33
84h	PRUSS_IEP_COMPARE11	COMPARE1 high	20AA E084h	20AE E084h	Section 6.4.11.3.34
88h	PRUSS_IEP_COMPARE02	COMPARE2 low	20AA E088h	20AE E088h	Section 6.4.11.3.35
8Ch	PRUSS_IEP_COMPARE12	COMPARE2 high	20AA E08Ch	20AE E08Ch	Section 6.4.11.3.36
90h	PRUSS_IEP_COMPARE03	COMPARE3 low	20AA E090h	20AE E090h	Section 6.4.11.3.37
94h	PRUSS_IEP_COMPARE13	COMPARE3 high	20AA E094h	20AE E094h	Section 6.4.11.3.38
98h	PRUSS_IEP_COMPARE04	COMPARE4 low	20AA E098h	20AE E098h	Section 6.4.11.3.39
9Ch	PRUSS_IEP_COMPARE14	COMPARE4 high	20AA E09Ch	20AE E09Ch	Section 6.4.11.3.40
A0h	PRUSS_IEP_COMPARE05	COMPARE5 low	20AA E0A0h	20AE E0A0h	Section 6.4.11.3.41
A4h	PRUSS_IEP_COMPARE15	COMPARE5 high	20AA E0A4h	20AE E0A4h	Section 6.4.11.3.42
A8h	PRUSS_IEP_COMPARE06	COMPARE6 low	20AA E0A8h	20AE E0A8h	Section 6.4.11.3.43
ACh	PRUSS_IEP_COMPARE16	COMPARE6 high	20AA E0ACh	20AE E0ACh	Section 6.4.11.3.44
B0h	PRUSS_IEP_COMPARE07	COMPARE7 low	20AA E0B0h	20AE E0B0h	Section 6.4.11.3.45
B4h	PRUSS_IEP_COMPARE17	COMPARE7 high	20AA E0B4h	20AE E0B4h	Section 6.4.11.3.46
B8h	PRUSS_IEP_RXIPG0	Status for the RX port which is attached to PRU0	20AA E0B8h	20AE E0B8h	Section 6.4.11.3.47
BCh	PRUSS_IEP_RXIPG1	Status for the RX port which is attached to PRU1	20AA E0BCh	20AE E0BCh	Section 6.4.11.3.48

Table 6-774. PRU-ICSS_IEP Registers (continued)

Offset	Acronym	Register Name	PRU_ICSS_0_ IEP Physical Address	PRU_ICSS_1_ IEP Physical Address	Section
C0h	PRUSS_IEP_COMPARE08	COMPARE8 low	20AA E0C0h	20AE E0C0h	Section 6.4.11.3.49
C4h	PRUSS_IEP_COMPARE18	COMPARE8 high	20AA E0C4h	20AE E0C4h	Section 6.4.11.3.50
C8h	PRUSS_IEP_COMPARE09	COMPARE9 low	20AA E0C8h	20AE E0C8h	Section 6.4.11.3.51
CCh	PRUSS_IEP_COMPARE19	COMPARE9 high	20AA E0CCh	20AE E0CCh	Section 6.4.11.3.52
D0h	PRUSS_IEP_COMPARE010	COMPARE10 low	20AA E0D0h	20AE E0D0h	Section 6.4.11.3.53
D4h	PRUSS_IEP_COMPARE110	COMPARE10 high	20AA E0D4h	20AE E0D4h	Section 6.4.11.3.54
D8h	PRUSS_IEP_COMPARE011	COMPARE11 low	20AA E0D8h	20AE E0D8h	Section 6.4.11.3.55
DCh	PRUSS_IEP_COMPARE111	COMPARE11 high	20AA E0DCh	20AE E0DCh	Section 6.4.11.3.56
E0h	PRUSS_IEP_COMPARE012	COMPARE12 low	20AA E0E0h	20AE E0E0h	Section 6.4.11.3.57
E4h	PRUSS_IEP_COMPARE112	COMPARE12 high	20AA E0E4h	20AE E0E4h	Section 6.4.11.3.58
E8h	PRUSS_IEP_COMPARE013	COMPARE13 low	20AA E0E8h	20AE E0E8h	Section 6.4.11.3.59
ECh	PRUSS_IEP_COMPARE113	COMPARE13 high	20AA E0ECh	20AE E0ECh	Section 6.4.11.3.60
F0h	PRUSS_IEP_COMPARE014	COMPARE14 low	20AA E0F0h	20AE E0F0h	Section 6.4.11.3.61
F4h	PRUSS_IEP_COMPARE114	COMPARE14 high	20AA E0F4h	20AE E0F4h	Section 6.4.11.3.62
F8h	PRUSS_IEP_COMPARE015	COMPARE15 low	20AA E0F8h	20AE E0F8h	Section 6.4.11.3.63
FCh	PRUSS_IEP_COMPARE115	COMPARE15 high	20AA E0FCh	20AE E0FCh	Section 6.4.11.3.64
100h	PRUSS_IEP_LOW_COUNTER_RESET_VALUE	Reset value of the Master Counter (lower 32-bits)	20AA E100h	20AE E100h	Section 6.4.11.3.65
104h	PRUSS_IEP_HIGH_COUNTER_RESET_VALUE	Reset value of the Master Counter (upper 32-bits)	20AA E104h	20AE E104h	Section 6.4.11.3.66
108h	PRUSS_IEP_PWM	PWM Sync Out	20AA E108h	20AE E108h	Section 6.4.11.3.67
180h	PRUSS_IEP_SYNC_CTRL	SYNC GENERATION CONTROL	20AA E180h	20AE E180h	Section 6.4.11.3.68
184h	PRUSS_IEP_SYNC_FIRST_STAT	SYNC GENERATION FIRST EVENT STATUS	20AA E184h	20AE E184h	Section 6.4.11.3.69
188h	PRUSS_IEP_SYNC0_STAT	SYNC0 STATUS	20AA E188h	20AE E188h	Section 6.4.11.3.70
18Ch	PRUSS_IEP_SYNC1_STAT	SYNC1 STATUS	20AA E18Ch	20AE E18Ch	Section 6.4.11.3.71
190h	PRUSS_IEP_SYNC_PWIDTH	SYNC PULSE WIDTH CONFIGURE	20AA E190h	20AE E190h	Section 6.4.11.3.72
194h	PRUSS_IEP_SYNC0_PERIOD	SYNC0 PERIOD CONFIGURE	20AA E194h	20AE E194h	Section 6.4.11.3.73
198h	PRUSS_IEP_SYNC1_DELAY	SYNC1 DELAY	20AA E198h	20AE E198h	Section 6.4.11.3.74
19Ch	PRUSS_IEP_SYNC_START	SYNC START CONFIGURE	20AA E19Ch	20AE E19Ch	Section 6.4.11.3.75

Table 6-774. PRU-ICSS_IEP Registers (continued)

Offset	Acronym	Register Name	PRU_ICSS_0_ IEP Physical Address	PRU_ICSS_1_ IEP Physical Address	Section
200h	PRUSS_IEP_WD_PREDIV	WATCHDOG PRE-DIVIDER	20AA E200h	20AE E200h	Section 6.4.11.3.76
204h	PRUSS_IEP_PDI_WD_TIM	PDI WATCHDOG TIMER CONFIGURE	20AA E204h	20AE E204h	Section 6.4.11.3.77
208h	PRUSS_IEP_PD_WD_TIM	PD WATCHDOG TIMER CONFIGURE	20AA E208h	20AE E208h	Section 6.4.11.3.78
20Ch	PRUSS_IEP_WD_STATUS	WATCHDOG STATUS	20AA E20Ch	20AE E20Ch	Section 6.4.11.3.79
210h	PRUSS_IEP_WD_EXP_CNT	WATCHDOG TIMER EXPIRATION COUNTER	20AA E210h	20AE E210h	Section 6.4.11.3.80
214h	PRUSS_IEP_WD_CTRL	WATCHDOG CONTROL	20AA E214h	20AE E214h	Section 6.4.11.3.81
300h	PRUSS_IEP_DIGIO_CTRL	DIGIO Control	20AA E300h	20AE E300h	Section 6.4.11.3.82
304h	PRUSS_IEP_DIGIO_STATUS	DIGIO Status	20AA E304h	20AE E304h	Section 6.4.11.3.83
308h	PRUSS_IEP_DIGIO_DATA_IN	DIGIO Data Input	20AA E308h	20AE E308h	Section 6.4.11.3.84
30Ch	PRUSS_IEP_DIGIO_DATA_IN_RAW	DIGIO Data Input. Direct Sample.	20AA E30Ch	20AE E30Ch	Section 6.4.11.3.85
310h	PRUSS_IEP_DIGIO_DATA_OUT	DIGIO Data Output	20AA E310h	20AE E310h	Section 6.4.11.3.86
314h	PRUSS_IEP_DIGIO_DATA_OUT_EN	DIGIO Data Input which controls tri-state of pr<k>_edio_data_out_en[3:0]	20AA E314h	20AE E314h	Section 6.4.11.3.87
318h	PRUSS_IEP_DIGIO_EXP	DIGIO, Defines which RX_EOF is used	20AA E318h	20AE E318h	Section 6.4.11.3.88

6.4.11.3.1 PRUSS_IEP_GLOBAL_CFG Register (Offset = 0h) [reset = 550h]

PRUSS_IEP_GLOBAL_CFG is shown in Figure 6-303 and described in Table 6-776.

GLOBAL CFG

Table 6-775. PRUSS_IEP_GLOBAL_CFG Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E000h
PRU_ICSS_1_IEP	20AE E000h

Figure 6-303. PRUSS_IEP_GLOBAL_CFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				CMP_INC			
R-0h				R/W-5h			
15	14	13	12	11	10	9	8
CMP_INC							
R/W-5h							
7	6	5	4	3	2	1	0
DEFAULT_INC				RESERVED			CNT_ENABLE
R/W-5h				R-0h			R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-776. PRUSS_IEP_GLOBAL_CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-8	CMP_INC	R/W	5h	Defines the increment value when compensation is active
7-4	DEFAULT_INC	R/W	5h	Defines the default increment value
3-1	RESERVED	R	0h	Reserved
0	CNT_ENABLE	R/W	0h	Counter enable. 0h: Disables the counter. The counter maintains the current count. 1h: Enables the counter.

Table 6-777. Register Call Summary for PRUSS_IEP_GLOBAL_CFG

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRUSS_IEP_GLOBAL_CFG Register (Offset = 0h) [reset = 550h]: [0]
- PRU-ICSS Industrial Ethernet Timer Basic Programming Sequence: [0][1][2]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.2 PRUSS_IEP_STATUS Register (Offset = 4h) [reset = 0h]

PRUSS_IEP_STATUS is shown in Figure 6-304 and described in Table 6-779.

STATUS

Table 6-778. PRUSS_IEP_STATUS Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E004h
PRU_ICSS_1_IEP	20AE E004h

Figure 6-304. PRUSS_IEP_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CNT_OVF
R-0h							RWr1Clr-0h

LEGEND: R = Read Only; RWr1Clr = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 6-779. PRUSS_IEP_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	CNT_OVF	RWr1Clr	0h	Counter overflow status. 0h: No overflow 1h: Overflow occurred

Table 6-780. Register Call Summary for PRUSS_IEP_STATUS

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRU-ICSS Industrial Ethernet Timer Basic Programming Sequence: [0]
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_STATUS Register (Offset = 4h) [reset = 0h]: [0]

6.4.11.3.3 PRUSS_IEP_COMPENSATION Register (Offset = 8h) [reset = 0h]

PRUSS_IEP_COMPENSATION is shown in Figure 6-305 and described in Table 6-782.

COMPENSATION

Table 6-781. PRUSS_IEP_COMPENSATION Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E008h
PRU_ICSS_1_IEP	20AE E008h

Figure 6-305. PRUSS_IEP_COMPENSATION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								COMPEN_CNT																							
R-0h								R/W-0h																							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-782. PRUSS_IEP_COMPENSATION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-0	COMPEN_CNT	R/W	0h	Compensation counter. Read returns the current COMPEN_CNT value. 0h: Compensation is disabled and counter will increment by DEFAULT_INC. Nh: Compensation is enabled until COMPEN_CNT decrements to 0. The COMPEN_CNT value decrements on every IEP_CLK cycle. When COMPEN_CNT is greater than 0, then count value increments by CMP_INC. NOTE: SLOW_COMPEN_CNT MUST be set to zero IF COMPEN_CNT is not zero.

Table 6-783. Register Call Summary for PRUSS_IEP_COMPENSATION

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPENSATION Register (Offset = 8h) [reset = 0h]: [0]
- PRU-ICSS Industrial Ethernet Timer Basic Programming Sequence: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.4 PRUSS_IEP_SLOW_COMPENSATION Register (Offset = Ch) [reset = 0h]

PRUSS_IEP_SLOW_COMPENSATION is shown in Figure 6-306 and described in Table 6-785.

SLOW COMPENSATION

Table 6-784. PRUSS_IEP_SLOW_COMPENSATION Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E00Ch
PRU_ICSS_1_IEP	20AE E00Ch

Figure 6-306. PRUSS_IEP_SLOW_COMPENSATION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLOW_COMPEN_CNT																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-785. PRUSS_IEP_SLOW_COMPENSATION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SLOW_COMPEN_CNT	R/W	0h	Slow compensation counter. Write 0h: Slow compensation is disabled and counter will increment by DEFAULT_INC. Write Nh: Compensation is enabled for 1 count for every SLOW_COMPEN_CNT cycle, this is free running and continuous until software clears the MMR. For example, SLOW_COMPEN_CNT = 16, every 16 clock cycles the compensation value is used for 1 count. Note: COMPEN_CNT MUST be set to zero IF SLOW_COMPEN_CNT is not zero. Software can read the number of cycles left until the compensation event. For example, software writes SLOW_COMPEN_CNT = 100h and reads SLOW_COMPEN_CNT = 7h. This means in 6 more IEP_CLK cycles before the counter reaches 1h for the compensation event. If software writes SLOW_COMPEN_CNT = 8000h before compensation event, then the counter will reset to 8000h.

Table 6-786. Register Call Summary for PRUSS_IEP_SLOW_COMPENSATION

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_SLOW_COMPENSATION Register (Offset = Ch) [reset = 0h]: [0]

6.4.11.3.5 PRUSS_IEP_LOW_COUNTER Register (Offset = 10h) [reset = 0h]

PRUSS_IEP_LOW_COUNTER is shown in Figure 6-307 and described in Table 6-788.

64 bit count value low

Table 6-787. PRUSS_IEP_LOW_COUNTER Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E010h
PRU_ICSS_1_IEP	20AE E010h

Figure 6-307. PRUSS_IEP_LOW_COUNTER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-788. PRUSS_IEP_LOW_COUNTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R/W	0h	64 bit count value (lower 32-bits). Increments by (DEFAULT_INC or CMP_INC) on every positive edge of ICSS_IEP_CLK (200MHz) or ICSS_VCLK_CLK.

Table 6-789. Register Call Summary for PRUSS_IEP_LOW_COUNTER

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRUSS_IEP_LOW_COUNTER Register (Offset = 10h) [reset = 0h]: [0]
- PRU-ICSS Industrial Ethernet Timer Basic Programming Sequence: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.6 PRUSS_IEP_HIGH_COUNTER Register (Offset = 14h) [reset = 0h]

PRUSS_IEP_HIGH_COUNTER is shown in Figure 6-308 and described in Table 6-791.

64 bit count value high

Table 6-790. PRUSS_IEP_HIGH_COUNTER Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E014h
PRU_ICSS_1_IEP	20AE E014h

Figure 6-308. PRUSS_IEP_HIGH_COUNTER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-791. PRUSS_IEP_HIGH_COUNTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R/W	0h	64 bit count value (upper 32-bits). Increments by (DEFAULT_INC or CMP_INC) on every positive edge of ICSS_IEP_CLK (200MHz) or ICSS_VCLK_CLK.

Table 6-792. Register Call Summary for PRUSS_IEP_HIGH_COUNTER

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRUSS_IEP_HIGH_COUNTER Register (Offset = 14h) [reset = 0h]: [0]
- PRU-ICSS Industrial Ethernet Timer Basic Programming Sequence: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.7 PRUSS_IEP_CAPTURE_CFG Register (Offset = 18h) [reset = 0001FC00h]

PRUSS_IEP_CAPTURE_CFG is shown in Figure 6-309 and described in Table 6-794.

CAPTURE CFG

Table 6-793. PRUSS_IEP_CAPTURE_CFG Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E018h
PRU_ICSS_1_IEP	20AE E018h

Figure 6-309. PRUSS_IEP_CAPTURE_CFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						CAP_ASYNC_EN	
R-0h						R/W-7Fh	
15	14	13	12	11	10	9	8
CAP_ASYNC_EN						CAP7F_1ST_EVENT_EN	CAP7R_1ST_EVENT_EN
R/W-7Fh						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CAP6F_1ST_EVENT_EN	CAP6R_1ST_EVENT_EN	CAP_1ST_EVENT_EN					
R/W-0h	R/W-0h	R/W-0h					

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-794. PRUSS_IEP_CAPTURE_CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17-10	CAP_ASYNC_EN	R/W	7Fh	Synchronization of the capture inputs to the ICSS_IEP_CLK/ ICSS_VCLK_CLK enable. Note if input capture signal is asynchronous to ICSS_IEP_CLK, enabling synchronization will cause the capture contents to be invalid. CAP_ASYNC_EN[n] maps to CAPR[n]. 0h: Disable synchronization 1h: Enable synchronization
9	CAP7F_1ST_EVENT_EN	R/W	0h	Capture 1st Event Enable for CAP7F. 0h: Continues mode. The capture status is not set when events occur. 1h: First Event mode. The capture status is set when the first event occurs and must be cleared before new data will fill buffer. Time value is captured when first event occurs and held until time is read.
8	CAP7R_1ST_EVENT_EN	R/W	0h	Capture 1st Event Enable for CAP7R. 0h: Continues mode. The capture status is not set when events occur. 1h: First Event mode. The capture status is set when the first event occurs and must be cleared before new data will fill buffer. Time value is captured when first event occurs and held until time is read.

Table 6-794. PRUSS_IEP_CAPTURE_CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	CAP6F_1ST_EVENT_EN	R/W	0h	Capture 1st Event Enable for CAP6F. 0h: Continues mode. The capture status is not set when events occur. 1h: First Event mode. The capture status is set when the first event occurs and must be cleared before new data will fill buffer. Time value is captured when first event occurs and held until time is read.
6	CAP6R_1ST_EVENT_EN	R/W	0h	Capture 1st Event Enable for CAP6R. 0h: Continues mode. The capture status is not set when events occur. 1h: First Event mode. The capture status is set when the first event occurs and must be cleared before new data will fill buffer. Time value is captured when first event occurs and held until time is read.
5-0	CAP_1ST_EVENT_EN	R/W	0h	Capture 1st Event Enable for registers. CAP_1ST_EVENT_EN[n] maps to CAPR[n]. 0h: Continues mode. The capture status is not set when events occur. 1h: First Event mode. The capture status is set when the first event occurs and must be cleared before new data will fill buffer. Time value is captured when first event occurs and held until time is read.

Table 6-795. Register Call Summary for PRUSS_IEP_CAPTURE_CFG

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_CAPTURE_CFG Register (Offset = 18h) [reset = 0001FC00h]: [0]

6.4.11.3.8 PRUSS_IEP_CAPTURE_STATUS Register (Offset = 1Ch) [reset = 0h]

PRUSS_IEP_CAPTURE_STATUS is shown in Figure 6-310 and described in Table 6-797.

CAPTURE STATUS

Table 6-796. PRUSS_IEP_CAPTURE_STATUS Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E01Ch
PRU_ICSS_1_IEP	20AE E01Ch

Figure 6-310. PRUSS_IEP_CAPTURE_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
CAP_RAW							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					CAP_VALID	CAPF7_VALID	CAPR7_VALID
R-0h					R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
CAPF6_VALID	CAPR6_VALID	CAPR_VALID					
R-0h	R-0h	R-0h					

LEGEND: R = Read Only; *-n* = value after reset

Table 6-797. PRUSS_IEP_CAPTURE_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	CAP_RAW	R	0h	Raw/Current status bit for each of the capture registers, where CAP_RAW[n] maps to CAPR[n]. 0h: Current state is low. 1h: Current state is high.
15-11	RESERVED	R	0h	Reserved
10	CAP_VALID	R	0h	Valid status for capture function. Reflects the ORed result from CAP_STATUS[9:0]. 0h: No Hit for any capture event, i.e., there are all 0 in CAP_STATUS[9:0]. 1h: Hit for 1 or more captures events is pending, i.e., there has at least one value equal to 1 in CAP_STATUS[9:0].
9	CAPF7_VALID	R	0h	Valid status for CAPF7 (fall). 0h: No Hit, no capture event occurred 1h: Hit, capture event occurred
8	CAPR7_VALID	R	0h	Valid status for CAPR7 (rise). 0h: No Hit, no capture event occurred 1h: Hit, capture event occurred
7	CAPF6_VALID	R	0h	Valid status for CAPF6 (fall). 0h: No Hit, no capture event occurred 1h: Hit, capture event occurred
6	CAPR6_VALID	R	0h	Valid status for CAPR6 (rise). 0h: No Hit, no capture event occurred 1h: Hit, capture event occurred

Table 6-797. PRUSS_IEP_CAPTURE_STATUS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	CAPR_VALID	R	0h	Valid status bit for each compare register, where CAPR_VALID[n] maps to CAPR[n] (rise). 0h: No Hit, no capture event occurred 1h: Hit, capture event occurred

Table 6-798. Register Call Summary for PRUSS_IEP_CAPTURE_STATUS

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRUSS_IEP_CAPTURE_STATUS Register (Offset = 1Ch) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.9 PRUSS_IEP_CAPTURE_RISE00 Register (Offset = 20h) [reset = 0h]

PRUSS_IEP_CAPTURE_RISE00 is shown in Figure 6-311 and described in Table 6-800.

CAPTURE RISE0 low

Table 6-799. PRUSS_IEP_CAPTURE_RISE00 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E020h
PRU_ICSS_1_IEP	20AE E020h

Figure 6-311. PRUSS_IEP_CAPTURE_RISE00 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-800. PRUSS_IEP_CAPTURE_RISE00 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR	R	0h	Value captured for CAPR<i> (fall) event, where i = 0 to 5. Lower 32-bits.

Table 6-801. Register Call Summary for PRUSS_IEP_CAPTURE_RISE00

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_CAPTURE_RISE00 Register (Offset = 20h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.10 PRUSS_IEP_CAPTURE_RISE10 Register (Offset = 24h) [reset = 0h]

PRUSS_IEP_CAPTURE_RISE10 is shown in Figure 6-312 and described in Table 6-803.

CAPTURE RISE0 high

Table 6-802. PRUSS_IEP_CAPTURE_RISE10 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E024h
PRU_ICSS_1_IEP	20AE E024h

Figure 6-312. PRUSS_IEP_CAPTURE_RISE10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-803. PRUSS_IEP_CAPTURE_RISE10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR	R	0h	Value captured for CAPR<i> (rise) event, where i = 0 to 5. Upper 32-bits.

Table 6-804. Register Call Summary for PRUSS_IEP_CAPTURE_RISE10

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_CAPTURE_RISE10 Register (Offset = 24h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.11 PRUSS_IEP_CAPTURE_RISE01 Register (Offset = 28h) [reset = 0h]

PRUSS_IEP_CAPTURE_RISE01 is shown in Figure 6-313 and described in Table 6-806.

CAPTURE RISE1 low

Table 6-805. PRUSS_IEP_CAPTURE_RISE01 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E028h
PRU_ICSS_1_IEP	20AE E028h

Figure 6-313. PRUSS_IEP_CAPTURE_RISE01 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-806. PRUSS_IEP_CAPTURE_RISE01 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR	R	0h	Value captured for CAPR<i> (fall) event, where i = 0 to 5. Lower 32-bits.

Table 6-807. Register Call Summary for PRUSS_IEP_CAPTURE_RISE01

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_CAPTURE_RISE01 Register (Offset = 28h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.12 PRUSS_IEP_CAPTURE_RISE11 Register (Offset = 2Ch) [reset = 0h]

PRUSS_IEP_CAPTURE_RISE11 is shown in Figure 6-314 and described in Table 6-809.

CAPTURE RISE1 high

Table 6-808. PRUSS_IEP_CAPTURE_RISE11 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E02Ch
PRU_ICSS_1_IEP	20AE E02Ch

Figure 6-314. PRUSS_IEP_CAPTURE_RISE11 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-809. PRUSS_IEP_CAPTURE_RISE11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR	R	0h	Value captured for CAPR<i> (rise) event, where i = 0 to 5. Upper 32-bits.

Table 6-810. Register Call Summary for PRUSS_IEP_CAPTURE_RISE11

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_CAPTURE_RISE11 Register (Offset = 2Ch) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.13 PRUSS_IEP_CAPTURE_RISE02 Register (Offset = 30h) [reset = 0h]

PRUSS_IEP_CAPTURE_RISE02 is shown in Figure 6-315 and described in Table 6-812.

CAPTURE RISE2 low

Table 6-811. PRUSS_IEP_CAPTURE_RISE02 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E030h
PRU_ICSS_1_IEP	20AE E030h

Figure 6-315. PRUSS_IEP_CAPTURE_RISE02 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-812. PRUSS_IEP_CAPTURE_RISE02 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR	R	0h	Value captured for CAPR<i> (fall) event, where i = 0 to 5. Lower 32-bits.

Table 6-813. Register Call Summary for PRUSS_IEP_CAPTURE_RISE02

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_CAPTURE_RISE02 Register (Offset = 30h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.14 PRUSS_IEP_CAPTURE_RISE12 Register (Offset = 34h) [reset = 0h]

PRUSS_IEP_CAPTURE_RISE12 is shown in Figure 6-316 and described in Table 6-815.

CAPTURE RISE2 high

Table 6-814. PRUSS_IEP_CAPTURE_RISE12 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E034h
PRU_ICSS_1_IEP	20AE E034h

Figure 6-316. PRUSS_IEP_CAPTURE_RISE12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-815. PRUSS_IEP_CAPTURE_RISE12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR	R	0h	Value captured for CAPR<i> (rise) event, where i = 0 to 5. Upper 32-bits.

Table 6-816. Register Call Summary for PRUSS_IEP_CAPTURE_RISE12

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_CAPTURE_RISE12 Register (Offset = 34h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.15 PRUSS_IEP_CAPTURE_RISE03 Register (Offset = 38h) [reset = 0h]

PRUSS_IEP_CAPTURE_RISE03 is shown in Figure 6-317 and described in Table 6-818.

CAPTURE RISE3 low

Table 6-817. PRUSS_IEP_CAPTURE_RISE03 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E038h
PRU_ICSS_1_IEP	20AE E038h

Figure 6-317. PRUSS_IEP_CAPTURE_RISE03 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-818. PRUSS_IEP_CAPTURE_RISE03 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR	R	0h	Value captured for CAPR<i> (fall) event, where i = 0 to 5. Lower 32-bits.

Table 6-819. Register Call Summary for PRUSS_IEP_CAPTURE_RISE03

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_CAPTURE_RISE03 Register (Offset = 38h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.16 PRUSS_IEP_CAPTURE_RISE13 Register (Offset = 3Ch) [reset = 0h]

PRUSS_IEP_CAPTURE_RISE13 is shown in Figure 6-318 and described in Table 6-821.

CAPTURE RISE3 high

Table 6-820. PRUSS_IEP_CAPTURE_RISE13 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E03Ch
PRU_ICSS_1_IEP	20AE E03Ch

Figure 6-318. PRUSS_IEP_CAPTURE_RISE13 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-821. PRUSS_IEP_CAPTURE_RISE13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR	R	0h	Value captured for CAPR<i> (rise) event, where i = 0 to 5. Upper 32-bits.

Table 6-822. Register Call Summary for PRUSS_IEP_CAPTURE_RISE13

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_CAPTURE_RISE13 Register (Offset = 3Ch) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.17 PRUSS_IEP_CAPTURE_RISE04 Register (Offset = 40h) [reset = 0h]

PRUSS_IEP_CAPTURE_RISE04 is shown in Figure 6-319 and described in Table 6-824.

CAPTURE RISE4 low

Table 6-823. PRUSS_IEP_CAPTURE_RISE04 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E040h
PRU_ICSS_1_IEP	20AE E040h

Figure 6-319. PRUSS_IEP_CAPTURE_RISE04 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-824. PRUSS_IEP_CAPTURE_RISE04 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR	R	0h	Value captured for CAPR<i> (fall) event, where i = 0 to 5. Lower 32-bits.

Table 6-825. Register Call Summary for PRUSS_IEP_CAPTURE_RISE04

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_CAPTURE_RISE04 Register (Offset = 40h) [reset = 0h]: [0]

6.4.11.3.18 PRUSS_IEP_CAPTURE_RISE14 Register (Offset = 44h) [reset = 0h]

PRUSS_IEP_CAPTURE_RISE14 is shown in Figure 6-320 and described in Table 6-827.

CAPTURE RISE4 high

Table 6-826. PRUSS_IEP_CAPTURE_RISE14 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E044h
PRU_ICSS_1_IEP	20AE E044h

Figure 6-320. PRUSS_IEP_CAPTURE_RISE14 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-827. PRUSS_IEP_CAPTURE_RISE14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR	R	0h	Value captured for CAPR<i> (rise) event, where i = 0 to 5. Upper 32-bits.

Table 6-828. Register Call Summary for PRUSS_IEP_CAPTURE_RISE14

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_CAPTURE_RISE14 Register (Offset = 44h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.19 PRUSS_IEP_CAPTURE_RISE05 Register (Offset = 48h) [reset = 0h]

PRUSS_IEP_CAPTURE_RISE05 is shown in Figure 6-321 and described in Table 6-830.

CAPTURE RISE5 low

Table 6-829. PRUSS_IEP_CAPTURE_RISE05 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E048h
PRU_ICSS_1_IEP	20AE E048h

Figure 6-321. PRUSS_IEP_CAPTURE_RISE05 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-830. PRUSS_IEP_CAPTURE_RISE05 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR	R	0h	Value captured for CAPR<i> (fall) event, where i = 0 to 5. Lower 32-bits.

Table 6-831. Register Call Summary for PRUSS_IEP_CAPTURE_RISE05

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_CAPTURE_RISE05 Register (Offset = 48h) [reset = 0h]: [0]

6.4.11.3.20 PRUSS_IEP_CAPTURE_RISE15 Register (Offset = 4Ch) [reset = 0h]

PRUSS_IEP_CAPTURE_RISE15 is shown in Figure 6-322 and described in Table 6-833.

CAPTURE RISE5 high

Table 6-832. PRUSS_IEP_CAPTURE_RISE15 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E04Ch
PRU_ICSS_1_IEP	20AE E04Ch

Figure 6-322. PRUSS_IEP_CAPTURE_RISE15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-833. PRUSS_IEP_CAPTURE_RISE15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR	R	0h	Value captured for CAPR<i> (rise) event, where i = 0 to 5. Upper 32-bits.

Table 6-834. Register Call Summary for PRUSS_IEP_CAPTURE_RISE15

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_CAPTURE_RISE15 Register (Offset = 4Ch) [reset = 0h]: [0]

6.4.11.3.21 PRUSS_IEP_CAPTURE_RISE06 Register (Offset = 50h) [reset = 0h]

PRUSS_IEP_CAPTURE_RISE06 is shown in Figure 6-323 and described in Table 6-836.

CAPTURE RISE6 low

Table 6-835. PRUSS_IEP_CAPTURE_RISE06 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E050h
PRU_ICSS_1_IEP	20AE E050h

Figure 6-323. PRUSS_IEP_CAPTURE_RISE06 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-836. PRUSS_IEP_CAPTURE_RISE06 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR	R	0h	Value captured for CAPR6 event. Lower 32-bits.

Table 6-837. Register Call Summary for PRUSS_IEP_CAPTURE_RISE06

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRU-ICSS Industrial Ethernet Timer Features: [0]
- PRUSS_IEP_CAPTURE_RISE06 Register (Offset = 50h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.22 PRUSS_IEP_CAPTURE_RISE16 Register (Offset = 54h) [reset = 0h]

PRUSS_IEP_CAPTURE_RISE16 is shown in Figure 6-324 and described in Table 6-839.

CAPTURE RISE6 high

Table 6-838. PRUSS_IEP_CAPTURE_RISE16 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E054h
PRU_ICSS_1_IEP	20AE E054h

Figure 6-324. PRUSS_IEP_CAPTURE_RISE16 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-839. PRUSS_IEP_CAPTURE_RISE16 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR	R	0h	Value captured for CAPR6 event. Upper 32-bits.

Table 6-840. Register Call Summary for PRUSS_IEP_CAPTURE_RISE16

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRU-ICSS Industrial Ethernet Timer Features: [0]
- PRUSS_IEP_CAPTURE_RISE16 Register (Offset = 54h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.23 PRUSS_IEP_CAPTURE_FALL06 Register (Offset = 58h) [reset = 0h]

PRUSS_IEP_CAPTURE_FALL06 is shown in Figure 6-325 and described in Table 6-842.

CAPTURE FALL6 low

Table 6-841. PRUSS_IEP_CAPTURE_FALL06 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E058h
PRU_ICSS_1_IEP	20AE E058h

Figure 6-325. PRUSS_IEP_CAPTURE_FALL06 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPF																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-842. PRUSS_IEP_CAPTURE_FALL06 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPF	R	0h	Value captured for CAPF6 (fall) event. Lower 32-bits.

Table 6-843. Register Call Summary for PRUSS_IEP_CAPTURE_FALL06

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRU-ICSS Industrial Ethernet Timer Features: [0]
- PRUSS_IEP_CAPTURE_FALL06 Register (Offset = 58h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.24 PRUSS_IEP_CAPTURE_FALL16 Register (Offset = 5Ch) [reset = 0h]

PRUSS_IEP_CAPTURE_FALL16 is shown in Figure 6-326 and described in Table 6-845.

CAPTURE FALL6 high

Table 6-844. PRUSS_IEP_CAPTURE_FALL16 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E05Ch
PRU_ICSS_1_IEP	20AE E05Ch

Figure 6-326. PRUSS_IEP_CAPTURE_FALL16 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPF																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-845. PRUSS_IEP_CAPTURE_FALL16 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPF	R	0h	Value captured for CAPF6 (fall) event. Lower 32-bits.

Table 6-846. Register Call Summary for PRUSS_IEP_CAPTURE_FALL16

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRU-ICSS Industrial Ethernet Timer Features: [0]
- PRUSS_IEP_CAPTURE_FALL16 Register (Offset = 5Ch) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.25 PRUSS_IEP_CAPTURE_RISE07 Register (Offset = 60h) [reset = 0h]

PRUSS_IEP_CAPTURE_RISE07 is shown in Figure 6-327 and described in Table 6-848.

CAPTURE RISE7 low

Table 6-847. PRUSS_IEP_CAPTURE_RISE07 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E060h
PRU_ICSS_1_IEP	20AE E060h

Figure 6-327. PRUSS_IEP_CAPTURE_RISE07 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-848. PRUSS_IEP_CAPTURE_RISE07 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR	R	0h	Value captured for CAPR7 (rise) event. Lower 32-bits.

Table 6-849. Register Call Summary for PRUSS_IEP_CAPTURE_RISE07

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRU-ICSS Industrial Ethernet Timer Features: [0]
- PRUSS_IEP_CAPTURE_RISE07 Register (Offset = 60h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.26 PRUSS_IEP_CAPTURE_RISE17 Register (Offset = 64h) [reset = 0h]

PRUSS_IEP_CAPTURE_RISE17 is shown in Figure 6-328 and described in Table 6-851.

CAPTURE RISE7 high

Table 6-850. PRUSS_IEP_CAPTURE_RISE17 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E064h
PRU_ICSS_1_IEP	20AE E064h

Figure 6-328. PRUSS_IEP_CAPTURE_RISE17 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-851. PRUSS_IEP_CAPTURE_RISE17 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR	R	0h	Value captured for CAPR7 (rise) event. Upper 32-bits.

Table 6-852. Register Call Summary for PRUSS_IEP_CAPTURE_RISE17

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRU-ICSS Industrial Ethernet Timer Features: [0]
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_CAPTURE_RISE17 Register (Offset = 64h) [reset = 0h]: [0]

6.4.11.3.27 PRUSS_IEP_CAPTURE_FALL07 Register (Offset = 68h) [reset = 0h]

PRUSS_IEP_CAPTURE_FALL07 is shown in Figure 6-329 and described in Table 6-854.

CAPTURE FALL7 low

Table 6-853. PRUSS_IEP_CAPTURE_FALL07 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E068h
PRU_ICSS_1_IEP	20AE E068h

Figure 6-329. PRUSS_IEP_CAPTURE_FALL07 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPF																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-854. PRUSS_IEP_CAPTURE_FALL07 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPF	R	0h	Value captured for CAPF7 (fall) event. Lower 32-bits.

Table 6-855. Register Call Summary for PRUSS_IEP_CAPTURE_FALL07

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRU-ICSS Industrial Ethernet Timer Features: [0]
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_CAPTURE_FALL07 Register (Offset = 68h) [reset = 0h]: [0]

6.4.11.3.28 PRUSS_IEP_CAPTURE_FALL17 Register (Offset = 6Ch) [reset = 0h]

PRUSS_IEP_CAPTURE_FALL17 is shown in Figure 6-330 and described in Table 6-857.

CAPTURE FALL7 high

Table 6-856. PRUSS_IEP_CAPTURE_FALL17 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E06Ch
PRU_ICSS_1_IEP	20AE E06Ch

Figure 6-330. PRUSS_IEP_CAPTURE_FALL17 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPF																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-857. PRUSS_IEP_CAPTURE_FALL17 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPF	R	0h	Value captured for CAPF7 (fall) event. Upper 32-bits.

Table 6-858. Register Call Summary for PRUSS_IEP_CAPTURE_FALL17

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRU-ICSS Industrial Ethernet Timer Features: [0]
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_CAPTURE_FALL17 Register (Offset = 6Ch) [reset = 0h]: [0]

6.4.11.3.29 PRUSS_IEP_COMPARE_CFG Register (Offset = 70h) [reset = 0h]

PRUSS_IEP_COMPARE_CFG is shown in Figure 6-331 and described in Table 6-860.

COMPARE CFG

Table 6-859. PRUSS_IEP_COMPARE_CFG Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E070h
PRU_ICSS_1_IEP	20AE E070h

Figure 6-331. PRUSS_IEP_COMPARE_CFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							CMP_EN
R-0h							R/W-0h
15	14	13	12	11	10	9	8
CMP_EN							
R/W-0h							
7	6	5	4	3	2	1	0
CMP_EN							CMP0_RST_CNT_EN
R/W-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-860. PRUSS_IEP_COMPARE_CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16-1	CMP_EN	R/W	0h	Enable bits for each of the compare registers CMP_EN =0 : Disables CMPj/k Event CMP_EN=1: Enables CMPj/k Event CMP_EN[0] (bit 1 of register) maps to CMP0 event
0	CMP0_RST_CNT_EN	R/W	0h	Enable the reset of the counter 0h: Disable 1h: Enable the reset of the counter if a CMP0 event occurs

Table 6-861. Register Call Summary for PRUSS_IEP_COMPARE_CFG

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRUSS_IEP_COMPARE_CFG Register (Offset = 70h) [reset = 0h]: [0]
- PRU-ICSS Industrial Ethernet Timer Features: [0]
- PRU-ICSS Industrial Ethernet Timer Basic Programming Sequence: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.30 PRUSS_IEP_COMPARE_STATUS Register (Offset = 74h) [reset = 0h]

PRUSS_IEP_COMPARE_STATUS is shown in Figure 6-332 and described in Table 6-863.

COMPARE STATUS

Table 6-862. PRUSS_IEP_COMPARE_STATUS Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E074h
PRU_ICSS_1_IEP	20AE E074h

Figure 6-332. PRUSS_IEP_COMPARE_STATUS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CMP_HIT															
R-0h																RWr1Clr-0h															

LEGEND: R = Read Only; RWr1Clr = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 6-863. PRUSS_IEP_COMPARE_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	CMP_HIT	RWr1Clr	0h	Status bit for each of the compare registers. "Match" indicates the current counter is greater than or equal to the compare value. Note it is the firmware's responsibility to handle the IEP overflow. CMP_HITj/k = 0: No match has occured CMP_HITj/k = 1: A match occured. The associated hardware event signal will assert and remain high until the status is cleared.

Table 6-864. Register Call Summary for PRUSS_IEP_COMPARE_STATUS

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRUSS_IEP_COMPARE_STATUS Register (Offset = 74h) [reset = 0h]: [0]
- PRU-ICSS Industrial Ethernet Timer Features: [0]
- PRU-ICSS Industrial Ethernet Timer Basic Programming Sequence: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.31 PRUSS_IEP_COMPARE00 Register (Offset = 78h) [reset = 0h]

PRUSS_IEP_COMPARE00 is shown in Figure 6-333 and described in Table 6-866.

COMPARE0 low

Table 6-865. PRUSS_IEP_COMPARE00 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E078h
PRU_ICSS_1_IEP	20AE E078h

Figure 6-333. PRUSS_IEP_COMPARE00 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-866. PRUSS_IEP_COMPARE00 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 0 low value

Table 6-867. Register Call Summary for PRUSS_IEP_COMPARE00

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE00 Register (Offset = 78h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.32 PRUSS_IEP_COMPARE10 Register (Offset = 7Ch) [reset = 0h]

PRUSS_IEP_COMPARE10 is shown in Figure 6-334 and described in Table 6-869.

COMPARE0 high

Table 6-868. PRUSS_IEP_COMPARE10 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E07Ch
PRU_ICSS_1_IEP	20AE E07Ch

Figure 6-334. PRUSS_IEP_COMPARE10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-869. PRUSS_IEP_COMPARE10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 0 high value

Table 6-870. Register Call Summary for PRUSS_IEP_COMPARE10

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_COMPARE10 Register (Offset = 7Ch) [reset = 0h]: [0]

6.4.11.3.33 PRUSS_IEP_COMPARE01 Register (Offset = 80h) [reset = 0h]

PRUSS_IEP_COMPARE01 is shown in Figure 6-335 and described in Table 6-872.

COMPARE1 low

Table 6-871. PRUSS_IEP_COMPARE01 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E080h
PRU_ICSS_1_IEP	20AE E080h

Figure 6-335. PRUSS_IEP_COMPARE01 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-872. PRUSS_IEP_COMPARE01 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 1 low value

Table 6-873. Register Call Summary for PRUSS_IEP_COMPARE01

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE01 Register (Offset = 80h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.34 PRUSS_IEP_COMPARE11 Register (Offset = 84h) [reset = 0h]

PRUSS_IEP_COMPARE11 is shown in Figure 6-336 and described in Table 6-875.

COMPARE1 high

Table 6-874. PRUSS_IEP_COMPARE11 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E084h
PRU_ICSS_1_IEP	20AE E084h

Figure 6-336. PRUSS_IEP_COMPARE11 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-875. PRUSS_IEP_COMPARE11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 1 high value

Table 6-876. Register Call Summary for PRUSS_IEP_COMPARE11

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE11 Register (Offset = 84h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.35 PRUSS_IEP_COMPARE02 Register (Offset = 88h) [reset = 0h]

PRUSS_IEP_COMPARE02 is shown in Figure 6-337 and described in Table 6-878.

COMPARE2 low

Table 6-877. PRUSS_IEP_COMPARE02 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E088h
PRU_ICSS_1_IEP	20AE E088h

Figure 6-337. PRUSS_IEP_COMPARE02 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-878. PRUSS_IEP_COMPARE02 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 2 low value

Table 6-879. Register Call Summary for PRUSS_IEP_COMPARE02

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE02 Register (Offset = 88h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.36 PRUSS_IEP_COMPARE12 Register (Offset = 8Ch) [reset = 0h]

PRUSS_IEP_COMPARE12 is shown in Figure 6-338 and described in Table 6-881.

COMPARE2 high

Table 6-880. PRUSS_IEP_COMPARE12 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E08Ch
PRU_ICSS_1_IEP	20AE E08Ch

Figure 6-338. PRUSS_IEP_COMPARE12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-881. PRUSS_IEP_COMPARE12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 2 high value

Table 6-882. Register Call Summary for PRUSS_IEP_COMPARE12

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_COMPARE12 Register (Offset = 8Ch) [reset = 0h]: [0]

6.4.11.3.37 PRUSS_IEP_COMPARE03 Register (Offset = 90h) [reset = 0h]

PRUSS_IEP_COMPARE03 is shown in Figure 6-339 and described in Table 6-884.

COMPARE3 low

Table 6-883. PRUSS_IEP_COMPARE03 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E090h
PRU_ICSS_1_IEP	20AE E090h

Figure 6-339. PRUSS_IEP_COMPARE03 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-884. PRUSS_IEP_COMPARE03 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 3 low value

Table 6-885. Register Call Summary for PRUSS_IEP_COMPARE03

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE03 Register (Offset = 90h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.38 PRUSS_IEP_COMPARE13 Register (Offset = 94h) [reset = 0h]

PRUSS_IEP_COMPARE13 is shown in Figure 6-340 and described in Table 6-887.

COMPARE3 high

Table 6-886. PRUSS_IEP_COMPARE13 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E094h
PRU_ICSS_1_IEP	20AE E094h

Figure 6-340. PRUSS_IEP_COMPARE13 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-887. PRUSS_IEP_COMPARE13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 3 high value

Table 6-888. Register Call Summary for PRUSS_IEP_COMPARE13

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_COMPARE13 Register (Offset = 94h) [reset = 0h]: [0]

6.4.11.3.39 PRUSS_IEP_COMPARE04 Register (Offset = 98h) [reset = 0h]

PRUSS_IEP_COMPARE04 is shown in Figure 6-341 and described in Table 6-890.

COMPARE4 low

Table 6-889. PRUSS_IEP_COMPARE04 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E098h
PRU_ICSS_1_IEP	20AE E098h

Figure 6-341. PRUSS_IEP_COMPARE04 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-890. PRUSS_IEP_COMPARE04 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 4 low value

Table 6-891. Register Call Summary for PRUSS_IEP_COMPARE04

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_COMPARE04 Register (Offset = 98h) [reset = 0h]: [0]

6.4.11.3.40 PRUSS_IEP_COMPARE14 Register (Offset = 9Ch) [reset = 0h]

PRUSS_IEP_COMPARE14 is shown in Figure 6-342 and described in Table 6-893.

COMPARE4 high

Table 6-892. PRUSS_IEP_COMPARE14 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E09Ch
PRU_ICSS_1_IEP	20AE E09Ch

Figure 6-342. PRUSS_IEP_COMPARE14 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-893. PRUSS_IEP_COMPARE14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 4 high value

Table 6-894. Register Call Summary for PRUSS_IEP_COMPARE14

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE14 Register (Offset = 9Ch) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.41 PRUSS_IEP_COMPARE05 Register (Offset = A0h) [reset = 0h]

PRUSS_IEP_COMPARE05 is shown in Figure 6-343 and described in Table 6-896.

COMPARE5 low

Table 6-895. PRUSS_IEP_COMPARE05 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0A0h
PRU_ICSS_1_IEP	20AE E0A0h

Figure 6-343. PRUSS_IEP_COMPARE05 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-896. PRUSS_IEP_COMPARE05 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 5 low value

Table 6-897. Register Call Summary for PRUSS_IEP_COMPARE05

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_COMPARE05 Register (Offset = A0h) [reset = 0h]: [0]

6.4.11.3.42 PRUSS_IEP_COMPARE15 Register (Offset = A4h) [reset = 0h]

PRUSS_IEP_COMPARE15 is shown in Figure 6-344 and described in Table 6-899.

COMPARE5 high

Table 6-898. PRUSS_IEP_COMPARE15 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0A4h
PRU_ICSS_1_IEP	20AE E0A4h

Figure 6-344. PRUSS_IEP_COMPARE15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-899. PRUSS_IEP_COMPARE15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 5 high value

Table 6-900. Register Call Summary for PRUSS_IEP_COMPARE15

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE15 Register (Offset = A4h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.43 PRUSS_IEP_COMPARE06 Register (Offset = A8h) [reset = 0h]

PRUSS_IEP_COMPARE06 is shown in Figure 6-345 and described in Table 6-902.

COMPARE6 low

Table 6-901. PRUSS_IEP_COMPARE06 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0A8h
PRU_ICSS_1_IEP	20AE E0A8h

Figure 6-345. PRUSS_IEP_COMPARE06 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-902. PRUSS_IEP_COMPARE06 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 6 low value

Table 6-903. Register Call Summary for PRUSS_IEP_COMPARE06

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE06 Register (Offset = A8h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.44 PRUSS_IEP_COMPARE16 Register (Offset = ACh) [reset = 0h]

PRUSS_IEP_COMPARE16 is shown in Figure 6-346 and described in Table 6-905.

COMPARE6 high

Table 6-904. PRUSS_IEP_COMPARE16 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0ACh
PRU_ICSS_1_IEP	20AE E0ACh

Figure 6-346. PRUSS_IEP_COMPARE16 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-905. PRUSS_IEP_COMPARE16 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 6 high value

Table 6-906. Register Call Summary for PRUSS_IEP_COMPARE16

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRUSS_IEP_COMPARE16 Register (Offset = ACh) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.45 PRUSS_IEP_COMPARE07 Register (Offset = B0h) [reset = 0h]

PRUSS_IEP_COMPARE07 is shown in Figure 6-347 and described in Table 6-908.

COMPARE7 low

Table 6-907. PRUSS_IEP_COMPARE07 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0B0h
PRU_ICSS_1_IEP	20AE E0B0h

Figure 6-347. PRUSS_IEP_COMPARE07 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-908. PRUSS_IEP_COMPARE07 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 7 low value

Table 6-909. Register Call Summary for PRUSS_IEP_COMPARE07

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_COMPARE07 Register (Offset = B0h) [reset = 0h]: [0]

6.4.11.3.46 PRUSS_IEP_COMPARE17 Register (Offset = B4h) [reset = 0h]

PRUSS_IEP_COMPARE17 is shown in Figure 6-348 and described in Table 6-911.

COMPARE7 high

Table 6-910. PRUSS_IEP_COMPARE17 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0B4h
PRU_ICSS_1_IEP	20AE E0B4h

Figure 6-348. PRUSS_IEP_COMPARE17 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-911. PRUSS_IEP_COMPARE17 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 7 high value

Table 6-912. Register Call Summary for PRUSS_IEP_COMPARE17

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE17 Register (Offset = B4h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.47 PRUSS_IEP_RXIPG0 Register (Offset = B8h) [reset = FFFF0000h]

PRUSS_IEP_RXIPG0 is shown in Figure 6-349 and described in Table 6-914.

This register can be used to determine the last RX IPG and the smallest RX IPG. RXIPG0 is the status for the RX port which is attached to PRU0.

Table 6-913. PRUSS_IEP_RXIPG0 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0B8h
PRU_ICSS_1_IEP	20AE E0B8h

Figure 6-349. PRUSS_IEP_RXIPG0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX_MIN_IPG																RX_IPG															
R/W-FFFFh																R-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-914. PRUSS_IEP_RXIPG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RX_MIN_IPG	R/W	FFFFh	Defines the minimum number of ICSS_IEP_CLK/ICCS_VCLK_CLK cycles that is RX_DV is sampled low. It stores the smallest RX_IPG duration. It can be read at any time and gets updated after RX_IPG is updated, if RX_MIN_IPG is greater than RX_IPG.
15-0	RX_IPG	R	0h	Records the current number of ICSS_IEP_CLK/ICSS_VCLK_CLK cycles RX_DV is sampled low. Value is updated after RX_DV transitions from low to high. It will saturate at FFFFh.

Table 6-915. Register Call Summary for PRUSS_IEP_RXIPG0

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_RXIPG0 Register (Offset = B8h) [reset = FFFF0000h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.48 PRUSS_IEP_RXIPG1 Register (Offset = BCh) [reset = FFFF0000h]

PRUSS_IEP_RXIPG1 is shown in Figure 6-350 and described in Table 6-917.

This register can be used to determine the last RX IPG and the smallest RX IPG. RXIPG1 is the status for the RX port which is attached to PRU1

Table 6-916. PRUSS_IEP_RXIPG1 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0BCh
PRU_ICSS_1_IEP	20AE E0BCh

Figure 6-350. PRUSS_IEP_RXIPG1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX_MIN_IPG																RX_IPG															
R/W-FFFFh																R-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-917. PRUSS_IEP_RXIPG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RX_MIN_IPG	R/W	FFFFh	Defines the minimum number of ICSS_IEP_CLK/ICSS_VCLK_CLK cycles that is RX_DV is sampled low. It stores the smallest RX_IPG duration. It can be read at any time and gets updated after RX_IPG is updated, if RX_MIN_IPG is greater than RX_IPG.
15-0	RX_IPG	R	0h	Records the current number of ICSS_IEP_CLK/ICSS_VCLK_CLK cycles RX_DV is sampled low. Value is updated after RX_DV transitions from low to high. It will saturate at FFFFh.

Table 6-918. Register Call Summary for PRUSS_IEP_RXIPG1

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_RXIPG1 Register (Offset = BCh) [reset = FFFF0000h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.49 PRUSS_IEP_COMPARE08 Register (Offset = C0h) [reset = 0h]

PRUSS_IEP_COMPARE08 is shown in Figure 6-351 and described in Table 6-920.

COMPARE8 low

Table 6-919. PRUSS_IEP_COMPARE08 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0C0h
PRU_ICSS_1_IEP	20AE E0C0h

Figure 6-351. PRUSS_IEP_COMPARE08 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-920. PRUSS_IEP_COMPARE08 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 8 low value

Table 6-921. Register Call Summary for PRUSS_IEP_COMPARE08

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE08 Register (Offset = C0h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.50 PRUSS_IEP_COMPARE18 Register (Offset = C4h) [reset = 0h]

PRUSS_IEP_COMPARE18 is shown in Figure 6-352 and described in Table 6-923.

COMPARE8 high

Table 6-922. PRUSS_IEP_COMPARE18 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0C4h
PRU_ICSS_1_IEP	20AE E0C4h

Figure 6-352. PRUSS_IEP_COMPARE18 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-923. PRUSS_IEP_COMPARE18 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Reset value (upper 32-bits). This register enables SW to define the reset state of the Master Counter, which can be reset by the following events (if enabled): CMP0 event; eHRPWM0_SYNCO event; eHRPWM3_SYNCO event. The RESET_VAL should be in increments of the DEFAULT_INC (default state is 5). For example, 0000_000Ah.

Table 6-924. Register Call Summary for PRUSS_IEP_COMPARE18

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRUSS_IEP_COMPARE18 Register (Offset = C4h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.51 PRUSS_IEP_COMPARE09 Register (Offset = C8h) [reset = 0h]

PRUSS_IEP_COMPARE09 is shown in Figure 6-353 and described in Table 6-926.

COMPARE9 low

Table 6-925. PRUSS_IEP_COMPARE09 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0C8h
PRU_ICSS_1_IEP	20AE E0C8h

Figure 6-353. PRUSS_IEP_COMPARE09 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-926. PRUSS_IEP_COMPARE09 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 9 low value

Table 6-927. Register Call Summary for PRUSS_IEP_COMPARE09

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE09 Register (Offset = C8h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.52 PRUSS_IEP_COMPARE19 Register (Offset = CCh) [reset = 0h]

PRUSS_IEP_COMPARE19 is shown in Figure 6-354 and described in Table 6-929.

COMPARE9 high

Table 6-928. PRUSS_IEP_COMPARE19 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0CCh
PRU_ICSS_1_IEP	20AE E0CCh

Figure 6-354. PRUSS_IEP_COMPARE19 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-929. PRUSS_IEP_COMPARE19 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 9 high value

Table 6-930. Register Call Summary for PRUSS_IEP_COMPARE19

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_COMPARE19 Register (Offset = CCh) [reset = 0h]: [0]

6.4.11.3.53 PRUSS_IEP_COMPARE010 Register (Offset = D0h) [reset = 0h]

PRUSS_IEP_COMPARE010 is shown in Figure 6-355 and described in Table 6-932.

COMPARE10 low

Table 6-931. PRUSS_IEP_COMPARE010 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0D0h
PRU_ICSS_1_IEP	20AE E0D0h

Figure 6-355. PRUSS_IEP_COMPARE010 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-932. PRUSS_IEP_COMPARE010 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 10 low value

Table 6-933. Register Call Summary for PRUSS_IEP_COMPARE010

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE010 Register (Offset = D0h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.54 PRUSS_IEP_COMPARE110 Register (Offset = D4h) [reset = 0h]

PRUSS_IEP_COMPARE110 is shown in Figure 6-356 and described in Table 6-935.

COMPARE10 high

Table 6-934. PRUSS_IEP_COMPARE110 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0D4h
PRU_ICSS_1_IEP	20AE E0D4h

Figure 6-356. PRUSS_IEP_COMPARE110 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-935. PRUSS_IEP_COMPARE110 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 10 high value

Table 6-936. Register Call Summary for PRUSS_IEP_COMPARE110

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_COMPARE110 Register (Offset = D4h) [reset = 0h]: [0]

6.4.11.3.55 PRUSS_IEP_COMPARE011 Register (Offset = D8h) [reset = 0h]

PRUSS_IEP_COMPARE011 is shown in Figure 6-357 and described in Table 6-938.

COMPARE11 low

Table 6-937. PRUSS_IEP_COMPARE011 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0D8h
PRU_ICSS_1_IEP	20AE E0D8h

Figure 6-357. PRUSS_IEP_COMPARE011 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-938. PRUSS_IEP_COMPARE011 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 11 low value

Table 6-939. Register Call Summary for PRUSS_IEP_COMPARE011

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE011 Register (Offset = D8h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.56 PRUSS_IEP_COMPARE111 Register (Offset = DCh) [reset = 0h]

PRUSS_IEP_COMPARE111 is shown in Figure 6-358 and described in Table 6-941.

COMPARE11 high

Table 6-940. PRUSS_IEP_COMPARE111 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0DCh
PRU_ICSS_1_IEP	20AE E0DCh

Figure 6-358. PRUSS_IEP_COMPARE111 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-941. PRUSS_IEP_COMPARE111 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 11 high value

Table 6-942. Register Call Summary for PRUSS_IEP_COMPARE111

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_COMPARE111 Register (Offset = DCh) [reset = 0h]: [0]

6.4.11.3.57 PRUSS_IEP_COMPARE012 Register (Offset = E0h) [reset = 0h]

PRUSS_IEP_COMPARE012 is shown in Figure 6-359 and described in Table 6-944.

COMPARE12 low

Table 6-943. PRUSS_IEP_COMPARE012 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0E0h
PRU_ICSS_1_IEP	20AE E0E0h

Figure 6-359. PRUSS_IEP_COMPARE012 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-944. PRUSS_IEP_COMPARE012 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 12 low value

Table 6-945. Register Call Summary for PRUSS_IEP_COMPARE012

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE012 Register (Offset = E0h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.58 PRUSS_IEP_COMPARE112 Register (Offset = E4h) [reset = 0h]

PRUSS_IEP_COMPARE112 is shown in Figure 6-360 and described in Table 6-947.

COMPARE12 high

Table 6-946. PRUSS_IEP_COMPARE112 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0E4h
PRU_ICSS_1_IEP	20AE E0E4h

Figure 6-360. PRUSS_IEP_COMPARE112 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-947. PRUSS_IEP_COMPARE112 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 12 high value

Table 6-948. Register Call Summary for PRUSS_IEP_COMPARE112

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE112 Register (Offset = E4h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.59 PRUSS_IEP_COMPARE013 Register (Offset = E8h) [reset = 0h]

PRUSS_IEP_COMPARE013 is shown in Figure 6-361 and described in Table 6-950.

COMPARE13 low

Table 6-949. PRUSS_IEP_COMPARE013 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0E8h
PRU_ICSS_1_IEP	20AE E0E8h

Figure 6-361. PRUSS_IEP_COMPARE013 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-950. PRUSS_IEP_COMPARE013 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 13 low value

Table 6-951. Register Call Summary for PRUSS_IEP_COMPARE013

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE013 Register (Offset = E8h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.60 PRUSS_IEP_COMPARE113 Register (Offset = ECh) [reset = 0h]

PRUSS_IEP_COMPARE113 is shown in Figure 6-362 and described in Table 6-953.

COMPARE13 high

Table 6-952. PRUSS_IEP_COMPARE113 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0ECh
PRU_ICSS_1_IEP	20AE E0ECh

Figure 6-362. PRUSS_IEP_COMPARE113 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-953. PRUSS_IEP_COMPARE113 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 13 high value

Table 6-954. Register Call Summary for PRUSS_IEP_COMPARE113

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_COMPARE113 Register (Offset = ECh) [reset = 0h]: [0]

6.4.11.3.61 PRUSS_IEP_COMPARE014 Register (Offset = F0h) [reset = 0h]

PRUSS_IEP_COMPARE014 is shown in Figure 6-363 and described in Table 6-956.

COMPARE14 low

Table 6-955. PRUSS_IEP_COMPARE014 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0F0h
PRU_ICSS_1_IEP	20AE E0F0h

Figure 6-363. PRUSS_IEP_COMPARE014 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-956. PRUSS_IEP_COMPARE014 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 14 low value

Table 6-957. Register Call Summary for PRUSS_IEP_COMPARE014

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_COMPARE014 Register (Offset = F0h) [reset = 0h]: [0]

6.4.11.3.62 PRUSS_IEP_COMPARE114 Register (Offset = F4h) [reset = 0h]

PRUSS_IEP_COMPARE114 is shown in Figure 6-364 and described in Table 6-959.

COMPARE14 high

Table 6-958. PRUSS_IEP_COMPARE114 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0F4h
PRU_ICSS_1_IEP	20AE E0F4h

Figure 6-364. PRUSS_IEP_COMPARE114 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-959. PRUSS_IEP_COMPARE114 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 14 high value

Table 6-960. Register Call Summary for PRUSS_IEP_COMPARE114

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE114 Register (Offset = F4h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.63 PRUSS_IEP_COMPARE015 Register (Offset = F8h) [reset = 0h]

PRUSS_IEP_COMPARE015 is shown in Figure 6-365 and described in Table 6-962.

COMPARE15 low

Table 6-961. PRUSS_IEP_COMPARE015 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0F8h
PRU_ICSS_1_IEP	20AE E0F8h

Figure 6-365. PRUSS_IEP_COMPARE015 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-962. PRUSS_IEP_COMPARE015 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 15 low value

Table 6-963. Register Call Summary for PRUSS_IEP_COMPARE015

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_COMPARE015 Register (Offset = F8h) [reset = 0h]: [0]

6.4.11.3.64 PRUSS_IEP_COMPARE115 Register (Offset = FCh) [reset = 0h]

PRUSS_IEP_COMPARE115 is shown in Figure 6-366 and described in Table 6-965.

COMPARE15 high

Table 6-964. PRUSS_IEP_COMPARE115 Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E0FCh
PRU_ICSS_1_IEP	20AE E0FCh

Figure 6-366. PRUSS_IEP_COMPARE115 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-965. PRUSS_IEP_COMPARE115 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP	R/W	0h	Compare 15 high value

Table 6-966. Register Call Summary for PRUSS_IEP_COMPARE115

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_COMPARE115 Register (Offset = FCh) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.65 PRUSS_IEP_LOW_COUNTER_RESET_VALUE Register (Offset = 100h) [reset = 0h]

PRUSS_IEP_LOW_COUNTER_RESET_VALUE is shown in Figure 6-367 and described in Table 6-968.

Reset value of the Master Counter (lower 32-bits).

Table 6-967. PRUSS_IEP_LOW_COUNTER_RESET_VALUE Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E100h
PRU_ICSS_1_IEP	20AE E100h

Figure 6-367. PRUSS_IEP_LOW_COUNTER_RESET_VALUE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESET_VAL																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-968. PRUSS_IEP_LOW_COUNTER_RESET_VALUE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESET_VAL	R/W	0h	Reset value (lower 32-bits). This register enables SW to define the reset state of the Master Counter, which can be reset by the following events (if enabled): CMP0 event; eHRPWM0_SYNCO event; eHRPWM3_SYNCO event. The RESET_VAL should be in increments of the DEFAULT_INC (default state is 5). For example, 0000_000Ah.

Table 6-969. Register Call Summary for PRUSS_IEP_LOW_COUNTER_RESET_VALUE

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_LOW_COUNTER_RESET_VALUE Register (Offset = 100h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.66 PRUSS_IEP_HIGH_COUNTER_RESET_VALUE Register (Offset = 104h) [reset = 0h]

PRUSS_IEP_HIGH_COUNTER_RESET_VALUE is shown in Figure 6-368 and described in Table 6-971.

Reset value of the Master Counter (upper 32-bits).

Table 6-970. PRUSS_IEP_HIGH_COUNTER_RESET_VALUE Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E104h
PRU_ICSS_1_IEP	20AE E104h

Figure 6-368. PRUSS_IEP_HIGH_COUNTER_RESET_VALUE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESET_VAL																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-971. PRUSS_IEP_HIGH_COUNTER_RESET_VALUE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESET_VAL	R/W	0h	This enables SW to define the reset state of the Master counter when it gets reset do to the following 3 possible events if enabled: CMP0 event; eHRPWM0_SYNCO event; eHRPWM3_SYNCO event. It should be in increments of the DEFAULT_INC, default state is 5 For example, 0000_000Ah

Table 6-972. Register Call Summary for PRUSS_IEP_HIGH_COUNTER_RESET_VALUE

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_HIGH_COUNTER_RESET_VALUE Register (Offset = 104h) [reset = 0h]: [0]

6.4.11.3.67 PRUSS_IEP_PWM Register (Offset = 108h) [reset = 0h]

PRUSS_IEP_PWM is shown in Figure 6-369 and described in Table 6-974.

PWM Sync Out

Table 6-973. PRUSS_IEP_PWM Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E108h
PRU_ICSS_1_IEP	20AE E108h

Figure 6-369. PRUSS_IEP_PWM Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				PWM3_HIT	PWM3_RST_CNT_EN	PWM0_HIT	PWM0_RST_CNT_EN
R-0h				RW1Clr-0h	R/W-0h	RW1Clr-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; RW1Clr = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 6-974. PRUSS_IEP_PWM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	PWM3_HIT	RW1Clr	0h	The raw status bit of eHRPWM3_SYNCO event. 0h: No eHRPWM3_SYNCO event 1h: eHRPWM3_SYNCO event occurred Write 1h to Clear.
2	PWM3_RST_CNT_EN	R/W	0h	Enable the reset of the counter by a eHRPWM3_SYNCO event. 0h: Disable 1h: Enable the reset of the counter if a eHRPWM3_SYNCO event occurs
1	PWM0_HIT	RW1Clr	0h	The raw status bit of eHRPWM0_SYNCO event. 0h: No eHRPWM0_SYNCO event 1h: eHRPWM0_SYNCO event occurred Write 1 to Clear.
0	PWM0_RST_CNT_EN	R/W	0h	Enable the reset of the counter by a eHRPWM0_SYNCO event. 0h: Disable 1h: Enable the reset of the counter if a eHRPWM0_SYNCO event occurs

Table 6-975. Register Call Summary for PRUSS_IEP_PWM

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRUSS_IEP_PWM Register (Offset = 108h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.68 PRUSS_IEP_SYNC_CTRL Register (Offset = 180h) [reset = 0h]

PRUSS_IEP_SYNC_CTRL is shown in Figure 6-370 and described in Table 6-977.

SYNC GENERATION CONTROL

Table 6-976. PRUSS_IEP_SYNC_CTRL Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E180h
PRU_ICSS_1_IEP	20AE E180h

Figure 6-370. PRUSS_IEP_SYNC_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							SYNC1_IND_EN
R-0h							R/W-0h
7	6	5	4	3	2	1	0
SYNC1_CYCLIC_EN	SYNC1_ACK_EN	SYNC0_CYCLIC_EN	SYNC0_ACK_EN	RESERVED	SYNC1_EN	SYNC0_EN	SYNC_EN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-977. PRUSS_IEP_SYNC_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	SYNC1_IND_EN	R/W	0h	SYNC1 independent mode enable. Independent mode means the SYNC1 signal can be different from SYNC0. 0h: Dependent mode 1h: Independent mode
7	SYNC1_CYCLIC_EN	R/W	0h	SYNC1 single shot or cyclic/auto generation mode enable 0h: Disable, single shot mode 1h: Enable, cyclic generation mode
6	SYNC1_ACK_EN	R/W	0h	SYNC1 acknowledgement mode enable 0h: Disable, SYNC1 will go low after pulse width is met. 1h: Enable, SYNC1 will remain asserted until receiving software acknowledges by reading PRUSS_IEP_SYNC1_STAT which clears on read.
5	SYNC0_CYCLIC_EN	R/W	0h	SYNC0 single shot or cyclic/auto generation mode enable 0h: Disable, single shot mode 1h: Enable, cyclic generation mode
4	SYNC0_ACK_EN	R/W	0h	SYNC0 acknowledgement mode enable 0h: Disable, SYNC0 will go low after pulse width is met. 1h: Enable, SYNC0 will remain asserted until receiving software acknowledges by reading PRUSS_IEP_SYNC0_STAT which clears on read.
3	RESERVED	R	0h	Reserved

Table 6-977. PRUSS_IEP_SYNC_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	SYNC1_EN	R/W	0h	SYNC1 generation enable 0h: Disable SYNC1 generation. If SYNC1 is low, it will stop immediately. If SYNC1 is high, it will stop after SYNC1 goes low 1h: Enable SYNC1 generation
1	SYNC0_EN	R/W	0h	SYNC0 generation enable 0h: Disable SYNC0 generation. If SYNC0 is low, it will stop immediately If SYNC0 is high, it will stop after SYNC0 goes low 1h: Enable SYNC0 generation
0	SYNC_EN	R/W	0h	SYNC generation enable 0h: Disable the generation and clocking of SYNC0 and SYNC1 logic. If SYNC0 AND SYNC1 is low, it will stop immediately. If SYNC0 OR SYNC1 is high, it will stop after SYCN0 AND SYNC1 goes low. Note that you might get 1 extra high pulse if this is disabled during a high pulse of one and the 2nd pulse goes high before the last pulse low if you do not de-assert sync0_en and sync1_en at the same time. SW should always de-assert both sync1_en and sync0_en at the same time as sync_en is de-asserted 1h: Enables SYNC0 and SYNC1 generation

Table 6-978. Register Call Summary for PRUSS_IEP_SYNC_CTRL

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRUSS_IEP_SYNC_CTRL Register (Offset = 180h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]
- : [4]

6.4.11.3.69 PRUSS_IEP_SYNC_FIRST_STAT Register (Offset = 184h) [reset = 0h]

PRUSS_IEP_SYNC_FIRST_STAT is shown in Figure 6-371 and described in Table 6-980.

SYNC GENERATION FIRST EVENT STATUS

Table 6-979. PRUSS_IEP_SYNC_FIRST_STAT Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E184h
PRU_ICSS_1_IEP	20AE E184h

Figure 6-371. PRUSS_IEP_SYNC_FIRST_STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						FIRST_SYNC1	FIRST_SYNC0
R-0h						R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 6-980. PRUSS_IEP_SYNC_FIRST_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	FIRST_SYNC1	R	0h	SYNC1 First Event status 0h: SYNC1 first event has NOT occurred 1h: SYNC1 first event has occurred. This bits is cleared when sync1_en = 0
0	FIRST_SYNC0	R	0h	SYNC0 First Event status 0h: SYNC0 first event has not occurred 1h: SYNC0 first event has occurred. This bits is cleared when sync0_en = 0

Table 6-981. Register Call Summary for PRUSS_IEP_SYNC_FIRST_STAT

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_SYNC_FIRST_STAT Register (Offset = 184h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.70 PRUSS_IEP_SYNC0_STAT Register (Offset = 188h) [reset = 0h]

PRUSS_IEP_SYNC0_STAT is shown in Figure 6-372 and described in Table 6-983.

SYNC0 STATUS

Table 6-982. PRUSS_IEP_SYNC0_STAT Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E188h
PRU_ICSS_1_IEP	20AE E188h

Figure 6-372. PRUSS_IEP_SYNC0_STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							SYNC0_PEND
R-0h							RWr1Clr-0h

LEGEND: R = Read Only; RWr1Clr = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 6-983. PRUSS_IEP_SYNC0_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	SYNC0_PEND	RWr1Clr	0h	SYNC0 pending state 0h: SYNC0 is not pending 1h: SYNC0 is pending or has occurred when SYNC0_ACK_EN = 0 (Disable). Write "1" to clear

Table 6-984. Register Call Summary for PRUSS_IEP_SYNC0_STAT

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRUSS_IEP_SYNC_CTRL Register (Offset = 180h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_SYNC0_STAT Register (Offset = 188h) [reset = 0h]: [0]

6.4.11.3.71 PRUSS_IEP_SYNC1_STAT Register (Offset = 18Ch) [reset = 0h]

PRUSS_IEP_SYNC1_STAT is shown in Figure 6-373 and described in Table 6-986.

SYNC1 STATUS

Table 6-985. PRUSS_IEP_SYNC1_STAT Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E18Ch
PRU_ICSS_1_IEP	20AE E18Ch

Figure 6-373. PRUSS_IEP_SYNC1_STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							SYNC1_PEND
R-0h							RWr1Clr-0h

LEGEND: R = Read Only; RWr1Clr = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 6-986. PRUSS_IEP_SYNC1_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	SYNC1_PEND	RWr1Clr	0h	SYNC1 pending state 0h: SYNC1 is not pending 1h: SYNC1 is pending or has occurred when SYNC1_ACK_EN = 0 (Disable). Write "1" to Clear

Table 6-987. Register Call Summary for PRUSS_IEP_SYNC1_STAT

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_SYNC_CTRL Register (Offset = 180h) [reset = 0h]: [0]
- PRUSS_IEP_SYNC1_STAT Register (Offset = 18Ch) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.72 PRUSS_IEP_SYNC_PWIDTH Register (Offset = 190h) [reset = 0h]

PRUSS_IEP_SYNC_PWIDTH is shown in Figure 6-374 and described in Table 6-989.

SYNC PULSE WIDTH CONFIGURE

Table 6-988. PRUSS_IEP_SYNC_PWIDTH Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E190h
PRU_ICSS_1_IEP	20AE E190h

Figure 6-374. PRUSS_IEP_SYNC_PWIDTH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYNC_HPW																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-989. PRUSS_IEP_SYNC_PWIDTH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SYNC_HPW	R/W	0h	Defines the number of clock cycles SYNC0/1 will be high. Note if SYNC0/1 is disabled during pulse width time (that is, SYNC_CTRL[SYNC0_EN | SYNC1_EN | SYNC_EN] = 0), the ongoing pulse will be terminated. 0h: 1 clock cycle. 1h: 2 clock cycles. Nh: N+1 clock cycles.

Table 6-990. Register Call Summary for PRUSS_IEP_SYNC_PWIDTH

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_SYNC_PWIDTH Register (Offset = 190h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]
- PRU-ICSS IEP Sync0/Sync1 Generation Modes: [0]

6.4.11.3.73 PRUSS_IEP_SYNC0_PERIOD Register (Offset = 194h) [reset = 1h]

PRUSS_IEP_SYNC0_PERIOD is shown in Figure 6-375 and described in Table 6-992.

SYNC0 PERIOD CONFIGURE

Table 6-991. PRUSS_IEP_SYNC0_PERIOD Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E194h
PRU_ICSS_1_IEP	20AE E194h

Figure 6-375. PRUSS_IEP_SYNC0_PERIOD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYNC0_PERIOD																															
R/W-1h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-992. PRUSS_IEP_SYNC0_PERIOD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SYNC0_PERIOD	R/W	1h	Defines the period between the rising edges of SYNC0. 0h: Reserved 1h: 2 clk cycles period N: N+1 clk cycles period

Table 6-993. Register Call Summary for PRUSS_IEP_SYNC0_PERIOD

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRUSS_IEP_SYNC0_PERIOD Register (Offset = 194h) [reset = 1h]: [0]
- PRU-ICSS_IEP Registers: [0]
- PRU-ICSS IEP Sync0/Sync1 Generation Modes: [0]

6.4.11.3.74 PRUSS_IEP_SYNC1_DELAY Register (Offset = 198h) [reset = 0h]

PRUSS_IEP_SYNC1_DELAY is shown in Figure 6-376 and described in Table 6-995.

SYNC1 DELAY

Table 6-994. PRUSS_IEP_SYNC1_DELAY Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E198h
PRU_ICSS_1_IEP	20AE E198h

Figure 6-376. PRUSS_IEP_SYNC1_DELAY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYNC1_DELAY																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-995. PRUSS_IEP_SYNC1_DELAY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SYNC1_DELAY	R/W	0h	When SYNC1_IND_EN = 0, defines number of clock cycles from the start of SYNC0 to the start of SYNC1. Note this is the delay before the start of SYNC1. 0h: No delay. 1h: 1 clock cycle delay. Nh: N clock cycles delay. When SYNC1_IND_EN = 1, defines the period between the rising edges of SYNC1. 0h: Reserved. 1h: 2 clock cycles period. Nh: N+1 clock cycles period.

Table 6-996. Register Call Summary for PRUSS_IEP_SYNC1_DELAY

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_SYNC1_DELAY Register (Offset = 198h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]
- PRU-ICSS IEP Sync0/Sync1 Generation Modes: [0][1]

6.4.11.3.75 PRUSS_IEP_SYNC_START Register (Offset = 19Ch) [reset = 0h]

PRUSS_IEP_SYNC_START is shown in Figure 6-377 and described in Table 6-998.

SYNC START CONFIGURE

Table 6-997. PRUSS_IEP_SYNC_START Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E19Ch
PRU_ICSS_1_IEP	20AE E19Ch

Figure 6-377. PRUSS_IEP_SYNC_START Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYNC_START																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-998. PRUSS_IEP_SYNC_START Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SYNC_START	R/W	0h	Defines the start time after the activation event. 0h: 1 clock cycle delay. Nh: N+1 clock cycles delay.

Table 6-999. Register Call Summary for PRUSS_IEP_SYNC_START

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- PRUSS_IEP_SYNC_START Register (Offset = 19Ch) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]
- PRU-ICSS IEP Sync0/Sync1 Generation Modes: [0]

6.4.11.3.76 PRUSS_IEP_WD_PREDIV Register (Offset = 200h) [reset = 4E20h]

PRUSS_IEP_WD_PREDIV is shown in Figure 6-378 and described in Table 6-1001.

WATCHDOG PRE-DIVIDER

Table 6-1000. PRUSS_IEP_WD_PREDIV Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E200h
PRU_ICSS_1_IEP	20AE E200h

Figure 6-378. PRUSS_IEP_WD_PREDIV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PRE_DIV															
R-0h																R/W-4E20h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-1001. PRUSS_IEP_WD_PREDIV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	PRE_DIV	R/W	4E20h	Defines the number of ICSS_IEP_CLK cycles per WD clock event. Note that the WD clock is a free-running clock. The value 0x4e20 (or 20000) generates a rate of 100 us if ICSS_IEP_CLK is 200 MHz. seconds/(WD event) = (clock cycles per WD event)/(clock cycles per second) = 20000/(200 x [10]^6) = 100us

Table 6-1002. Register Call Summary for PRUSS_IEP_WD_PREDIV

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_PD_WD_TIM Register (Offset = 208h) [reset = 3E8h]: [0]
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_WD_PREDIV Register (Offset = 200h) [reset = 4E20h]: [0]
- PRUSS_IEP_PDI_WD_TIM Register (Offset = 204h) [reset = 3E8h]: [0]

6.4.11.3.77 PRUSS_IEP_PDI_WD_TIM Register (Offset = 204h) [reset = 3E8h]

PRUSS_IEP_PDI_WD_TIM is shown in Figure 6-379 and described in Table 6-1004.

PDI WATCHDOG TIMER CONFIGURE

Table 6-1003. PRUSS_IEP_PDI_WD_TIM Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E204h
PRU_ICSS_1_IEP	20AE E204h

Figure 6-379. PRUSS_IEP_PDI_WD_TIM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PDI_WD_TIME															
R-0h																R/W-3E8h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-1004. PRUSS_IEP_PDI_WD_TIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	PDI_WD_TIME	R/W	3E8h	Defines the number of WD ticks (or increments) for PDI WD, that is, the number of WD increments. If PRUSS_IEP_WD_PREDIV[15-0] PRE_DIV is set to 100us, then the value 0x03e8 (or 1000) provides a rate of 100ms. Read returns the current count. Counter is reset by software write to register or when Digital Data In capture occurs. WD is disabled if WD time is set to 0x0. Note when an expiration event occurs, the expiration counter (PDI_EXP_CNT) increments and status (PDI_WD_STAT) clears.

Table 6-1005. Register Call Summary for PRUSS_IEP_PDI_WD_TIM

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_PDI_WD_TIM Register (Offset = 204h) [reset = 3E8h]: [0]

6.4.11.3.78 PRUSS_IEP_PD_WD_TIM Register (Offset = 208h) [reset = 3E8h]

PRUSS_IEP_PD_WD_TIM is shown in Figure 6-380 and described in Table 6-1007.

PD WATCHDOG TIMER CONFIGURE

Table 6-1006. PRUSS_IEP_PD_WD_TIM Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E208h
PRU_ICSS_1_IEP	20AE E208h

Figure 6-380. PRUSS_IEP_PD_WD_TIM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PD_WD_TIME															
R-0h																R/W-3E8h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-1007. PRUSS_IEP_PD_WD_TIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	PD_WD_TIME	R/W	3E8h	Defines the number of WD ticks (or increments) for PD WD, that is, the number of WD increments. If PRUSS_IEP_WD_PREDIV[15-0] PRE_DIV is set to 100us, then 0x03e8 (or 1000) provides a rate of 100ms. Read returns the current count. Counter is reset by software write to register or every write access to Sync Managers with WD trigger enable bit set. WD is disabled if WD time is set to 0x0. Expiration actions: Increment expiration counter, clear status. Digital Data out forced to zero if pr[k]_edio_oe_ext = 1 and PRUSS_IEP_DIGIO_EXP[0] SW_DATA_OUT_UPDATE = 0.

Table 6-1008. Register Call Summary for PRUSS_IEP_PD_WD_TIM

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_PD_WD_TIM Register (Offset = 208h) [reset = 3E8h]: [0]

6.4.11.3.79 PRUSS_IEP_WD_STATUS Register (Offset = 20Ch) [reset = 00010001h]

PRUSS_IEP_WD_STATUS is shown in Figure 6-381 and described in Table 6-1010.

WATCHDOG STATUS

Table 6-1009. PRUSS_IEP_WD_STATUS Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E20Ch
PRU_ICSS_1_IEP	20AE E20Ch

Figure 6-381. PRUSS_IEP_WD_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							PDI_WD_STAT
R-0h							R-1h
15	14	13	12	11	10	9	8
Galileo BCM 047:RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							PD_WD_STAT
R-0h							R-1h

LEGEND: R = Read Only; *-n* = value after reset

Table 6-1010. PRUSS_IEP_WD_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	PDI_WD_STAT	R	1h	WD PDI status. 0h: Expired (PDI_WD_EXP event generated) 1h: Active or disabled
15-1	RESERVED	R	0h	Reserved
0	PD_WD_STAT	R	1h	WD PD status (triggered by Sync Mangers status). 0h: Expired (PD_WD_EXP event generated) 1h: Active or disabled

Table 6-1011. Register Call Summary for PRUSS_IEP_WD_STATUS

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_WD_STATUS Register (Offset = 20Ch) [reset = 00010001h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.80 PRUSS_IEP_WD_EXP_CNT Register (Offset = 210h) [reset = 0h]

PRUSS_IEP_WD_EXP_CNT is shown in Figure 6-382 and described in Table 6-1013.

WATCHDOG TIMER EXPIRATION COUNTER

Table 6-1012. PRUSS_IEP_WD_EXP_CNT Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E210h
PRU_ICSS_1_IEP	20AE E210h

Figure 6-382. PRUSS_IEP_WD_EXP_CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PD_EXP_CNT								PDI_EXP_CNT							
RWrClr-0h								RWrClr-0h							

LEGEND: R = Read Only; RWrClr = Read/Cleared upon Write; *-n* = value after reset

Table 6-1013. PRUSS_IEP_WD_EXP_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	PD_EXP_CNT	RWrClr	0h	WD PD expiration counter. Counter increments on every PD time out and stops at FFh.
7-0	PDI_EXP_CNT	RWrClr	0h	WD PDI expiration counter. Counter increments on every PDI time out and stops at FFh.

Table 6-1014. Register Call Summary for PRUSS_IEP_WD_EXP_CNT

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_WD_EXP_CNT Register (Offset = 210h) [reset = 0h]: [0]

6.4.11.3.81 PRUSS_IEP_WD_CTRL Register (Offset = 214h) [reset = 0h]

PRUSS_IEP_WD_CTRL is shown in Figure 6-383 and described in Table 6-1016.

WATCHDOG CONTROL

Table 6-1015. PRUSS_IEP_WD_CTRL Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E214h
PRU_ICSS_1_IEP	20AE E214h

Figure 6-383. PRUSS_IEP_WD_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							PDI_WD_EN
R-0h							R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							PD_WD_EN
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-1016. PRUSS_IEP_WD_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	PDI_WD_EN	R/W	0h	Watchdog PDI 0h: Disable 1h: Enable
15-1	RESERVED	R	0h	Reserved
0	PD_WD_EN	R/W	0h	Watchdog PD 0h: Disable 1h: Enable

Table 6-1017. Register Call Summary for PRUSS_IEP_WD_CTRL

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_WD_CTRL Register (Offset = 214h) [reset = 0h]: [0]

6.4.11.3.82 PRUSS_IEP_DIGIO_CTRL Register (Offset = 300h) [reset = 4h]

PRUSS_IEP_DIGIO_CTRL is shown in Figure 6-384 and described in Table 6-1019.

DIGITAL INPUT OUTPUT CONTROL

Table 6-1018. PRUSS_IEP_DIGIO_CTRL Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E300h
PRU_ICSS_1_IEP	20AE E300h

Figure 6-384. PRUSS_IEP_DIGIO_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
OUT_MODE		IN_MODE		WD_MODE	BIDI_MODE	OUTVALID_MODE	OUTVALID_POL
R/W-0h		R/W-0h		R/W-0h	R-1h	R/W-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-1019. PRUSS_IEP_DIGIO_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	OUT_MODE	R/W	0h	Defines events that triggers data out to be updated. Note if OUTVALID_MODE is set, then data out is forced to zero if a WD PD expiration occurs (PD_WD_EXP) from the WD block and pr<k>_edio_oe_ext = 1. 0h: PRU0/1_RX_EOF 1h: Reserved 2h: DC SYNC0 event 3h: DC SYNC1 event
5-4	IN_MODE	R/W	0h	Defines event that triggers data in to be sampled 0h: PRU0/1_RX_SOF 1h: Rising edge of external PR<k>_EDC_LATCH0_IN signal 2h: DC rising edge of SYNC0 event 3h: DC rising edge of SYNC1 event
3	WD_MODE	R/W	0h	Defines Watchdog behavior 0h: Outputs are reset immediately after watchdog expires 1h: Outputs are reset with next output event that follows watchdog expiration

Table 6-1019. PRUSS_IEP_DIGIO_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	BIDI_MODE	R	1h	Defines the digital input/output direction. NOTE THAT DUE TO INTEGRATION, ACTUAL MODE IS UNIDIRECTIONAL IN THIS DEVICE. 0h: Unidirectional mode: digital input/output direction of pins configured individually 1h: Bidirectional mode: all I/O pins are bidirectional and direction configuration is ignored
1	OUTVALID_MODE	R/W	0h	Defines the outvalid mode behavior. 0h: Output event signaling 1h: Output data is updated if watchdog is triggered. Output data is forced to zero if PD_WD_EXP from the WD block and pr1_edio_oe_ext = 1
0	OUTVALID_POL	R	0h	Defines OUTVALID polarity 0h: Active High 1h: Active Low

Table 6-1020. Register Call Summary for PRUSS_IEP_DIGIO_CTRL

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- Basic Programming Model: [0]
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_DIGIO_DATA_IN Register (Offset = 308h) [reset = -h]: [0]
- PRU-ICSS Industrial Ethernet Digital IOs: [0]
- PRUSS_IEP_DIGIO_DATA_OUT Register (Offset = 310h) [reset = 0h]: [0]
- DIGIO Block Diagrams: [0]
- PRUSS_IEP_DIGIO_CTRL Register (Offset = 300h) [reset = 4h]: [0]

6.4.11.3.83 PRUSS_IEP_DIGIO_STATUS Register (Offset = 304h) [reset = 0h]

PRUSS_IEP_DIGIO_STATUS is shown in Figure 6-385 and described in Table 6-1022.

DIGITAL INPUT OUTPUT STATUS

Table 6-1021. PRUSS_IEP_DIGIO_STATUS Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E304h
PRU_ICSS_1_IEP	20AE E304h

Figure 6-385. PRUSS_IEP_DIGIO_STATUS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIGIO_STAT																															
R-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-1022. PRUSS_IEP_DIGIO_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DIGIO_STAT	R	0h	Reserved

Table 6-1023. Register Call Summary for PRUSS_IEP_DIGIO_STATUS

PRU-ICSS Industrial Ethernet Peripheral (IEP)
- PRUSS_IEP_DIGIO_STATUS Register (Offset = 304h) [reset = 0h]: [0]
- PRU-ICSS_IEP Registers: [0]

6.4.11.3.84 PRUSS_IEP_DIGIO_DATA_IN Register (Offset = 308h) [reset = -h]

PRUSS_IEP_DIGIO_DATA_IN is shown in Figure 6-386 and described in Table 6-1025.

DIGITAL DATA INPUT

Table 6-1024. PRUSS_IEP_DIGIO_DATA_IN Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E308h
PRU_ICSS_1_IEP	20AE E308h

Figure 6-386. PRUSS_IEP_DIGIO_DATA_IN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_IN																															
R--h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-1025. PRUSS_IEP_DIGIO_DATA_IN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA_IN	R	-h	Data input. Digital inputs can be configured to be sampled in four ways. 1h: Digital inputs are sampled at the start of each frame. The SOF signal can be used externally to update the input data, because the SOF is signaled before input data is sampled. 2h: The sample time can be controlled externally by using them PR<k>_EDC_LATCH0_IN signal. 3h: Digital inputs are sampled at SYNC0 events. 4h: Digital inputs are sampled at SYNC1 events. These can be configured by [5-4] IN_MODE bit field in the PRUSS_IEP_DIGIO_CTRL register.

Table 6-1026. Register Call Summary for PRUSS_IEP_DIGIO_DATA_IN

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- Basic Programming Model: [0]
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_DIGIO_DATA_IN Register (Offset = 308h) [reset = -h]: [0]

6.4.11.3.85 PRUSS_IEP_DIGIO_DATA_IN_RAW Register (Offset = 30Ch) [reset = -h]

PRUSS_IEP_DIGIO_DATA_IN_RAW is shown in Figure 6-387 and described in Table 6-1028.

DIGITAL DATA INPUT DIRECT SAMPLE

Table 6-1027. PRUSS_IEP_DIGIO_DATA_IN_RAW Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E30Ch
PRU_ICSS_1_IEP	20AE E30Ch

Figure 6-387. PRUSS_IEP_DIGIO_DATA_IN_RAW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_IN_RAW																															
R--h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 6-1028. PRUSS_IEP_DIGIO_DATA_IN_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA_IN_RAW	R	-h	Data input which direct sample of PR<k>_EDIO_DATA[0:31]. Only PR<k>_EDIO_DATA[0:3] are exported to device pins in this device.

Table 6-1029. Register Call Summary for PRUSS_IEP_DIGIO_DATA_IN_RAW

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- Basic Programming Model: [0]
- PRUSS_IEP_DIGIO_DATA_IN_RAW Register (Offset = 30Ch) [reset = -h]: [0]
- PRU-ICSS_IEP Registers: [0]
- Features: [0]

6.4.11.3.86 PRUSS_IEP_DIGIO_DATA_OUT Register (Offset = 310h) [reset = 0h]

PRUSS_IEP_DIGIO_DATA_OUT is shown in Figure 6-388 and described in Table 6-1031.

DIGITAL DATA OUTPUT

Table 6-1030. PRUSS_IEP_DIGIO_DATA_OUT Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E310h
PRU_ICSS_1_IEP	20AE E310h

Figure 6-388. PRUSS_IEP_DIGIO_DATA_OUT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_OUT																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-1031. PRUSS_IEP_DIGIO_DATA_OUT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA_OUT	R/W	0h	Data output. Digital outputs can be configured to be updated in four ways. 1h: Digital outputs are updated at the end of each frame (EOF mode). 2h: Digital outputs are updated with SYNC0 events 3h: Digital outputs are updated SYNC1 events. 4h: Digital outputs are updated at the end of a frame which triggered the Process Data Watchdog. Digital Outputs are only updated if the frame was correct (WD_TRIG mode). These can be configured by [7-6] OUT_MODE bit field in the PRUSS_IEP_DIGIO_CTRL.

Table 6-1032. Register Call Summary for PRUSS_IEP_DIGIO_DATA_OUT

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- Basic Programming Model: [0][1]
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_DIGIO_DATA_OUT Register (Offset = 310h) [reset = 0h]: [0]

6.4.11.3.87 PRUSS_IEP_DIGIO_DATA_OUT_EN Register (Offset = 314h) [reset = 0h]

PRUSS_IEP_DIGIO_DATA_OUT_EN is shown in Figure 6-389 and described in Table 6-1034.

DIGITAL DATA OUT ENABLE

Table 6-1033. PRUSS_IEP_DIGIO_DATA_OUT_EN Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E314h
PRU_ICSS_1_IEP	20AE E314h

Figure 6-389. PRUSS_IEP_DIGIO_DATA_OUT_EN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_OUT_EN																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 6-1034. PRUSS_IEP_DIGIO_DATA_OUT_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA_OUT_EN	R/W	0h	Data input which controls tri-state of PR<k>_EDIO_DATA[0:3] 0h: Driver enabled. 1h: Sets outputs to HiZ.

Table 6-1035. Register Call Summary for PRUSS_IEP_DIGIO_DATA_OUT_EN

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- Basic Programming Model: [0]
- PRU-ICSS_IEP Registers: [0]
- PRUSS_IEP_DIGIO_DATA_OUT_EN Register (Offset = 314h) [reset = 0h]: [0]

6.4.11.3.88 PRUSS_IEP_DIGIO_EXP Register (Offset = 318h) [reset = 20h]

PRUSS_IEP_DIGIO_EXP is shown in Figure 6-390 and described in Table 6-1037.

DIGIO EXPANSION REGISTER

Table 6-1036. PRUSS_IEP_DIGIO_EXP Instances

Instance	Physical Address
PRU_ICSS_0_IEP	20AA E318h
PRU_ICSS_1_IEP	20AE E318h

Figure 6-390. PRUSS_IEP_DIGIO_EXP Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		EOF_SEL	SOF_SEL	SOF_DLY			
R-0h		R/W-0h	R/W-0h	R/W-0h			
7	6	5	4	3	2	1	0
OUTVALID_DLY				RESERVED	SW_OUTVALID	OUTVALID_OVR_EN	SW_DATA_OUT_UPDATE
R/W-2h				R-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 6-1037. PRUSS_IEP_DIGIO_EXP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13	EOF_SEL	R/W	0h	Defines which RX_EOF is used for PR<k>_EDIO_DATA_IN[0:3] capture 0h: PRU0_RX_EOF 1h: PRU1_RX_EOF
12	SOF_SEL	R/W	0h	Defines which RX_SOF is used for PR<k>_EDIO_DATA_IN[0:3] capture 0h: PRU0_RX_SOF 1h: PRU1_RX_SOF
11-8	SOF_DLY	R/W	0h	Define the number of iep_clk (ICSS_IEP_CLK) cycle delay of SOF PR<k>_EDIO_DATA_IN[0:3] capture
7-4	OUTVALID_DLY	R/W	2h	Define the number of iep_clk (ICSS_IEP_CLK) cycle delay on assertion of PR<k>_EDIO_OUTVALID. Min is 2 clock cycles. Max is 16 clock cycles
3	RESERVED	R	0h	Reserved
2	SW_OUTVALID	R/W	0h	PR<k>_EDIO_OUTVALID = SW_OUTVALID, only if OUTVALID_OVR_EN is set.
1	OUTVALID_OVR_EN	R/W	0h	Software override enable 0h: Disable override 1h: Enable override

Table 6-1037. PRUSS_IEP_DIGIO_EXP Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	SW_DATA_OUT_UPDATE	R/W	0h	Defines the value of pr<k>_edio_data_out when OUTVALID_OVR_EN = 1. Read 0: Start bit event has not occurred Read 1: Start bit event occurred Write 0: No effect Write 1: Causes an update of pr<k>_edio_data_out by software data out

Table 6-1038. Register Call Summary for PRUSS_IEP_DIGIO_EXP

PRU-ICSS Industrial Ethernet Peripheral (IEP)

- Basic Programming Model: [0][1]
- PRUSS_IEP_DIGIO_EXP Register (Offset = 318h) [reset = 20h]: [0]
- PRU-ICSS_IEP Registers: [0]
- DIGIO Block Diagrams: [0]
- PRUSS_IEP_PD_WD_TIM Register (Offset = 208h) [reset = 3E8h]: [0]

Memory Subsystem

This chapter describes the Memory Subsystem for the device.

Topic **Page**

7.1 Multicore Shared Memory Controller (MSMC)

This section describes the Multicore Shared Memory Controller (MSMC) for the device.

7.1.1 MSMC Overview

The Multicore Shared Memory Controller (MSMC) manages traffic among the device ARMSS, DSP, DMA, other master peripherals, and the EMIF controller. It also provides a shared on-chip SRAM that is accessible by the ARMSS, DSP and the master peripherals in the device.

The MSMC module has the following features:

- CPU/1 frequency of operation (that is, frequency same as that of the ARMSS/DSP)
- One 256-bit master interface for connection to external SDRAM (through EMIF controller)
- One 256-bit master interface for connection to TeraNet_DMA
- One 256-bit slave interface for the DSP
- One 256-bit slave interface for the ARMSS
- One 256-bit slave interface for accesses to the shared SRAM
- One 256-bit slave interface for accesses to the external SDRAM
- Memory protection for accesses to both the shared SRAM and external SDRAM spaces
- Address extension from 32-bit to 36-bit for larger addressing space
- Error Detection and Correction (EDC) and scrubbing support for the MSMC SRAM
- Level 2 or Level 3 shared SRAM that is accessible by the device ARMSS, DSP and the master peripherals
- Coherency between ARMSS L1/L2 cache and EDMA/system master peripherals (through SES/SMS ports) in the SRAM space and SDRAM space

7.1.2 MSMC Integration

This section describes the module integration in the device including information about clocks, resets, and hardware requests.

Figure 7-1 shows the integration of the MSMC module in the device.

Figure 7-1. MSMC Integration

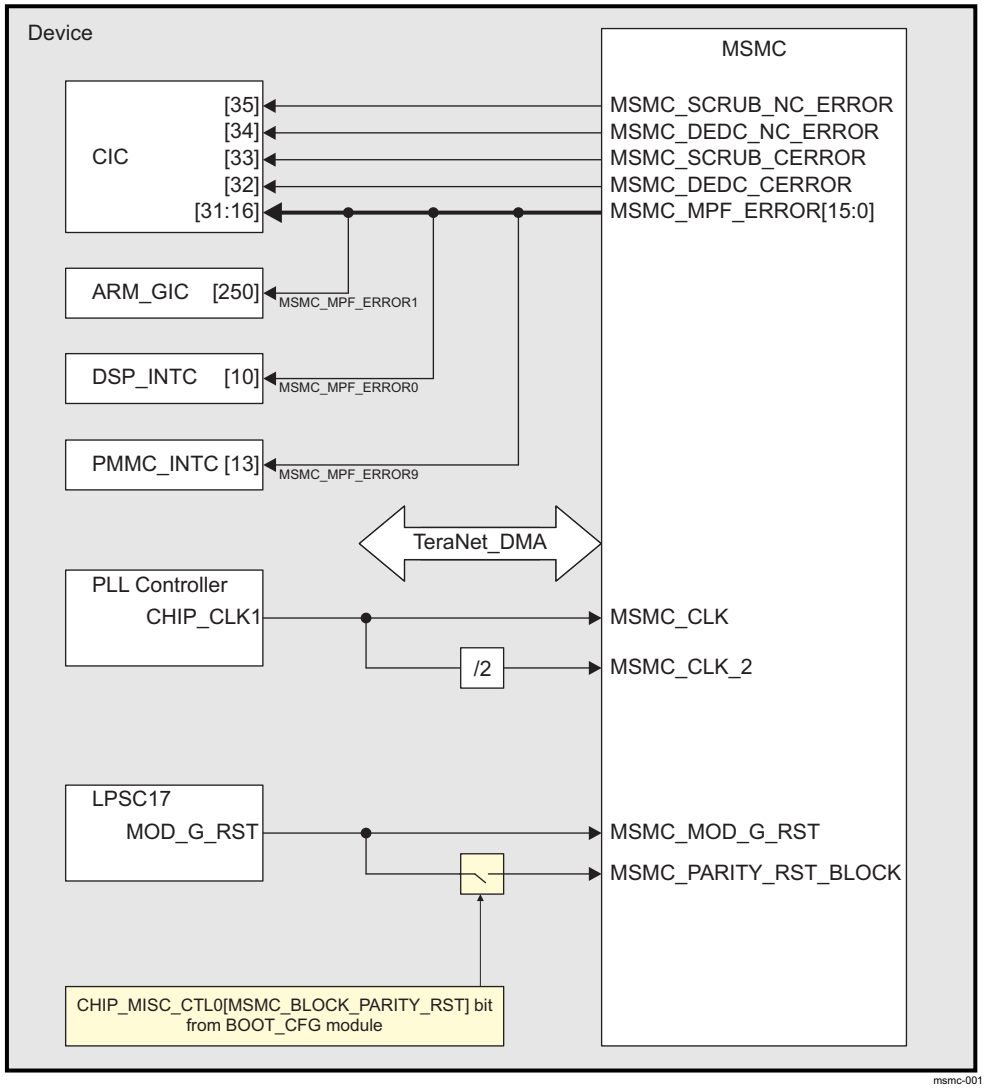


Table 7-1 through Table 7-3 summarize the integration of the MSMC module in the device.

Table 7-1. MSMC Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
MSMC	PD7	LPSC17	TeraNet_DMA

Table 7-2. MSMC Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
MSMC	MSMC_CLK	CHIP_CLK1	PLL Controller	Clocks to the MSMC module and its associated SRAM. The SRAM is clocked by MSMC_CLK.
	MSMC_CLK_2	CHIP_CLK1/2	PLL Controller	
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
MSMC	MSMC_PARITY_RST_BLOCK	MOD_G_RST	LPSC17	Parity RAM reset. This reset is controlled by the CHIP_MISC_CTL0[12] MSMC_BLOCK_PARITY_RST bit which resides in the BOOT_CFG module.
	MSMC_MOD_G_RST	MOD_G_RST	LPSC17	MSMC module global reset

Table 7-3. MSMC Hardware Requests

Interrupt Requests						
Module Instance	**Event Name**	**Mapped To Input Event [Number]**				**Description**
		ARM GIC	**CIC**	**DSP INTC**	**PMMC INTC**	
MSMC	MSMC_MPF_ERROR0	-	[16]	[10]	-	Memory protection fault interrupt for master with PrivID = 0
	MSMC_MPF_ERROR1	[250]	[17]	-	-	Memory protection fault interrupt for master with PrivID = 1
	MSMC_MPF_ERROR2	-	[18]	-	-	Memory protection fault interrupt for master with PrivID = 2
	MSMC_MPF_ERROR3	-	[19]	-	-	Memory protection fault interrupt for master with PrivID = 3
	MSMC_MPF_ERROR4	-	[20]	-	-	Memory protection fault interrupt for master with PrivID = 4
	MSMC_MPF_ERROR5	-	[21]	-	-	Memory protection fault interrupt for master with PrivID = 5
	MSMC_MPF_ERROR6	-	[22]	-	-	Memory protection fault interrupt for master with PrivID = 6
	MSMC_MPF_ERROR7	-	[23]	-	-	Memory protection fault interrupt for master with PrivID = 7
	MSMC_MPF_ERROR8	-	[24]	-	-	Memory protection fault interrupt for master with PrivID = 8
	MSMC_MPF_ERROR9	-	[25]	-	[13]	Memory protection fault interrupt for master with PrivID = 9
	MSMC_MPF_ERROR10	-	[26]	-	-	Memory protection fault interrupt for master with PrivID = 10
	MSMC_MPF_ERROR11	-	[27]	-	-	Memory protection fault interrupt for master with PrivID = 11
	MSMC_MPF_ERROR12	-	[28]	-	-	Memory protection fault interrupt for master with PrivID = 12
	MSMC_MPF_ERROR13	-	[29]	-	-	Memory protection fault interrupt for master with PrivID = 13
	MSMC_MPF_ERROR14	-	[30]	-	-	Memory protection fault interrupt for master with PrivID = 14
	MSMC_MPF_ERROR15	-	[31]	-	-	Memory protection fault interrupt for master with PrivID = 15
	MSMC_DEDC_CERROR	-	[32]	-	-	Correctable (1-bit) soft error detected on SRAM read
	MSMC_SCRUB_CERROR	-	[33]	-	-	Correctable (1-bit) soft error detected during scrub cycle
	MSMC_DEDC_NC_ERROR	-	[34]	-	-	Non-correctable (2-bit) soft error detected on SRAM read
	MSMC_SCRUB_NC_ERROR	-	[35]	-	-	Non-correctable (2-bit) soft error detected during scrub cycle

NOTE: For more information about the MSMC interrupts, see Section 7.1.3.5.

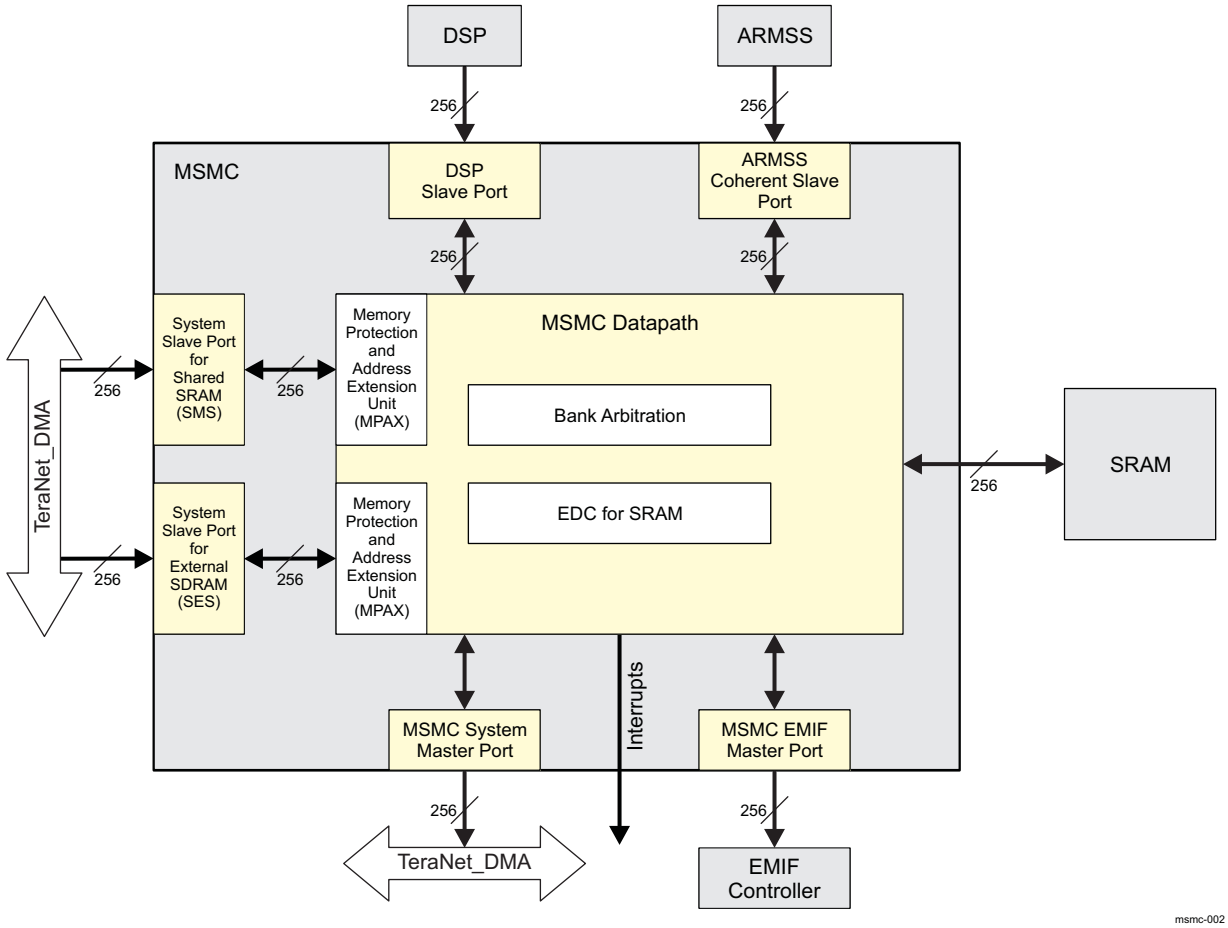
For more information about the MSMC resets, see Section 7.1.3.7.

7.1.3 MSMC Functional Description

7.1.3.1 MSMC Block Diagram

Figure 7-2 shows a high-level view of the MSMC module that includes the main interfaces, memory, and subunits.

Figure 7-2. MSMC Functional Block Diagram



The MSMC has one slave interface to connect to the DSP, one slave interface to connect to the ARMSS, two slave interfaces to connect to the system interconnect (TeraNet), one master port to connect to the EMIF controller, and one master port to connect to the system interconnect (TeraNet).

NOTE: Although the SES and SMS ports are 256 bits wide the maximum bandwidth of a master accessing MSMC through these ports is limited to 128 bits * CHIP_CLK1/3 because SES and SMS are slaves on TeraNet_M_3_128_0 as shown in Figure 3-2. TeraNet_M_3_128_0 is 128 bits wide and runs at CHIP_CLK1/3 frequency. This bandwidth limitation does not apply to ARMSS and DSP accesses as these masters have dedicated MSMC slave ports.

NOTE: The maximum bandwidth of DSP-to-MSMC-SRAM accesses is 256 bits * CHIP_CLK1 as DSP is directly connected to MSMC. The maximum bandwidth of ARMSS-to-MSMC-SRAM accesses is limited to 128 bits * CHIP_CLK1 although MSMC offers 256-bit bus width. Concurrent accesses by different masters to the MSMC SRAM are prioritized and time-sliced as there is a single SRAM bank. For SDRAM accesses the 256-bit wide EMIF interface of MSMC runs at CHIP_CLK1/2 frequency.

7.1.3.1.1 DSP and ARMSS Slave Interfaces

The MSMC has slave interfaces to connect to the ACE (AXI Coherence Extensions) port of the ARMSS and MDMA port of the DSP. The ARMSS and DSP use these interfaces to access the MSMC on-chip SRAM, the external SDRAM and the EMIF configuration registers through the MSMC EMIF Master Port or system level resources through the MSMC System Master Port.

7.1.3.1.2 System Slave Interfaces

The MSMC has two slave interfaces to handle accesses from the system masters to the MSMC SRAM and the external SDRAM.

7.1.3.1.2.1 System EMIF Access Slave Interface (SES)

The SES interface handles accesses to the external SDRAM and the configuration registers inside the EMIF module that originate from a system master different than ARMSS and DSP.

Accesses presented on this interface to any addresses outside of the address range mapped to the external memory or the EMIF configuration registers result in an addressing error returned to the requesting master.

NOTE: When MSMC SRAM is remapped to an external address space using MPAX, such accesses will not result in an addressing error as long as the accesses are within the valid external memory address range.

The address width on this interface is 32 bits. Address extension to a 36-bit address is done inside the MSMC as described in Section 7.1.3.3.

7.1.3.1.2.2 System MSMC SRAM Access Slave Interface (SMS)

The SMS interface handles accesses to MSMC SRAM that originate from a system master different than ARMSS and DSP. Accesses from system masters to the MSMC configuration registers are also expected to be presented on this interface.

Any accesses from the SMS interface that do not address the MSMC SRAM or configuration registers result in an addressing error returned to the requesting master.

7.1.3.1.3 System Master Interface

The MSMC has one master interface for the DSP and ARMSS to access system resources other than:

- MSMC SRAM
- MSMC configuration regiters
- EMIF SDRAM
- EMIF configuration regiters

Traffic from the system slave interfaces does not pass through the system master interface.

7.1.3.1.4 EMIF Master Interface

The EMIF module is connected to the MSMC through the EMIF master interface. The address width for this interface is 36 bits because it supports extended memory addressing space beyond 4 GB. The MSMC implements an address extension to 36 bits as described in Section 7.1.3.3.

7.1.3.2 MSMC Memory

7.1.3.2.1 MSMC SRAM

MSMC SRAM can serve as a Shared Level 2 or Level 3 memory

- Shared Level 2 memory — The MSMC memory is cacheable by DSP L1D and L1P caches. DSP L2 will not cache requests to the MSMC SRAM.

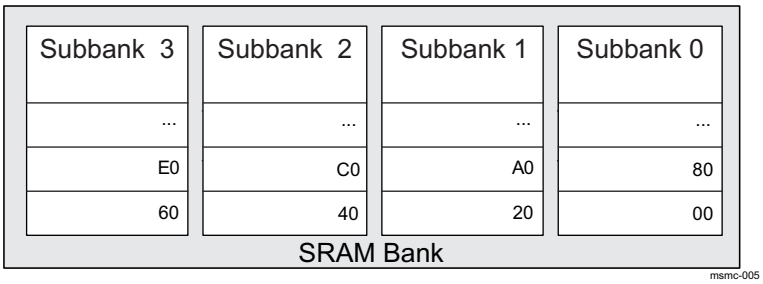
- Shared Level 3 memory — The MSMC memory is not directly cacheable at the DSP L2 but is cacheable in DSP L1D and L1P. However, if it is remapped to an external address using the address-extension capabilities in the DSP MPAX, the MSMC memory can be cached as a shared L3 memory both in the L1 and L2 caches of the DSP. To achieve this, the caching must be enabled in MAR registers (using MAR[PC] bit) for the remapped region. The MSMC memory is directly cacheable in ARMSS L2 memory by defining the MSMC SRAM region as normal cacheable memory in ARMSS MMU.

7.1.3.2.2 MSMC Memory Organisation

The memory is organized as one 128-byte wide bank that consists internally of four subbanks which hold adjacently-addressed locations. 32-byte aligned addresses within that bank are located in different subbanks. The 128-byte banking structure aligns with the DSP L2 cache line size.

Figure 7-3 shows the MSMC memory organization.

Figure 7-3. MSMC Memory Organisation



7.1.3.2.3 MSMC Bandwidth Management

The MSMC has an arbitration scheme which fairly allocates accesses of requestors with same priority level. This arbitration scheme does not have software controls. However, it is not sufficient to only ensure boundary for the wait times experienced by low priority requests. As a result requestors could starve for access when there is heavy traffic at high priority levels. To avoid indefinite starvation for low priority requests, the MSMC has a bandwidth management scheme that limits starvation times using the so called starvation bound registers. The MSMC has a starvation bound register (SBND) per requestor. The registers are programmed with a desired starvation boundary in MSMC cycles for the requestor's accesses. These registers are shown in Table 7-4.

The central arbiter for the memory bank and the EMIF master port contain a starvation counter for each of the requestors being tracked. Whenever a SCNT bitfield of a SBND register is programmed, the counter for the requestor is initialized with the same value.

Table 7-4. Starvation Counters per Requestor

Bit Field	Function
MSMC_SBNDC0[23-16] SCNTCE	Reload value (pre-scaled by 16) for the starvation counter for DSP requests at the EMIF arbiter.
MSMC_SBNDE[23-16] SCNTEE	Reload value (pre-scaled by 16) for the starvation counter for SES requests at the EMIF arbiter.
MSMC_SBNDC0[7-0] SCNTCM	Reload value for the starvation counters for DSP requests at the RAM bank arbiter.
MSMC_SBNDM[7-0] SCNTMM	Reload value for the starvation counters for SMS requests at the RAM bank arbiter.
MSMC_SBNDE[7-0] SCNTEM	Reload value for the starvation counters for SES requests at the RAM bank arbiter.

For the EMIF arbiter, if the interface to the EMIF is stalled, the SCNT countdown is suspended for that cycle.

When the SCNT reaches zero, the priority of the request is elevated to zero (the highest priority level). After the elevated priority is serviced, the starvation counter is reloaded from the SBND register for the requestor. Further accesses from the requestor are based on the original priority.

7.1.3.3 Memory Protection and Address Extension (MPAX)

Through the MPAX feature the MSMC module can support an external memory space addressable with a 36 bits even though the device addressing remains 32 bits. The DSP uses its own MPAX units to extend 32-bit addresses to 36-bit addresses before presenting them to the MSMC module. The ARMSS can optionally use the MMU with LPAE (Large Physical Address Extension) to support 40-bit physical addressing but the 4 MSBs of the physical address should be set to 0h in the ARMSS MMU.

The slave interfaces on the MSMC that receive addresses from all other masters in the system must extend the address inside the MSMC. These interfaces also provide support for memory protection for accesses from system masters to the MSMC SRAM, external SDRAM, and the EMIF configuration registers.

Both system slave interfaces (SES and SMS) have an MPAX unit which combines these functions and is also similar to the MPAX unit inside the DSP.

7.1.3.3.1 MPAX Segment Operation

The MPAX process is performed for a variable-sized segment of memory and is controlled by a register pair for each segment: MPAXH and MPAXL control registers.

- The MPAXH specifies the base address and size of the segment to match.
- The MPAXL specifies the replacement address and permissions for the segment.

Each MPAX unit provides eight control register pairs per PrivID of the system masters which allows operating on eight independent and potentially overlapping variable-size memory segments. For the PrivID values assigned to various system masters see Table 3-16 in Section 3.2, *Memory Protection Units*.

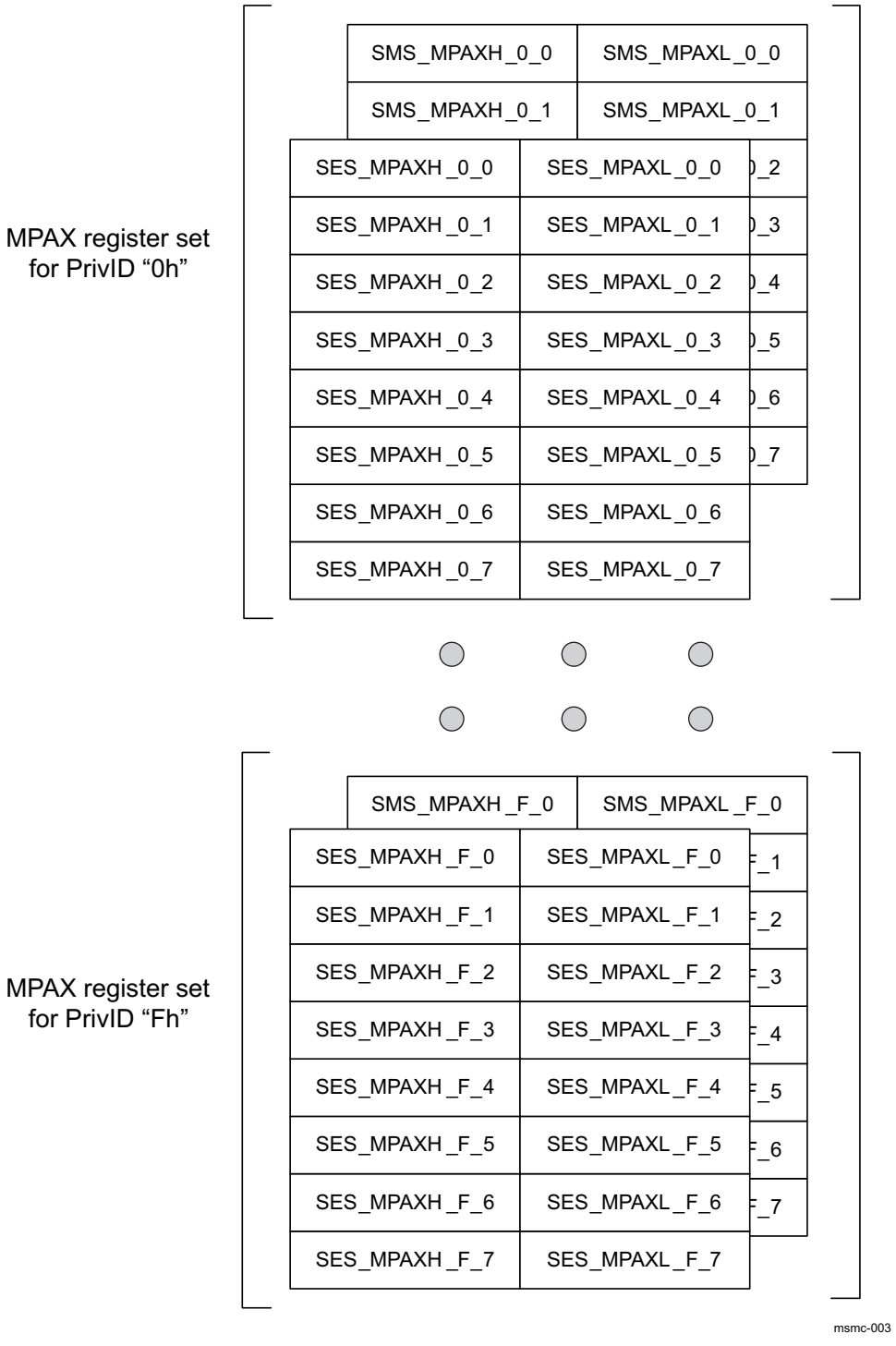
The MSMC configuration registers are not protected by the MPAX hardware. Addresses presented for any system accesses to the MSMC control registers through the SMS port are used as such.

Figure 7-4 shows how the MPAX segment registers are organized. The MPAX segment registers are described in Section 7.1.4.7.

All of the MPAX registers in the MSMC can be read by the ARMSS and DSP and also by system masters through the SMS port. Write-access control to these registers can be coordinated with the aid of semaphores external to the MSMC as well as through the locking mechanism of MSMC configuration registers.

Accesses of system masters to the segment registers are automatically protected as these registers are addressed by PrivID such that each PrivID has access to only its set of eight pairs of segment registers. Accesses to MPAX segment registers with a mismatched PrivID generates a protection error.

Figure 7-4. MPAX Segment Register Set Layout



7.1.3.3.2 MPAX Segment Register Reset Values

At reset, the MPAX segment 0 register pair has initial values that set up unrestricted access to the full MSMC SRAM address space and 2 GB of the EMIF address space. All other segments come up with the permission bits and size set to 0 (that is, no access map to those segments).

- The SMS_MPAXH_n_0 registers have reset value of 0C00 0017h and the SMS_MPAXL_n_0 registers (where n = PrivID) have reset value of 00C0 00BFh which means that segment 0 is sized to 16 MB

and matches any accesses to the address range 0CXX XXXXh.

- The SES_MPAXH_n_0 registers have reset value of 8000 001Eh and the SES_MPAXL_n_0 registers (where n = PrivID) have reset value of 8000 00BFh which means that segment 0 is sized to 2 GB and matches any accesses to the address range 8XXX XXXXh. This 2 GB space starts at the external memory base address of 8000 0000h.
- The SMS_MPAXH and SMS_MPAXL registers for segments 1 through 7 come out of reset as 0C00 0000h and 00C0 0080h respectively. The SES_MPAXH and SES_MPAXL registers for segments 1 through 7 come out of reset as 0000 0000h and 0000 0080h respectively.

These reset settings are provided for convenience of code setting-up. It is recommended that boot code sets up these registers as appropriate for the application.

7.1.3.3.3 Memory Protection

For the SES and SMS, the MPAXH register contains the Segment Base Address (BADDR) field that is matched to the incoming address on the SES port to identify the addressed segment. The BADDR and the Segment Size (SEGSZ) field describe the placement and size of the controlled segment. SEGSZ is a five-bit field that can be used to specify segment sizes ranging from 4 KB to 4 GB in powers of two, as listed in Table 7-5.

Table 7-5. MPAX Segment Size Encoding

SEGSZ	Meaning	SEGSZ	Meaning	SEGSZ	Meaning	SEGSZ	Meaning
0h	Segment disabled	8h	Reserved	10h	128KB	18h	32MB
1h	Reserved	9h	Reserved	11h	256KB	19h	64MB
2h	Reserved	Ah	Reserved	12h	512KB	1Ah	128MB
3h	Reserved	Bh	4KB	13h	1MB	1Bh	256MB
4h	Reserved	Ch	8KB	14h	2MB	1Ch	512MB
5h	Reserved	Dh	16KB	15h	4MB	1Dh	1GB
6h	Reserved	Eh	32KB	16h	8MB	1Eh	2GB
7h	Reserved	Fh	64KB	17h	16MB	1Fh	4GB

Segments are always sized to a power of two and start on a corresponding power-of-two boundary (for example, a 4 KB segment always starts on a 4 K boundary, and a 4 GB segment corresponds to the entire 32-bit address space). For the MPAX unit in the SMS port, the maximum allowed segment size is 16 MB (SEGSZ = 17h), which would cover the full MSMC storage space.

For the MPAX unit in the SES port, a sufficient number of upper bits (depending on the size specified in the SEGSZ field) of the incoming address are matched against the BADDR field of all the MPAXH registers corresponding to the access PrivID to select the MPAXH control register pair to use for memory protection and extension. For example, for 4 KB segments, all 20 bits of the BADDR field must match the upper 20 bits of the system address. For 16 MB segments, the upper eight bits of the BADDR field must match the upper eight bits of the DSP address. The remaining bits are ignored. For 4 GB segments, no BADDR bits are consulted and all addresses match.

- If an address does not match any of the programmed MPAXH registers and is not a MSMC configuration register address, the access permissions for the access are considered to be 0. This results in a protection fault.
- If an address matches multiple MPAXH registers (overlapping segment descriptors), the highest numbered MPAX register-pair is selected (that is, if the address matches BADDR in MPAXH5 and in MPAXH2, MPAXH5 is selected). Using this priority-based matching behavior, multiple MPAX entries can be programmed for overlapping segments to achieve either non-power-of-two sized segments or subsegments with different memory protection (and/or extension) parameters.

7.1.3.3.3.1 Memory Protection Fault Reporting

The MSMC memory protection fault reporting registers are listed in Table 7-6 and described in Section 7.1.4.3.

Table 7-6. MSMC Protection Fault Reporting Register List

Acronym	Register Description
MSMC_SMPFAR	Shared Memory Protection Fault Address Register
MSMC_SMPFXR	Shared Memory Protection Fault Extension Register
MSMC_SMPFR	Shared Memory Protection Fault Requestor Register
MSMC_SMPFCR	Shared Memory Protection Fault Control Register

The MPAXL register contains the permission attributes for supervisor and user accesses. If an access mismatches MPAXL with the corresponding permission attributes, a memory fault is triggered that results in the following actions:

- A memory protection fault interrupt (MSMC_MPF_ERROR**n**, where **n** = PrivID of the requestor making the offending access) is generated on the fault access, if the corresponding interrupt is enabled in the MSMC_SMIESTAT register. This is also recorded in the MSMC_SMIRSTAT register (see Section 7.1.3.5) that contains one event bit per PrivID that is set when an access from the PrivID faults and is cleared on writing 1h to the associated bit in the MSMC_SMIRC register. Only one fault is notified per PrivID. Until the interrupt status is cleared in MSMC_SMIRC, the next interrupt for that PrivID is not generated.
- The access address causing the fault is captured in the MSMC_SMPFAR register. If the fault is the result of the address not matching any of the Segment Base Address, the MSMC_SMPFXR[0] NM bit is set. The master ID and PrivID associated with the access are recorded in the MSMC_SMPFR register and a bus-protection-error status is returned to the requesting master for the fault access. For the Master ID and PrivID values assigned to various system masters see Section 3.2, *Memory Protection Units*.
- There is only one fault recorded in the MSMC_SMPFAR and MSMC_SMPFXR registers. The master is notified with a fault interrupt and receives a bus error for the access. In response, the master must clear the recorded fault by writing 1h to the MSMC_SMPFCR[0] CLR bit before further faults can be recorded.

7.1.3.3.4 Address Extension

For access to the SES interface, the 32-bit address is extended to 36 bits by replacing the appropriate number of upper bits of the address (based on segment size) with a corresponding portion of the replacement address field (RADDR) in the MPAXL register that has four more upper bits, as shown in Table 7-7.

Table 7-7. Replacement Address Used as Per-Segment Size

SEGSZ	SES / SMS RADDR bits	SEGSZ	SES RADDR bits	SMS RADDR bits	SEGSZ	SES RADDR bits	SMS RADDR bits	SEGSZ	SES RADDR bits
0h	Segment disabled	8h	Reserved	Reserved	10h	[23-5]	[19-5]	18h	[23-13]
1h	Reserved	9h	Reserved	Reserved	11h	[23-6]	[19-6]	19h	[23-14]
2h	Reserved	Ah	Reserved	Reserved	12h	[23-7]	[19-7]	1Ah	[23-15]
3h	Reserved	Bh	[23-0]	[19-0]	13h	[23-8]	[19-8]	1Bh	[23-16]
4h	Reserved	Ch	[23-1]	[19-1]	14h	[23-9]	[19-9]	1Ch	[23-17]
5h	Reserved	Dh	[23-2]	[19-2]	15h	[23-10]	[19-10]	1Dh	[23-18]
6h	Reserved	Eh	[23-3]	[19-3]	16h	[23-11]	[19-11]	1Eh	[23-19]
7h	Reserved	Fh	[23-4]	[19-4]	17h	[23-12]	[19-12]	1Fh	[23-20]

This operation is identical to that in the DSP MPAX unit. Programming the same values into the BADDR, RADDR, and SEGZ fields results in an identical address-extension operation for a system master access.

7.1.3.3.4.1 SES Aliased Access to MSMC RAM

Using address remapping in the MPAX unit at the SES port, it is possible to map some or all of the MSMC SRAM into external memory space and access it through the SES port. This provides the same ability to alias MSMC memory to external address space for system masters as is available to the DSP through its MPAX unit.

The ability to alias external addresses to the SRAM has the following constraints:

- Crossing between MSMC SRAM and external memory endpoints in the course of a transaction on SES is not supported and results in an addressing error for the endpoint first addressed in the command.
- Addressing errors get prioritized for reporting over protection errors.

7.1.3.3.4.2 SMS MPAX Address Extension

For the SMS interface, the address-extension operation is very similar except that the generated address maintains bits 31-24 to be the same as the original address (that is, the start address of the MSMC SRAM). The SMS_MPAXL contains a correspondingly smaller RADDR field. Note that this resultant address is still a 32-bit address within the address space of the MSMC SRAM and the operation is that of remapping the address segment to a different location within the MSMC SRAM.

It is possible to violate the permissions associated with a segment and checked as part of the memory-protection check by extending the address into a target segment that has more restrictive permissions. This is actually a valid configuration of the MPAX registers to carve out subsegments that overlap segments with differing permissions.

- If segment overlap is not desired or intended, it is a good practice to align the level of restriction in the segment permissions with the MPAX segment matching priority scheme, that is, configure a less restrictive segment A in a lower numbered MPAX pair than a more restrictive segment B.
- If the permissions of segment A are intended to be used for the overlap region between the two segments A and B, configure A in a higher numbered MPAX pair than the one for segment B.

7.1.3.3.4.3 Address Extension Error Reporting

If a SES address extension results in an address that is outside the range of the EMIF configuration registers of external SDRAM, the MSMC EMIF Master Port receives an addressing error from the EMIF and relays this error to the SES interface.

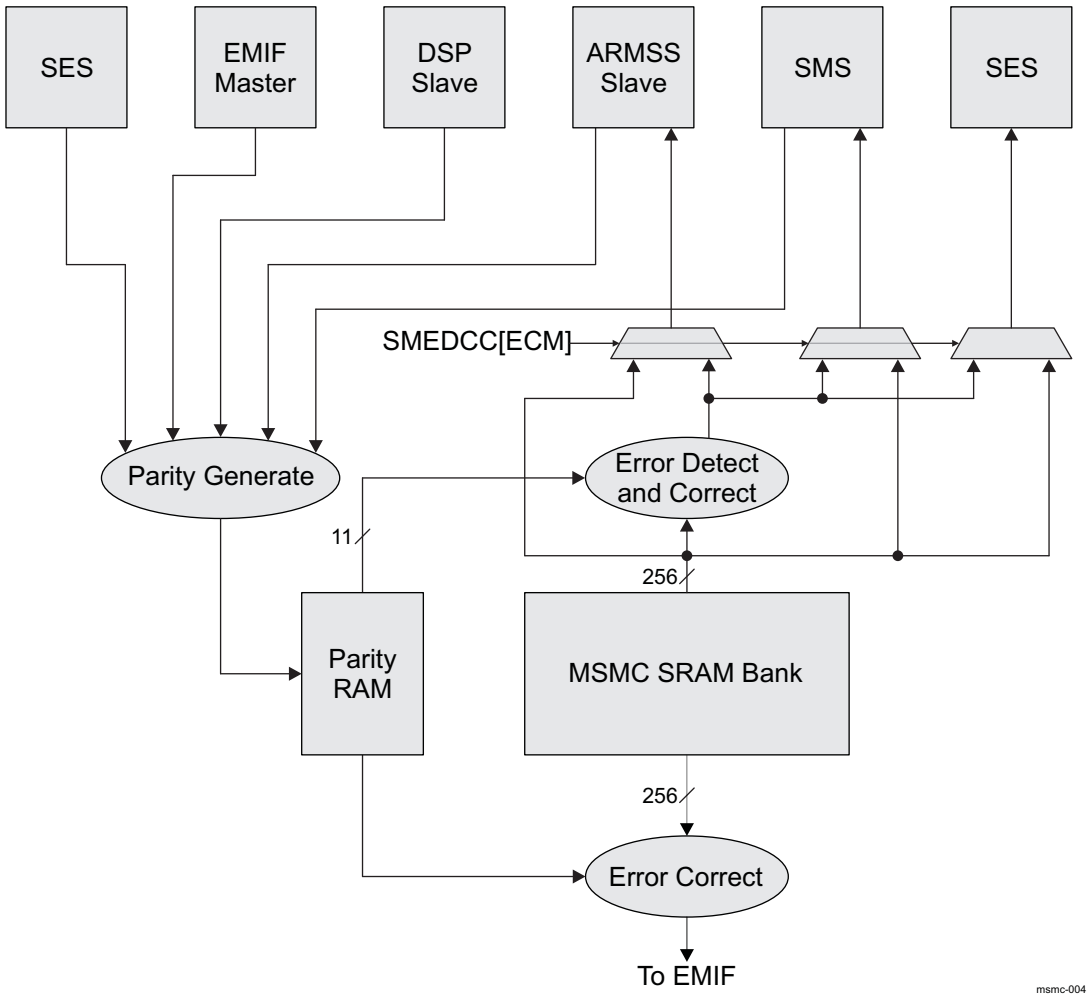
7.1.3.4 Error Detection and Correction (EDC)

The MSMC has error detection and correction hardware to protect the contents of the MSMC memory storage against corruption due to transient (soft) errors. The level of protection provided and the scheme used is one-bit error correction and two-bit error detection with parity codes calculated over a 256-bit datum.

7.1.3.4.1 Parity Generation and Checking

Figure 7-5 shows the hardware available that supports parity generation, storage, and correction. The parity code generated for each location in the MSMC SRAM bank is stored in a corresponding location in the associated parity RAM. That parity RAM location also stores its own parity code which allows detection of transient errors in the parity RAM itself.

Figure 7-5. Error Detection and Correction



7.1.3.4.2 EDC Operation

In the MSMC EDC scheme, parity is generated and tracked for each 256-bit datum which corresponds to a RAM location and a data phase for slave port access. A parity RAM entry may become invalid when data is written to the SRAM with a granularity smaller than 256 bits (for example, when the L2 cache for the DSP is off/frozen/bypassed and a write miss to the MSMC SRAM region occurs in its L1D cache). In this case, the valid bit in the corresponding parity RAM entry is reset.

The parity code for a line is recalculated when the line is transferred through any of the outgoing paths shown in Figure 7-5. The assumption on the reliability of the data being stored and transferred to/from the MSMC is that any incoming data for MSMC storage is reliable and a parity code generated from it is valid.

The EDC hardware consists of the registers listed in Table 7-8 and described in Section 7.1.4.2.

Table 7-8. MSMC EDC Registers

Register	Register Description
MSMC_SMCERRAR	MSMC SRAM correctable EDC error address register
MSMC_SMCERRXR	MSMC SRAM correctable EDC extended error register
MSMC_SMEDCC	MSMC SRAM EDC control register
MSMC_SMCEA	MSMC scrubbing error corrected address register
MSMC_SMSECC	MSMC scrubbing error corrected counter register

Table 7-8. MSMC EDC Registers (continued)

Register	Register Description
MSMC_SMNCERRAR	MSMC SRAM non-correctable EDC error address register
MSMC_SMNCERRXR	MSMC SRAM non-correctable EDC extended error register
MSMC_SMNCEA	MSMC scrubbing non-correctable address register

7.1.3.4.2.1 Error Correction Mode

Error correction adds latency for data-read accesses from slave ports. The MSMC_SMEDCC[30] ECM bit controls if error correction is enabled on all the slave port read paths. Setting the ECM bit to 0h disables error correction for slave accesses (the direct return path in Figure 7-5). Setting the ECM bit to 1h enables the error correction. Any further writes to this bit are ignored which means, once enabled, error correction stays enabled until the MSMC is reset.

7.1.3.4.2.1.1 Soft Error Correction

For data being transferred out of the MSMC storage, the EDC hardware generates the parity for the data and checks against the corresponding stored parity for errors and log information in the EDC registers. The actions and reporting are summarized in Table 7-9. More details about the interrupts can be found in Section 7.1.3.5.

Table 7-9. Soft Error Correction Actions

Read Access Port	ECM state	Soft Error Type	Interrupt Indicated in:	Data Returned
ARMSS/DSP Slave, SES, SMS	1	Correctable	MSMC_SMIRSTAT[3] CEI	Corrected
	1	Non-Correctable	MSMC_SMIRSTAT[2] NCEI	0
	1	Parity RAM	-	Uncorrected
	0	Correctable	MSMC_SMIRSTAT[3] CEI	Uncorrected
	0	Non-Correctable	MSMC_SMIRSTAT[2] NCEI	Uncorrected
	0	Parity RAM	-	Uncorrected

7.1.3.4.3 EDC Error Reporting

When an error is detected, the EDC hardware reports the error details in the EDC error reporting registers as follows:

- For a correctable (one-bit) error:
  - The lower 32 bits of the 36-bit address used in accessing the corrupted location are stored in the MSMC_SMCERRAR register and the upper 4 bits of the 36-bit address in the MSMC_SMCERRXR[3-0] SEEADDR bit field.
  - The syndrome for the corrected location is logged in the MSMC_SMCERRXR[23-16] ESYN bit field.
  - The PrivID associated with the access is saved in the MSMC_SMCERRXR[7-4] SEPID bit field. If the request came in from the SMS or the SES port, the MSMC_SMCERRXR[8] SER bit is set.
  - Generates MSMC_DEDC_CERROR interrupt if it is enabled in the MSMC_SMIESTAT register.
- For a non-correctable (two-bit) error:
  - The lower 32 bits of the 36-bit address used in accessing the corrupted location are stored in the MSMC_SMNCERRAR register and the upper 4 bits of the 36-bit address in the MSMC_SMNCERRXR[3-0] SEEADDR bit field.
  - The PrivID associated with the access is saved in the MSMC_SMNCERRXR[7-4] SEPID bit field. If the request came in from the SMS or the SES port, the MSMC_SMNCERRXR[8] SER bit is set.
  - Generates MSMC_DEDC_NC_ERROR interrupt if it is enabled in the MSMC_SMIESTAT register.
- After they are written, the error-logging registers MSMC_SMCERRAR/MSMC_SMNCERRAR and MSMC_SMCERRXR/MSMC_SMNCERRXR store the contents until further error logging is re-enabled

by clearing the associated interrupt status with a write to the MSMC_SMIRC register (see Section 7.1.3.5).

7.1.3.4.4 Background Parity Refresh (Scrubbing)

As parity is tracked at a granularity equal to the width of a subbank (32 bytes), writes that are smaller than 32 bytes can invalidate the parity information for a line. MSMC contains a background error correction hardware called Scrubbing Engine that periodically refreshes the parity bits for the memory.

The MSMC scrubbing engine is a state machine that periodically cycles through each location of each memory subbank in the MSMC, reading and correcting the data, recalculating the parity bits for the data, and storing the data and parity information. Each such “scrubbing cycle” consists of a series of read-modify-write “scrub bursts” to the memory subbanks.

7.1.3.4.4.1 Scrubbing Rate

The frequency with which each scrubbing cycle is initiated and the delay between each burst by the scrubbing engine is programmed using the MSMC_SMEDCC register:

- The MSMC_SMEDCC[7-0] REFDEL bit field is programmed to control the number of MSMC clock cycles between each scrub burst. To prevent the bursts of the scrubbing engine from posing a significant performance impact, the value in the REFDEL bit field is prescaled by 1024.
- When prescaling of the REFDEL is enabled, a value of 0h is interpreted to be the same as a value of 1h. When the prescaling is disabled, a value of 0h is interpreted as 0, that is, no delay between scrubbing bursts.
- The operation of the scrubbing engine is enabled by default after reset and parity RAM initialization but may be disabled by setting the MSMC_SMEDCC[31] SEN bit. There is no setup required to enable the scrubbing operation aside from checking for an end of the reset sequence as described in Section 7.1.3.4.5.

7.1.3.4.4.2 Scrubbing Error Logging and Statistics Collection

The MSMC scrubbing engine can log errors and collect statistics about scrubbing errors locally:

- If the scrubbing engine access detects a location where contents have been corrected, it logs the address in the MSMC_SMCEA[23-0] SECA bit field, logs the syndrome value (that identifies the erroneous bit in the data) in the MSMC_SMCEA[31-24] ESYN bit field, and increments the MSMC_SMSECC[15-0] SCEC counter bit field. It also generates MSMC_SCRUB_CERROR interrupt (if it is enabled in the MSMC_SMIESTAT register) indicating correctable (1-bit) soft error detected during scrub cycle.
- If the scrubbing engine access detects a two-bit error that is not correctable, it logs the address in the MSMC_SMNCEA[23-0] SENCA bit field and increments the MSMC_SMSECC[31-16] SNCEC counter bit field. It also generates MSMC_SCRUB_NC_ERROR interrupt (if it is enabled in the MSMC_SMIESTAT register) indicating non-correctable (2-bit) soft error detected during scrub cycle.
- Once written, the error logging registers MSMC_SMCEA and MSMC_SMNCEA store the contents till further error logging is re-enabled and these registers are cleared by clearing the associated interrupt status with a write to the MSMC_SMIRC register (see Section 7.1.3.5).
- The MSMC_SMCEA and MSMC_SMNCEA registers are both saturating counters and count all correctable and non-correctable errors, respectively, independent of the logging-enable state of the MSMC_SMCEA and MSMC_SMNCEA registers.

Periodically, the software can read the statistics collected to analyze persistent soft errors. Reading saturated counts in the error counters should indicate to the software that the error counters need to be checked more frequently to get a better indication of the error rate. If the error rate is high, scrub cycles should be initiated more frequently to decrease the probability of an un-repairable two-bit error. If the error rate is low, less frequent scrubbing may be needed to reduce power consumption and reduce interference with functional memory-access traffic.

7.1.3.4.5 Parity RAM Initialization at Reset

The MSMC EDC hardware invalidates all the parity RAM entries at reset. During this initialization cycle error correction or detection is not provided but accesses from the slaves or scrubbing engine are not stalled:

- Read accesses to the memory are serviced but no data check is done, that is, the memory locations are not protected from soft errors during this period. These initialization accesses are done with the prescaler disabled. As a result, no spurious events are generated till the parity RAM is initialized completely. At the end of the initialization the default prescaler delay of 1024 is applied.
- The MSMC_SMEDCC[26] PRR bit shows the status of this process and is 1h when the Parity RAM is ready for use in EDC and scrubbing operations. PRR is 0h after reset while the entries are being initialized. Software must check this bit before making the first read accesses to the MSMC SRAM after reset.
- Write accesses of a full 32-byte location during this period generates the corresponding parity value into the parity RAM location. When a write access arrives in the same cycle as the initialization write of the associated parity RAM location, the former takes precedence.

7.1.3.5 MSMC Interrupts

The MSMC module generates a bunch of interrupts as shown in Table 7-3.

The MSMC has a set of interrupt status and enable registers that control the generation of interrupts at the MSMC module boundary.

The MSMC interrupt operation is controlled by the registers listed in Table 7-10 and described in Section 7.1.4.6.

Table 7-10. MSMC Interrupt Control Registers

Register	Description
MSMC_SMESTAT	Interrupt status register. ANDed value of MSMC_SMIRSTAT and MSMC_SMIESTAT registers. A bit in this register is set if an interrupt condition occurs and the corresponding interrupt is enabled through the MSMC_SMIESTAT register.
MSMC_SMIRSTAT	Interrupt raw status register.
MSMC_SMIRC	Interrupt raw status clear register. Writing 0h to a bit has no effect. Writing 1h clears the coresponding bit in the MSMC_SMIRSTAT register.
MSMC_SMIESTAT	Interrupt enable register. Writing 1h to a bit enables the corresponding interrupt. Reading 0h indicates that the interrupt is disabled.
MSMC_SMIEC	Interrupt disable register. Writing 1h to a bit disables the corresponding interrupt.

The MSMC_SMIRSTAT register stores the raw interrupt status for each interrupt line. A bitfield in this register is set to 1h if the corresponding interrupt has occurred. An interrupt can be generated either when the event occurs in hardware (hardware interrupt) or when software writes to the corresponding bit in the MSMC_SMIRSTAT register (software interrupt for simulating an event). Regardless of how the interrupt is generated, it is cleared by writing 1h to the corresponding bit in the MSMC_SMIRC register.

NOTE: The memory protection fault interrupt for each PrivID is recorded in the MSMC_SMIRSTAT and the associated address and master information is logged into the MSMC_SMPFAR and MSMC_SMPFXR registers (see Section 7.1.4.3). Software must clear the MSMC_SMPFAR and MSMC_SMPFXR registers before clearing the corresponding bits for the PrivID in the MSMC_SMIRSTAT register so that the event status and associated master information are kept synchronized. This order is not enforced in hardware.

For interrupt signalling and status capture the following rules apply:

- For a given interrupt line, once an interrupt is signaled, its status is set and needs to be cleared before another occurrence can be signaled. The status for an interrupt line is set even if it is not enabled in the MSMC_SMIESTAT. Therefore, it is recommended for software to clear the interrupt status while enabling the interrupt. This can be done atomically with a double-word-store writing to the MSMC_SMIESTAT and the MSMC_SMIRC registers together.

- Because there are three sources that can modify a bit in the MSMC_SMIRSTAT, the following precedence rules apply:
  - A hardware interrupt takes precedence over a software interrupt if they arrive at the same time. This is not an intended usage mode as one would either enable a hardware interrupt to signal a real hardware event or use a software write to simulate the same event.
  - A write to set an event in the MSMC_SMIRSTAT register takes precedence over clearing an event in MSMC_SMIRC register. This situation generates an interrupt corresponding to the event that is set.

7.1.3.6 MSMC Register Access Control

The MSMC configuration registers are accessible by the DSP and all system masters that can access the MSMC through the SMS port. Updates to these registers must not be made while there are accesses through any of the system ports utilizing the registers being written to. For example, software needs to ensure that updates to a SES MPAX register are done only when there are no ongoing transfers from the same PrivID using the SES interface.

NOTE: The MSMC configuration register hardware cannot guarantee the determinism of a transfer if the configuration registers used by the transfer are being modified simultaneously. If such an update is done to the registers, the outcome of the transfer is not deterministic and undefined.

While reads to these registers from any of the masters are not restricted, it is often useful to restrict write access to the registers to prevent runaway code on one of the cores or an incorrectly-configured system master from corrupting the state of MSMC configuration. While a semaphore-based mutex access control is still expected to be used by software on the cores (and any masters that may program the registers), MSMC provides a simple lock mechanism to protect against runaway pointer writes.

MSMC register access control is managed by the lock registers listed in Table 7-11 and described in Section 7.1.4.5.

For the purpose of write-access control the MSMC registers are grouped into three categories:

- SMS MPAX registers
- SES MPAX registers
- Non-MPAX registers.

For each category, a write-lock bit (one per PrivID for the MPAX registers) controls whether writes are enabled.

Table 7-11. MSMC Configuration Write Lock Registers

Register	Description	Comment
MSMC_CFGLCK	Configuration Lock control for non-MPAX registers	Lock/unlock/status bits for non-MPAX registers with the exception of the write lock registers for all three categories
MSMC_CFGULCK	Configuration Unlock control for non-MPAX registers	
MSMC_CFGLCKSTAT	Configuration Lock Status for non-MPAX registers	
MSMC_SMS_MPAX_LCK	Configuration Lock control for SMS MPAX registers	16 lock/unlock/status bits (one bit per PrivID)
MSMC_SMS_MPAX_ULCK	Configuration Unlock control for SMS MPAX registers	
MSMC_SMS_MPAX_LCKSTAT	Configuration Lock Status for SMS MPAX registers	
MSMC_SES_MPAX_LCK	Configuration Lock control for SES MPAX registers	
MSMC_SES_MPAX_ULCK	Configuration Unlock control for SES MPAX registers	
MSMC_SES_MPAX_LCKSTAT	Configuration Lock Status for SES MPAX registers	

All the LCK and ULCK registers are modified only when the upper 16 bits of the write data match the fixed MGCID key field of these registers. Any other value used in the upper 16 bits of the write data does not modify its WLCK/WEN bits. For each category, the sequence of steps is as follows:

- To lock, write *MGCID_WLCK* into the LCK register (for example, write *2CD1_0440* to lock the SMS MPAX registers for PrivIDs 7 and 11). For non-MPAX registers, write *2CD0_0001* to the

MSMC_CFGLCK register. Until it is unlocked, all further writes to that category registers generate protection errors.

- To unlock, write *MGCID_WEN* into the ULCK register (for example, write *2CD1_0440* to unlock the SMS MPAX registers for PrivIDs 7 and 11). For non-MPAX registers, write *2CD0_0001* to the MSMC_CFGULCK register. Until it is locked, all further writes to that category of configuration registers generate protection errors.

NOTE: This provides a simple write protection mechanism that protects against unintentional modification only. It is recommended that the software uses a semaphore to ensure exclusive access when modifying the MSMC lock registers.

7.1.3.7 Reset Considerations

The MSMC module resets on all global resets. Upon reset, the following sequence apply:

- All MSMC configuration registers are reset to their initial default state.
- The parity RAM entries associated with the MSMC SRAM are invalidated as described in Section 7.1.3.4.5.
- The MSMC pipeline is reset to an idle state terminating any outstanding accesses.

In addition, the MSMC_PARITY_RST_BLOCK reset signal can be blocked or not depending on the value of the CHIP_MISC_CTL0[12] MSMC_BLOCK_PARITY_RST bit in the BOOT_CFG module. Writing a value of 1h to this bit blocks the reset.

7.1.3.8 MSMC Memory Regions

Table 7-12 shows the MSMC associated memory regions.

Table 7-12. MSMC Memory Regions

Region	Start Address	End Address	Region Size
MSMC SRAM	0C00 0000h	0C0F FFFFh	1MB
MSMC configuration registers	0BC0 0000h	0BCF FFFFh	1MB

7.1.3.9 MSMC Cache Coherence

MSMC supports hardware cache coherence between the ARMSS L1/L2 caches and EDMA/system master peripherals for shared SRAM and SDRAM spaces. This feature allows the sharing of MSMC SRAM and SDRAM data spaces by these masters on the chip, without having to use explicit software cache maintenance techniques.

MSMC does not support memory coherence for these spaces:

- EMIF Configuration Registers Space
- MSMC Configuration Registers Space
- System Master Port Peripherals/Memory
- Any memory not directly connected to MSMC.

7.1.3.9.1 Cache Coherence Operation

Coherent masters typically track more states than the traditional valid/dirty combinations to maintain coherence. MSMC uses the ACE (AXI Coherence Extensions) coherence protocol which supports the following five states (MOESI)

- Modified - Cache line is unique (no other caches currently have it) and dirty
- Owned - Cache line is shared (other caches may have it) and dirty
- Exclusive - Cache line is unique and clean
- Shared - Cache line is shared and clean

- Invalid - No allocated line.

In order to maintain coherence between cached masters, MSMC can initiate requests to coherent masters for shared regions of memory. These requests are called as snoop requests and are performed as per the ACE protocol. MSMC maintains coherence by snooping coherent caches for the latest value when an access hits a shared memory region (SES/SMS MPAXHn[7] US = 0h).

The following example demonstrates how coherence is maintained between EDMA and ARMSS, when EDMA writes to shareable memory space.

1. EDMA (master with no cache) issues a write to shareable space.
2. MSMC coherence controller issues Invalidate and Writeback snoop to the CPU.
3. CPU invalidates the line from its cache and responds with the dirty data
4. MSMC merges the EDMA write data with the victim and commits to memory.

The following example demonstrates how coherence is maintained between EDMA and ARMSS, when EDMA reads from shareable memory space.

1. EDMA issues a read to shareable space.
2. MSMC coherence controller issues a ReadOnce snoop to the CPU.
3. CPU responds with cached data.
4. MSMC returns the read data to EDMA.

Software has the ability to control which memory regions are shared among certain sets of coherent masters using ARMSS MMU and using SMS_MPAXH or SES_MPAXH register for EDMA/system master peripherals. Software should ensure that the shareability mappings between the types of masters are consistent to avoid unexpected behavior.

For example, a memory segment marked as outer shared in ARMSS MMU and marked as unshared in the respective SMS_MPAXH[7] US or SES_MPAXH[7] US bit is an inconsistent mapping and can cause unpredictable behavior.

7.1.4 MSMC Registers

Table 7-14 lists the MSMC configuration registers. All other register offset addresses not listed in Table 7-14 should be considered as reserved locations and the register contents should not be modified.

Table 7-13. MSMC Instances

Instance	Base Address
MSMC	0BC0 0000h

Table 7-14. MSMC Registers

Offset	Acronym	Register Name	MSMC Physical Address	Section
0h	MSMC_PID	MSMC Peripheral ID register	0BC0 0000h	Section 7.1.4.1
8h	MSMC_SMCERRAR	MSMC SRAM correctable EDC error address register	0BC0 0008h	Section 7.1.4.2.2
Ch	MSMC_SMCERRXR	MSMC SRAM correctable EDC extended error register	0BC0 000Ch	Section 7.1.4.2.3
10h	MSMC_SMEDCC	MSMC SRAM EDC control register	0BC0 0010h	Section 7.1.4.2.1
14h	MSMC_SMCEA	MSMC Scrubbing Error Corrected Address register	0BC0 0014h	Section 7.1.4.2.6
18h	MSMC_SMSECC	MSMC Scrubbing Error Counter register	0BC0 0018h	Section 7.1.4.2.8
1Ch	MSMC_SMPFAR	MSMC Memory Protection Fault Address register	0BC0 001Ch	Section 7.1.4.3.1
20h	MSMC_SMPFXR	MSMC Memory Protection Fault Extension register	0BC0 0020h	Section 7.1.4.3.2
24h	MSMC_SMPFR	MSMC Memory Protection Fault Requestor register	0BC0 0024h	Section 7.1.4.3.3
28h	MSMC_SMPFCR	MSMC Memory Protection Fault Control register	0BC0 0028h	Section 7.1.4.3.4
30h	MSMC_SBNDC0	Starvation bound register for DSP slave port	0BC0 0030h	Section 7.1.4.4.1
50h	MSMC_SBNDM	Starvation bound register for SMS port	0BC0 0050h	Section 7.1.4.4.2
54h	MSMC_SBNDE	Starvation bound register for SES port	0BC0 0054h	Section 7.1.4.4.3
5Ch	MSMC_CFGLCK	Configuration Lock control for non-MPAX registers	0BC0 005Ch	Section 7.1.4.5.1
60h	MSMC_CFGULCK	Configuration Unlock control for non-MPAX registers	0BC0 0060h	Section 7.1.4.5.2
64h	MSMC_CFGLCKSTAT	Configuration Lock Status for non-MPAX registers	0BC0 0064h	Section 7.1.4.5.3
68h	MSMC_SMS_MPAX_LCK	Configuration Lock control for SMS MPAX registers	0BC0 0068h	Section 7.1.4.5.4
6Ch	MSMC_SMS_MPAX_ULCK	Configuration Unlock control for SMS MPAX registers	0BC0 006Ch	Section 7.1.4.5.5
70h	MSMC_SMS_MPAX_LCKSTAT	Configuration Lock Status for SMS MPAX registers	0BC0 0070h	Section 7.1.4.5.6
74h	MSMC_SES_MPAX_LCK	Configuration Lock control for SES MPAX registers	0BC0 0074h	Section 7.1.4.5.7
78h	MSMC_SES_MPAX_ULCK	Configuration Unlock control for SES MPAX registers	0BC0 0078h	Section 7.1.4.5.8
7Ch	MSMC_SES_MPAX_LCKSTAT	Configuration Lock Status for SES MPAX registers	0BC0 007Ch	Section 7.1.4.5.9
80h	MSMC_SMESTAT	Interrupt status register. ANDed value of MSMC_SMIRSTAT and MSMC_SMIESTAT registers	0BC0 0080h	Section 7.1.4.6.1
84h	MSMC_SMIRSTAT	Interrupt raw status register	0BC0 0084h	Section 7.1.4.6.2

Table 7-14. MSMC Registers (continued)

Offset	Acronym	Register Name	MSMC Physical Address	Section
88h	MSMC_SMIRC	Interrupt raw status clear register	0BC0 0088h	Section 7.1.4.6.3
8Ch	MSMC_SMIESTAT	Interrupt enable register	0BC0 008Ch	Section 7.1.4.6.4
90h	MSMC_SMIEC	Interrupt disable register	0BC0 0090h	Section 7.1.4.6.5
C4h	MSMC_SMNCERRAR	MSMC SRAM non-correctable EDC error address register	0BC0 00C4h	Section 7.1.4.2.4
C8h	MSMC_SMNCERRXR	MSMC SRAM non-correctable EDC extended error register	0BC0 00C8h	Section 7.1.4.2.5
CCh	MSMC_SMNCEA	MSMC Scrubbing Non-Correctable Address register	0BC0 00CCh	Section 7.1.4.2.7
200h + (z*8)	MSMC_SMS_MPAXL_x_y	SMS MPAX register pair y for PrivID x x = 0 to 15; y = 0 to 7; z = 0 to 127 (x;y;z) = (0;0;0, 0;1;1, 0;2;2, ... 15;5;125, 15;6;126, 15;7;127)	0BC0 0200h + (z*8)	Section 7.1.4.7.1
204h + (z*8)	MSMC_SMS_MPAXH_x_y		0BC0 0204h + (z*8)	Section 7.1.4.7.2
600h + (z*8)	MSMC_SES_MPAXL_x_y	SES MPAX register pair y for PrivID x x = 0 to 15; y = 0 to 7; z = 0 to 127 (x;y;z) = (0;0;0, 0;1;1, 0;2;2, ... 15;5;125, 15;6;126, 15;7;127)	0BC0 0600h + (z*8)	Section 7.1.4.7.3
604h + (z*8)	MSMC_SES_MPAXH_x_y		0BC0 0604h + (z*8)	Section 7.1.4.7.4

7.1.4.1 MSMC_PID Register (Offset = 0h) [reset = 45000A02h]

The peripheral identification register (MSMC_PID) uniquely identifies the MSMC and the specific revision of the MSMC.

The MSMC_PID is shown in Figure 7-6 and described in Table 7-16.

Table 7-15. MSMC_PID Instances

Instance	Physical Address
MSMC	0BC0 0000h

Figure 7-6. MSMC_PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PID																															
R-4500 0A02h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 7-16. MSMC_PID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PID	R	4500 0A02h	TI internal data. Identifies revision of peripheral.

Table 7-17. Register Call Summary for MSMC_PID

MSMC Registers
- MSMC Registers: [0]
- MSMC_PID Register (Offset = 0h) [reset = 45000A02h]: [0][1]

7.1.4.2 EDC Registers

7.1.4.2.1 MSMC_SMEDCC Register (Offset = 10h) [reset = 40000001h]

EDC operation is controlled by MSMC_SMEDCC.

The MSMC_SMEDCC is shown in Figure 7-7 and described in Table 7-19.

Table 7-18. MSMC_SMEDCC Instances

Instance	Physical Address
MSMC	0BC0 0010h

Figure 7-7. MSMC_SMEDCC Register

31	30	29	28	27	26	25	24
SEN	ECM	RESERVED			PRR	RESERVED	
RW-0h	RW-1h	R-0h			RW-0h	R-0h	
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
REFDEL							
RW-1h							

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 7-19. MSMC_SMEDCC Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SEN	RW	0h	Scrubbing Engine Enable • 0 = The operation of the scrubbing engine is enabled by default after reset and parity RAM initialization • 1 = The operation of the scrubbing engine is disabled
30	ECM	RW	1h	Error Correction Mode • 0 = Disables error correction • 1 = Enables error correction
29-27	RESERVED	R	0h	Reads return 0 and writes have no effect.
26	PRR	RW	0h	The PRR (Parity RAM Ready) bit shows the status of Parity RAM. • 0 = Parity RAM is not ready, this will be the state following a reset while the entries are being initialized. • 1 = Parity RAM is ready for use in EDC and scrubbing operations. Software must consult this bit before making the first read access to MSMC RAM after reset.
25-8	RESERVED	R	0h	Reads return 0 and writes have no effect.
7-0	REFDEL	RW	1h	Value = 0-FFh Controls the number of MSMC clock cycles between each scrub burst. To prevent the bursts from the scrubbing engine from posing a significant performance impact, the value in the REFDEL register is pre-scaled by 1024.

Table 7-20. Register Call Summary for MSMC_SMEDCC

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • Scrubbing Rate: [0][1][2] • Error Correction Mode: [0] • EDC Operation: [0] • Parity RAM Initialization at Reset: [0]
EDC Registers • MSMC_SMEDCC Register (Offset = 10h) [reset = 1h]: [0][1]

7.1.4.2.2 MSMC_SMCERRAR Register (Offset = 8h) [reset = 0h]

On detection of a correctable one-bit error, the EDC hardware reports the error details in the correctable EDC error reporting registers.

The MSMC_SMCERRAR is shown in Figure 7-8 and described in Table 7-22.

Table 7-21. MSMC_SMCERRAR Instances

Instance	Physical Address
MSMC	0BC0 0008h

Figure 7-8. MSMC_SMCERRAR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEADDR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 7-22. MSMC_SMCERRAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SEADDR	R	0h	Value = 0-FFFF FFFFh The lower 32 bits of the 36 bit address used in the accessing the corrupted location is stored here. The upper 4 bits of the 36 bit address in stored in SEEADDR field of the MSMC_SMCERRXR register.

Table 7-23. Register Call Summary for MSMC_SMCERRAR

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • EDC Error Reporting: [0][1] • EDC Operation: [0]
EDC Registers • MSMC_SMCERRXR Register (Offset = Ch) [reset = 0h]: [0] • MSMC_SMCERRAR Register (Offset = 8h) [reset = 0h]: [0]

7.1.4.2.3 MSMC_SMCERRXR Register (Offset = Ch) [reset = 0h]

On detection of a correctable one-bit error, the EDC hardware reports the error details in the correctable EDC error reporting registers.

The MSMC_SMCERRXR is shown in Figure 7-9 and described in Table 7-25.

Table 7-24. MSMC_SMCERRXR Instances

Instance	Physical Address
MSMC	0BC0 000Ch

Figure 7-9. MSMC_SMCERRXR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
ESYN							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							SER
R-0h							R-0h
7	6	5	4	3	2	1	0
SEPID				SEEADDR			
R-0h				R-0h			

LEGEND: R = Read Only; *-n* = value after reset

Table 7-25. MSMC_SMCERRXR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return 0 and writes have no effect.
23-16	ESYN	R	0h	Value = 0-FFh Logs the syndrome value that identifies the erroneous bit in the data.
15-9	RESERVED	R	0h	Reads return 0 and writes have no effect.
8	SER	R	0h	• 0 = Request came in from the DSP. • 1 = Request came in from the SMS or the SES port.
7-4	SEPID	R	0h	Value = 0-Fh The PrivID associated with the access.
3-0	SEEADDR	R	0h	Value = 0-Fh The upper 4 bits of the 36 bit address used in the accessing the corrupted location is stored here. The lower 32 bits of the 36 bit address in stored in SEADDR field of the MSMC_SMCERRAR register.

Table 7-26. Register Call Summary for MSMC_SMCERRXR

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • EDC Error Reporting: [0][1][2][3][4] • EDC Operation: [0]
EDC Registers • MSMC_SMCERRXR Register (Offset = Ch) [reset = 0h]: [0] • MSMC_SMCERRAR Register (Offset = 8h) [reset = 0h]: [0]

7.1.4.2.4 MSMC_SMNCERRAR Register (Offset = C4h) [reset = 0h]

On detection of a non-correctable two-bit error, the EDC hardware reports the error details in the non-correctable EDC error reporting registers.

The MSMC_SMNCERRAR is shown in Figure 7-10 and described in Table 7-28.

Table 7-27. MSMC_SMNCERRAR Instances

Instance	Physical Address
MSMC	0BC0 00C4h

Figure 7-10. MSMC_SMNCERRAR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEADDR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 7-28. MSMC_SMNCERRAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SEADDR	R	0h	Value = 0-FFFF FFFFh The lower 32 bits of the 36 bit address used in the accessing the corrupted location is stored here. The upper 4 bits of the 36 bit address in stored in SEEADDR field of the MSMC_SMNCERRXR register.

Table 7-29. Register Call Summary for MSMC_SMNCERRAR

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • EDC Error Reporting: [0][1] • EDC Operation: [0]
EDC Registers • MSMC_SMNCERRAR Register (Offset = C4h) [reset = 0h]: [0] • MSMC_SMNCERRXR Register (Offset = C8h) [reset = 0h]: [0]

7.1.4.2.5 MSMC_SMNCERRXR Register (Offset = C8h) [reset = 0h]

On detection of a non-correctable two-bit error, the EDC hardware reports the error details in the non-correctable EDC error reporting registers.

The MSMC_SMNCERRXR is shown in Figure 7-11 and described in Table 7-31.

Table 7-30. MSMC_SMNCERRXR Instances

Instance	Physical Address
MSMC	0BC0 00C8h

Figure 7-11. MSMC_SMNCERRXR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							SER
R-0h							R-0h
7	6	5	4	3	2	1	0
SEPID				SEEADDR			
R-0h				R-0h			

LEGEND: R = Read Only; *-n* = value after reset

Table 7-31. MSMC_SMNCERRXR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reads return 0 and writes have no effect.
8	SER	R	0h	• 0 = Request came in from the DSP. • 1 = Request came in from the SMS or the SES port.
7-4	SEPID	R	0h	Value = 0-Fh The PrivID associated with the access.
3-0	SEEADDR	R	0h	Value = 0-Fh The upper 4 bits of the 36 bit address used in the accessing the corrupted location is stored here. The lower 32 bits of the 36 bit address in stored in SEADDR field of the MSMC_SMNCERRAR register.

Table 7-32. Register Call Summary for MSMC_SMNCERRXR

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • EDC Error Reporting: [0][1][2][3] • EDC Operation: [0]
EDC Registers • MSMC_SMNCERRAR Register (Offset = C4h) [reset = 0h]: [0] • MSMC_SMNCERRXR Register (Offset = C8h) [reset = 0h]: [0]

7.1.4.2.6 MSMC_SMCEA Register (Offset = 14h) [reset = 0h]

On detection of a correctable one-bit error, the scrubbing engine reports the address in the MSMC_SMCEA register.

The MSMC_SMCEA is shown in Figure 7-12 and described in Table 7-34.

Table 7-33. MSMC_SMCEA Instances

Instance	Physical Address
MSMC	0BC0 0014h

Figure 7-12. MSMC_SMCEA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ESYN								SECA																							
R-0h								R-0h																							

LEGEND: R = Read Only; *-n* = value after reset

Table 7-34. MSMC_SMCEA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	ESYN	R	0h	Value = 0-FFh Logs the syndrome value that identifies the erroneous bit in the data.
23-0	SECA	R	0h	Value = 0-FF FFFFh Scrubbing Error Correctable Address. If the scrubbing engine access detects that a location contents have been corrected, it logs the address.

Table 7-35. Register Call Summary for MSMC_SMCEA

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • Scrubbing Error Logging and Statistics Collection: [0][1][2][3][4] • EDC Operation: [0]
EDC Registers • MSMC_SMCEA Register (Offset = 14h) [reset = 0h]: [0][1]

7.1.4.2.7 MSMC_SMNCEA Register (Offset = CCh) [reset = 0h]

On detection of a non-correctable two-bit error, the scrubbing engine reports the address in the MSMC_SMNCEA register.

The MSMC_SMNCEA is shown in Figure 7-13 and described in Table 7-37.

Table 7-36. MSMC_SMNCEA Instances

Instance	Physical Address
MSMC	0BC0 00CCh

Figure 7-13. MSMC_SMNCEA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								SENCA																							
R-0h								R-0h																							

LEGEND: R = Read Only; *-n* = value after reset

Table 7-37. MSMC_SMNCEA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return 0 and writes have no effect.
23-0	SENCA	R	0h	• 0-FF FFFFh • Value = Scrubbing Error Non-Correctable Address.

Table 7-38. Register Call Summary for MSMC_SMNCEA

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • Scrubbing Error Logging and Statistics Collection: [0][1][2][3] • EDC Operation: [0]
EDC Registers • MSMC_SMNCEA Register (Offset = CCh) [reset = 0h]: [0][1]

7.1.4.2.8 MSMC_SMSECC Register (Offset = 18h) [reset = 0h]

On detection of a correctable one-bit error, the scrubbing engine increments the SCEC (Scrub Correctable Error Counter) field in the MSMC_SMSECC register. On detection of a non-correctable two-bit error, the scrubbing engine increments the SNCEC (Scrub Non-Correctable Error Counter) field in the MSMC_SMSECC register.

The MSMC_SMSECC is shown in Figure 7-14 and described in Table 7-40.

Table 7-39. MSMC_SMSECC Instances

Instance	Physical Address
MSMC	0BC0 0018h

Figure 7-14. MSMC_SMSECC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SNCEC																SCEC															
RW-0h																RW-0h															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-40. MSMC_SMSECC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	SNCEC	RW	0h	Value = 0-FFFFh Increments the counter on detection of a non-correctable two-bit error.
15-0	SCEC	RW	0h	Value = 0-FFFFh Increments the counter on detection of a correctable one-bit error.

Table 7-41. Register Call Summary for MSMC_SMSECC

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • Scrubbing Error Logging and Statistics Collection: [0][1] • EDC Operation: [0]
EDC Registers • MSMC_SMSECC Register (Offset = 18h) [reset = 0h]: [0][1][2]

7.1.4.3 Memory Protection Fault Reporting Registers

7.1.4.3.1 MSMC_SMPFAR Register (Offset = 1Ch) [reset = 0h]

The MSMC_SMPFAR is shown in Figure 7-15 and described in Table 7-43.

Table 7-42. MSMC_SMPFAR Instances

Instance	Physical Address
MSMC	0BC0 001Ch

Figure 7-15. MSMC_SMPFAR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FAULT_ADDRESS																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 7-43. MSMC_SMPFAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FAULT_ADDRESS	R	0h	0-FFFF FFFFh = Fault Access Address

Table 7-44. Register Call Summary for MSMC_SMPFAR

MSMC Registers • MSMC Registers: [0]
Memory Protection Fault Reporting Registers • MSMC_SMPFCR Register (Offset = 28h) [reset = 0h]: [0] • MSMC_SMPFAR Register (Offset = 1Ch) [reset = 0h]: [0]
MSMC Functional Description • MSMC Interrupts: [0][1] • Memory Protection Fault Reporting: [0][1][2]

7.1.4.3.2 MSMC_SMPFXR Register (Offset = 20h) [reset = 0h]

The MSMC_SMPFXR is shown in Figure 7-16 and described in Table 7-46.

Table 7-45. MSMC_SMPFXR Instances

Instance	Physical Address
MSMC	0BC0 0020h

Figure 7-16. MSMC_SMPFXR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							NM
R-0h							R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 7-46. MSMC_SMPFXR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	NM	R	0h	• 0 = Fault is not caused by address mismatch of the segment BADDR. • 1 = Fault is caused by the address not matching any of the segment BADDR.

Table 7-47. Register Call Summary for MSMC_SMPFXR

MSMC Registers • MSMC Registers: [0]
Memory Protection Fault Reporting Registers • MSMC_SMPFXR Register (Offset = 20h) [reset = 0h]: [0] • MSMC_SMPFCR Register (Offset = 28h) [reset = 0h]: [0]
MSMC Functional Description • MSMC Interrupts: [0][1] • Memory Protection Fault Reporting: [0][1][2]

7.1.4.3.3 MSMC_SMPFR Register (Offset = 24h) [reset = 0h]

The MSMC_SMPFR is shown in Figure 7-17 and described in Table 7-49.

Table 7-48. MSMC_SMPFR Instances

Instance	Physical Address
MSMC	0BC0 0024h

Figure 7-17. MSMC_SMPFR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				FPID			
R-0h				R-0h			
7	6	5	4	3	2	1	0
FMSTID							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 7-49. MSMC_SMPFR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reads return 0 and writes have no effect.
11-8	FPID	R	0h	0-Fh = PrivID associated with the fault access.
7-0	FMSTID	R	0h	0-FFh = Master ID associated with the fault access.

Table 7-50. Register Call Summary for MSMC_SMPFR

MSMC Registers • MSMC Registers: [0]
Memory Protection Fault Reporting Registers • MSMC_SMPFR Register (Offset = 24h) [reset = 0h]: [0]
MSMC Functional Description • Memory Protection Fault Reporting: [0][1]

7.1.4.3.4 MSMC_SMPFCR Register (Offset = 28h) [reset = 0h]

The MSMC_SMPFCR is shown in Figure 7-18 and described in Table 7-52.

Table 7-51. MSMC_SMPFCR Instances

Instance	Physical Address
MSMC	0BC0 0028h

Figure 7-18. MSMC_SMPFCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CLR
R-0h							RW-0h

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-52. MSMC_SMPFCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	CLR	RW	0h	• 0 = Writing 0 has no effect. • 1 = Writing 1 to this bit clears the recorded fault in MSMC_SMPFAR and MSMC_SMPFXR registers, only after which further faults can be recorded.

Table 7-53. Register Call Summary for MSMC_SMPFCR

MSMC Registers • MSMC Registers: [0]
Memory Protection Fault Reporting Registers • MSMC_SMPFCR Register (Offset = 28h) [reset = 0h]: [0]
MSMC Functional Description • Memory Protection Fault Reporting: [0][1]

7.1.4.4 Bandwidth Management Control Registers

7.1.4.4.1 MSMC_SBNDC0 Register (Offset = 30h) [reset = 1F001Fh]

The MSMC_SBNDC0 is shown in Figure 7-19 and described in Table 7-55.

Table 7-54. MSMC_SBNDC0 Instances

Instance	Physical Address
MSMC	0BC0 0030h

Figure 7-19. MSMC_SBNDC0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
SCNTCE							
RW-1Fh							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
SCNTCM							
RW-1Fh							

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 7-55. MSMC_SBNDC0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return 0 and writes have no effect.
23-16	SCNTCE	RW	1Fh	Value = 0-FFh Reload value (pre-scaled by 16) for the starvation counter for DSP requests at the EMIF arbiter.
15-8	RESERVED	R	0h	Reads return 0 and writes have no effect.
7-0	SCNTCM	RW	1Fh	Value = 0-FFh Reload value for the starvation counters for DSP requests at the RAM bank arbiter.

Table 7-56. Register Call Summary for MSMC_SBNDC0

MSMC Registers • MSMC Registers: [0]
Bandwidth Management Control Registers • MSMC_SBNDC0 Register (Offset = 30h) [reset = 1F001Fh]: [0]
MSMC Functional Description • MSMC Bandwidth Management: [0][1]

7.1.4.4.2 MSMC_SBNDM Register (Offset = 50h) [reset = 1Fh]

The MSMC_SBNDM is shown in Figure 7-20 and described in Table 7-58.

Table 7-57. MSMC_SBNDM Instances

Instance	Physical Address
MSMC	0BC0 0050h

Figure 7-20. MSMC_SBNDM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SCNTMM							
R-0h																								RW-1Fh							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-58. MSMC_SBNDM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reads return 0 and writes have no effect.
7-0	SCNTMM	RW	1Fh	Value = 0-FFh Reload value for the starvation counters for SMS requests at the RAM bank arbiter.

Table 7-59. Register Call Summary for MSMC_SBNDM

MSMC Registers • MSMC Registers: [0]
Bandwidth Management Control Registers • MSMC_SBNDM Register (Offset = 50h) [reset = 1Fh]: [0]
MSMC Functional Description • MSMC Bandwidth Management: [0]

7.1.4.4.3 MSMC_SBNDE Register (Offset = 54h) [reset = 1F001Fh]

The MSMC_SBNDE is shown in Figure 7-21 and described in Table 7-61.

Table 7-60. MSMC_SBNDE Instances

Instance	Physical Address
MSMC	0BC0 0054h

Figure 7-21. MSMC_SBNDE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
SCNTEE							
RW-1Fh							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
SCNTEM							
RW-1Fh							

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 7-61. MSMC_SBNDE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return 0 and writes have no effect.
23-16	SCNTEE	RW	1Fh	Value = 0-FFh Reload value (prescaled by 16) for the starvation counter for SES requests at the EMIF arbiter.
15-8	RESERVED	R	0h	Reads return 0 and writes have no effect.
7-0	SCNTEM	RW	1Fh	Value = 0-FFh Reload value for the starvation counters for SES requests at the RAM bank arbiter.

Table 7-62. Register Call Summary for MSMC_SBNDE

MSMC Registers • MSMC Registers: [0]
Bandwidth Management Control Registers • MSMC_SBNDE Register (Offset = 54h) [reset = 1F001Fh]: [0]
MSMC Functional Description • MSMC Bandwidth Management: [0][1]

7.1.4.5 MSMC Configuration Write Lock Registers

7.1.4.5.1 MSMC_CFGLCK Register (Offset = 5Ch) [reset = 2CD00000h]

The MSMC_CFGLCK is shown in Figure 7-22 and described in Table 7-64.

Table 7-63. MSMC_CFGLCK Instances

Instance	Physical Address
MSMC	0BC0 005Ch

Figure 7-22. MSMC_CFGLCK Register

31	30	29	28	27	26	25	24
MGCID							
W-2CD0h							
23	22	21	20	19	18	17	16
MGCID							
W-2CD0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							WLCK
R-0h							W-0h

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 7-64. MSMC_CFGLCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MGCID	W	2CD0h	2CD0h = Writing this key value along with setting WLCK to 1 locks the non-MPAX registers.
15-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	WLCK	W	0h	• 0 = Writing 0 has no effect. • 1 = Writing this bit to 1 along with setting MGCID key = 2CD0h locks the non-MPAX registers.

Table 7-65. Register Call Summary for MSMC_CFGLCK

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • MSMC Register Access Control: [0][1]
MSMC Configuration Write Lock Registers • MSMC_CFGLCK Register (Offset = 5Ch) [reset = 2CD00000h]: [0]

7.1.4.5.2 MSMC_CFGULCK Register (Offset = 60h) [reset = 2CD00000h]

The MSMC_CFGULCK is shown in Figure 7-23 and described in Table 7-67.

Table 7-66. MSMC_CFGULCK Instances

Instance	Physical Address
MSMC	0BC0 0060h

Figure 7-23. MSMC_CFGULCK Register

31	30	29	28	27	26	25	24
MGCID							
W-2CD0h							
23	22	21	20	19	18	17	16
MGCID							
W-2CD0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							WEN
R-0h							W-0h

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 7-67. MSMC_CFGULCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MGCID	W	2CD0h	2CD0h = Writing this key value along with setting WEN to 1 unlocks the non-MPAX registers.
15-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	WEN	W	0h	• 0 = Writing 0 has no effect. • 1 = Writing this bit to 1 along with setting MGCID key = 2CD0h unlocks the non-MPAX registers.

Table 7-68. Register Call Summary for MSMC_CFGULCK

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • MSMC Register Access Control: [0][1]
MSMC Configuration Write Lock Registers • MSMC_CFGULCK Register (Offset = 60h) [reset = 2CD00000h]: [0]

7.1.4.5.3 MSMC_CFGLCKSTAT Register (Offset = 64h) [reset = 0h]

The MSMC_CFGLCKSTAT is shown in Figure 7-24 and described in Table 7-70.

Table 7-69. MSMC_CFGLCKSTAT Instances

Instance	Physical Address
MSMC	0BC0 0064h

Figure 7-24. MSMC_CFGLCKSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							WSTAT
R-0h							R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 7-70. MSMC_CFGLCKSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	WSTAT	R	0h	Indicates the lock's current status. • 0 = non-MPAX registers are unlocked. • 1 = non-MPAX registers are locked.

Table 7-71. Register Call Summary for MSMC_CFGLCKSTAT

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • MSMC Register Access Control: [0]
MSMC Configuration Write Lock Registers • MSMC_CFGLCKSTAT Register (Offset = 64h) [reset = 0h]: [0]

7.1.4.5.4 MSMC_SMS_MPAX_LCK Register (Offset = 68h) [reset = 2CD10000h]

The MSMC_SMS_MPAX_LCK is shown in Figure 7-25 and described in Table 7-73.

Table 7-72. MSMC_SMS_MPAX_LCK Instances

Instance	Physical Address
MSMC	0BC0 0068h

Figure 7-25. MSMC_SMS_MPAX_LCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MGCID																WLCK															
W-2CD1h																W-0h															

LEGEND: W = Write Only; *-n* = value after reset

Table 7-73. MSMC_SMS_MPAX_LCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MGCID	W	2CD1h	Value = 2CD1h Writing this key value along with setting WLCK[n] to 1 locks the SMS MPAX registers for PrivID n.
15-0	WLCK	W	0h	Value = 0-FFFFh Bit n denotes lock bit for PrivID n. Writing WLCK[n] bit to 0 has no effect. Writing WLCK[n] bit to 1 along with setting MGCID key = 2CD1h locks the SMS MPAX registers for PrivID n.

Table 7-74. Register Call Summary for MSMC_SMS_MPAX_LCK

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • MSMC Register Access Control: [0]
MSMC Configuration Write Lock Registers • MSMC_SMS_MPAX_LCK Register (Offset = 68h) [reset = 2CD10000h]: [0]

7.1.4.5.5 MSMC_SMS_MPAX_ULCK Register (Offset = 6Ch) [reset = 2CD10000h]

The MSMC_SMS_MPAX_ULCK is shown in Figure 7-26 and described in Table 7-76.

Table 7-75. MSMC_SMS_MPAX_ULCK Instances

Instance	Physical Address
MSMC	0BC0 006Ch

Figure 7-26. MSMC_SMS_MPAX_ULCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MGCID																WEN															
W-2CD1h																W-0h															

LEGEND: W = Write Only; *-n* = value after reset

Table 7-76. MSMC_SMS_MPAX_ULCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MGCID	W	2CD1h	Value = 2CD1h Writing this key value along with setting WEN[n] to 1 unlocks the SMS MPAX registers for PrivID n.
15-0	WEN	W	0h	Value = 0-FFFFh Bit n denotes unlock bit for PrivID n. Writing WEN[n] bit to 0 has no effect. Writing WEN[n] bit to 1 along with setting MGCID key = 2CD1h unlocks the SMS MPAX registers for PrivID n.

Table 7-77. Register Call Summary for MSMC_SMS_MPAX_ULCK

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • MSMC Register Access Control: [0]
MSMC Configuration Write Lock Registers • MSMC_SMS_MPAX_ULCK Register (Offset = 6Ch) [reset = 2CD10000h]: [0]

7.1.4.5.6 MSMC_SMS_MPAX_LCKSTAT Register (Offset = 70h) [reset = 0h]

The MSMC_SMS_MPAX_LCKSTAT is shown in Figure 7-27 and described in Table 7-79.

Table 7-78. MSMC_SMS_MPAX_LCKSTAT Instances

Instance	Physical Address
MSMC	0BC0 0070h

Figure 7-27. MSMC_SMS_MPAX_LCKSTAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																WSTAT															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 7-79. MSMC_SMS_MPAX_LCKSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reads return 0 and writes have no effect.
15-0	WSTAT	R	0h	Value = 0-FFFFh Bit n indicates the lock's current status for PrivID n. • 0 - SMS MPAX registers are unlocked. • 1 - SMS MPAX registers are locked.

Table 7-80. Register Call Summary for MSMC_SMS_MPAX_LCKSTAT

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • MSMC Register Access Control: [0]
MSMC Configuration Write Lock Registers • MSMC_SMS_MPAX_LCKSTAT Register (Offset = 70h) [reset = 0h]: [0]

7.1.4.5.7 MSMC_SES_MPAX_LCK Register (Offset = 74h) [reset = 2CD20000h]

The MSMC_SES_MPAX_LCK is shown in Figure 7-28 and described in Table 7-82.

Table 7-81. MSMC_SES_MPAX_LCK Instances

Instance	Physical Address
MSMC	0BC0 0074h

Figure 7-28. MSMC_SES_MPAX_LCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MGCID																WLCK															
W-2CD2h																W-0h															

LEGEND: W = Write Only; *-n* = value after reset

Table 7-82. MSMC_SES_MPAX_LCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MGCID	W	2CD2h	Value = 2CD2h Writing this key value along with setting WLCK[n] to 1 locks the SES MPAX registers for PrivID n.
15-0	WLCK	W	0h	Value = 0-FFFFh Bit n denotes lock bit for PrivID n. Writing WLCK[n] bit to 0 has no effect. Writing WLCK[n] bit to 1 along with setting MGCID key = 2CD2h locks the SES MPAX registers for PrivID n.

Table 7-83. Register Call Summary for MSMC_SES_MPAX_LCK

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • MSMC Register Access Control: [0]
MSMC Configuration Write Lock Registers • MSMC_SES_MPAX_LCK Register (Offset = 74h) [reset = 2CD20000h]: [0]

7.1.4.5.8 MSMC_SES_MPAX_ULCK Register (Offset = 78h) [reset = 2CD20000h]

The MSMC_SES_MPAX_ULCK is shown in Figure 7-29 and described in Table 7-85.

Table 7-84. MSMC_SES_MPAX_ULCK Instances

Instance	Physical Address
MSMC	0BC0 0078h

Figure 7-29. MSMC_SES_MPAX_ULCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MGCID																WEN															
W-2CD2h																W-0h															

LEGEND: W = Write Only; *-n* = value after reset

Table 7-85. MSMC_SES_MPAX_ULCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MGCID	W	2CD2h	Value = 2CD2h Writing this key value along with setting WEN[n] to 1 unlocks the SES MPAX registers for PrivID n.
15-0	WEN	W	0h	Value = 0-FFFFh Bit n denotes unlock bit for PrivID n. Writing WEN[n] bit to 0 has no effect. Writing WEN[n] bit to 1 along with setting MGCID key = 2CD2h unlocks the SES MPAX registers for PrivID n.

Table 7-86. Register Call Summary for MSMC_SES_MPAX_ULCK

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • MSMC Register Access Control: [0]
MSMC Configuration Write Lock Registers • MSMC_SES_MPAX_ULCK Register (Offset = 78h) [reset = 2CD20000h]: [0]

7.1.4.5.9 MSMC_SES_MPAX_LCKSTAT Register (Offset = 7Ch) [reset = 0h]

The MSMC_SES_MPAX_LCKSTAT is shown in Figure 7-30 and described in Table 7-88.

Table 7-87. MSMC_SES_MPAX_LCKSTAT Instances

Instance	Physical Address
MSMC	0BC0 007Ch

Figure 7-30. MSMC_SES_MPAX_LCKSTAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																WSTAT															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 7-88. MSMC_SES_MPAX_LCKSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reads return 0 and writes have no effect.
15-0	WSTAT	R	0h	• Value = 0-FFFFh • Bit n indicates the lock's current status for PrivID n. • 0 - SES MPAX registers are unlocked. • 1 - SES MPAX registers are locked.

Table 7-89. Register Call Summary for MSMC_SES_MPAX_LCKSTAT

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • MSMC Register Access Control: [0]
MSMC Configuration Write Lock Registers • MSMC_SES_MPAX_LCKSTAT Register (Offset = 7Ch) [reset = 0h]: [0]

7.1.4.6 MSMC Interrupt Control Registers

7.1.4.6.1 MSMC_SMESTAT Register (Offset = 80h) [reset = 0h]

The MSMC_SMESTAT register provides the status of those interrupts that are enabled; that is, a bitfield in the MSMC_SMESTAT register is set if the associated interrupt is both enabled and has occurred.

The MSMC_SMESTAT is shown in Figure 7-31 and described in Table 7-91.

Table 7-90. MSMC_SMESTAT Instances

Instance	Physical Address
MSMC	0BC0 0080h

Figure 7-31. MSMC_SMESTAT Register

31	30	29	28	27	26	25	24
PFESTAT							
R-0h							
23	22	21	20	19	18	17	16
PFESTAT							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CEES	NCEES	CSES	NCSES
R-0h				R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 7-91. MSMC_SMESTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	PFESTAT	R	0h	• Value = 0-FFFFh • Bit n denotes the protection fault interrupt is enabled and pending for PrivID n.
15-4	RESERVED	R	0h	• Reads return 0 and writes have no effect.
3	CEES	R	0h	• 0 = No EDC error. • 1 = Correctable EDC error interrupt is enabled and pending.
2	NCEES	R	0h	• 0 = No EDC error. • 1 = Non-correctable EDC error interrupt is enabled and pending.
1	CSES	R	0h	• 0 = No scrubbing error. • 1 = Correctable scrubbing error interrupt is enabled and pending.
0	NCSES	R	0h	• 0 - No scrubbing error. • 1 = Non-correctable scrubbing error interrupt is enabled and pending.

Table 7-92. Register Call Summary for MSMC_SMESTAT

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • MSMC Interrupts: [0]
MSMC Interrupt Control Registers • MSMC_SMESTAT Register (Offset = 80h) [reset = 0h]: [0][1][2]

7.1.4.6.2 MSMC_SMIRSTAT Register (Offset = 84h) [reset = 0h]

The MSMC_SMIRSTAT register stores the raw interrupt status for each interrupt line, a bitfield is set if the associated interrupt has occurred. An interrupt can be generated and its raw status recorded in MSMC_SMIRSTAT either when the event occurs in hardware (hardware interrupt) or when software writes to the associated bitfield in the MSMC_SMIRSTAT register (software interrupt for simulating the event).

The MSMC_SMIRSTAT is shown in Figure 7-32 and described in Table 7-94.

Table 7-93. MSMC_SMIRSTAT Instances

Instance	Physical Address
MSMC	0BC0 0084h

Figure 7-32. MSMC_SMIRSTAT Register

31	30	29	28	27	26	25	24
PFISTAT							
RW-0h							
23	22	21	20	19	18	17	16
PFISTAT							
RW-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CEI	NCEI	CSI	NCSI
R-0h				RW-0h	RW-0h	RW-0h	RW-0h

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-94. MSMC_SMIRSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	PFISTAT	RW	0h	• Value = 0-FFFFh • Bit n denotes a protection fault for PrivID n. • Write 1 to bit n sets the protection fault interrupt status for PrivID n.
15-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3	CEI	RW	0h	• 0 = No EDC error. • 1 - Correctable EDC error. Write 1 to set the correctable EDC error interrupt status.
2	NCEI	RW	0h	• 0 = No EDC error. • 1 = Non-correctable EDC error. Write 1 to set the non-correctable EDC error interrupt status.
1	CSI	RW	0h	• 0 = No scrubbing error. • 1 = Correctable scrubbing error. Write 1 to set the correctable scrubbing error interrupt status.
0	NCSI	RW	0h	• 0 = No scrubbing error. • 1 = Non-correctable scrubbing error. Write 1 to set the non-correctable scrubbing error interrupt status.

Table 7-95. Register Call Summary for MSMC_SMIRSTAT

MSMC Registers • MSMC Registers: [0][1]
MSMC Functional Description • Error Correction Mode: [0][1][2][3] • MSMC Interrupts: [0][1][2][3][4][5][6][7][8] • Memory Protection Fault Reporting: [0]
MSMC Interrupt Control Registers • MSMC_SMIRSTAT Register (Offset = 84h) [reset = 0h]: [0][1][2][3]

7.1.4.6.3 MSMC_SMIRC Register (Offset = 88h) [reset = 0h]

The MSMC_SMIRC is shown in Figure 7-33 and described in Table 7-97.

Table 7-96. MSMC_SMIRC Instances

Instance	Physical Address
MSMC	0BC0 0088h

Figure 7-33. MSMC_SMIRC Register

31	30	29	28	27	26	25	24
PFIC							
W-0h							
23	22	21	20	19	18	17	16
PFIC							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CEC	NCEC	CSC	NCSC
R-0h				W-0h	W-0h	W-0h	W-0h

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 7-97. MSMC_SMIRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	PFIC	W	0h	• Value = 0-FFFFh • Write 1 to bit n clears the protection fault interrupt status for PrivID n.
15-4	RESERVED	R	0h	• Reads return 0 and writes have no effect.
3	CEC	W	0h	• 0 = No effect. • 1 = Clear the correctable EDC error interrupt status.
2	NCEC	W	0h	• 0 = No effect. • 1 = Clear the non-correctable EDC error interrupt status.
1	CSC	W	0h	• 0 = No effect. • 1 = Clear the correctable scrubbing error interrupt status.
0	NCSC	W	0h	• 0 = No effect. • 1 = Clear the non-correctable scrubbing error interrupt status.

Table 7-98. Register Call Summary for MSMC_SMIRC

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • Scrubbing Error Logging and Statistics Collection: [0] • EDC Error Reporting: [0] • MSMC Interrupts: [0][1][2][3] • Memory Protection Fault Reporting: [0][1]
MSMC Interrupt Control Registers • MSMC_SMIRC Register (Offset = 88h) [reset = 0h]: [0]

7.1.4.6.4 MSMC_SMIESTAT Register (Offset = 8Ch) [reset = 0h]

The MSMC_SMIESTAT is shown in Figure 7-34 and described in Table 7-100.

Table 7-99. MSMC_SMIESTAT Instances

Instance	Physical Address
MSMC	0BC0 008Ch

Figure 7-34. MSMC_SMIESTAT Register

31	30	29	28	27	26	25	24
PFIESTAT							
RW-0h							
23	22	21	20	19	18	17	16
PFIESTAT							
RW-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CEIE	NCEIE	CSIE	NCSIE
R-0h				RW-0h	RW-0h	RW-0h	RW-0h

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-100. MSMC_SMIESTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	PFIESTAT	RW	0h	• Value = 0-FFFFh • Bit n denotes the protection fault interrupt is enabled for PrivID n.
15-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3	CEIE	RW	0h	• 0 = Correctable EDC error interrupt is disabled. • 1 = Correctable EDC error interrupt is enabled.
2	NCEIE	RW	0h	• 0 = Non-correctable EDC error interrupt is disabled. • 1 = Non-correctable EDC error interrupt is enabled.
1	CSIE	RW	0h	• 0 = Correctable scrubbing error interrupt is disabled. • 1 = Correctable scrubbing error interrupt is enabled.
0	NCSIE	RW	0h	• 0 = Non-correctable scrubbing error interrupt is disabled. • 1 = Non-correctable scrubbing error interrupt is enabled.

Table 7-101. Register Call Summary for MSMC_SMIESTAT

MSMC Registers • MSMC Registers: [0][1]
MSMC Functional Description • Scrubbing Error Logging and Statistics Collection: [0][1] • EDC Error Reporting: [0][1] • MSMC Interrupts: [0][1][2][3][4] • Memory Protection Fault Reporting: [0]
MSMC Interrupt Control Registers • MSMC_SMIESTAT Register (Offset = 8Ch) [reset = 0h]: [0]

7.1.4.6.5 MSMC_SMIEC Register (Offset = 90h) [reset = 0h]

The MSMC_SMIEC is shown in Figure 7-35 and described in Table 7-103.

Table 7-102. MSMC_SMIEC Instances

Instance	Physical Address
MSMC	0BC0 0090h

Figure 7-35. MSMC_SMIEC Register

31	30	29	28	27	26	25	24
PFIEC							
W-0h							
23	22	21	20	19	18	17	16
PFIEC							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CEEC	NCEEC	CSEC	NCSEC
R-0h				W-0h	W-0h	W-0h	W-0h

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 7-103. MSMC_SMIEC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	PFIEC	W	0h	• Value = 0-FFFFh • Write 1 to bit n clears the protection fault interrupt enable for PrivID n.
15-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3	CEEC	W	0h	• 0 = No effect. • 1 = Clear the correctable EDC error interrupt enable (disable the interrupt).
2	NCEEC	W	0h	• 0 = No effect. • 1 = Clear the non-correctable EDC error interrupt enable (disable the interrupt).
1	CSEC	W	0h	• 0 = No effect. • 1 = Clear the correctable scrubbing error interrupt enable (disable the interrupt).
0	NCSEC	W	0h	• 0 = No effect. • 1 = Clear the non-correctable scrubbing error interrupt enable (disable the interrupt).

Table 7-104. Register Call Summary for MSMC_SMIEC

MSMC Registers • MSMC Registers: [0]
MSMC Functional Description • MSMC Interrupts: [0]
MSMC Interrupt Control Registers • MSMC_SMIEC Register (Offset = 90h) [reset = 0h]: [0]

7.1.4.7 MPAX Segment Registers

7.1.4.7.1 MSMC_SMS_MPAXL_x_y Register (Offset = 200h + z*8) [reset = C00000h]

The MSMC_SMS_MPAXL_x_y is shown in Figure 7-36 and described in Table 7-106.
x = 0 to 15; y = 0 to 7; z = 0 to 127
(x;y;z) = (0;0;0, 0;1;1, 0;2;2, ... 15;5;125, 15;6;126, 15;7;127)

Table 7-105. MSMC_SMS_MPAXL_x_y Instances

Instance	Physical Address
MSMC	0BC0 0200h + (z*8)

Figure 7-36. MSMC_SMS_MPAXL_x_y Register

31	30	29	28	27	26	25	24
RESERVED				CONSTANT			
R-0h				R-Ch			
23	22	21	20	19	18	17	16
CONSTANT				RADDR			
R-Ch				RW-0h			
15	14	13	12	11	10	9	8
RADDR							
RW-0h							
7	6	5	4	3	2	1	0
RESERVED		SR	SW	SX	UR	UW	UX
RW-0h		RW-0h	RW-0h	RW-0h	RW-0h	RW-0h	RW-0h

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 7-106. MSMC_SMS_MPAXL_x_y Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reads return 0 and writes have no effect.
27-20	CONSTANT	R	Ch	0Ch = Constant value
19-8	RADDR	RW	0h	0-FFFh = Replacement Address — Bits that replace and extend the upper address bits matched by BADDR.
7-6	RESERVED	RW	0h	Reads return 0 and writes have no effect.
5	SR	RW	0h	Supervisor read access type. • 0 = Normal operation. • 1 = Indicates a supervisor read request.
4	SW	RW	0h	Supervisor write access type. • 0 = Normal operation. • 1 = Indicates a supervisor write request.
3	SX	RW	0h	Supervisor execute access type. • 0 = Normal operation. • 1 = Indicates a supervisor execute request.
2	UR	RW	0h	User read access type. • 0 = Normal operation. • 1 = Indicates a user read request.
1	UW	RW	0h	User write access type. • 0 = Normal operation. • 1 = Indicates a user write request.
0	UX	RW	0h	User execute access type. • 0 = Normal operation. • 1 - Indicates a user execute request.

Table 7-107. Register Call Summary for MSMC_SMS_MPAXL_x_y

MSMC Registers • MSMC Registers: [0]
MPAX Segment Registers • MSMC_SMS_MPAXL_x_y Register (Offset = 200h + z*8) [reset = C00000h]: [0]

7.1.4.7.2 MSMC_SMS_MPAXH_x_y Register (Offset = 204h + z*8) [reset = C000000h]

The MSMC_SMS_MPAXH_x_y is shown in Figure 7-37 and described in Table 7-109.
x = 0 to 15; y = 0 to 7; z = 0 to 127
(x;y;z) = (0;0;0, 0;1;1, 0;2;2, ... 15;5;125, 15;6;126, 15;7;127)

Table 7-108. MSMC_SMS_MPAXH_x_y Instances

Instance	Physical Address
MSMC	0BC0 0204h + (z*8)

Figure 7-37. MSMC_SMS_MPAXH_x_y Register

31	30	29	28	27	26	25	24
CONSTANT							
R-Ch							
23	22	21	20	19	18	17	16
BADDR							
RW-0h							
15	14	13	12	11	10	9	8
BADDR				RESERVED			
RW-0h				R-0h			
7	6	5	4	3	2	1	0
US	RESERVED			SEGSZ			
RW-0h	R-0h			RW-0h			

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-109. MSMC_SMS_MPAXH_x_y Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	CONSTANT	R	Ch	0Ch = Constant value
23-12	BADDR	RW	0h	Value = 0-FFFh Base Address — The Segment Base Address field defines the placement of the controlled segment. This field is used to match against the incoming address on the system slave port to identify the addressed segment. For SMS and SES, MPAXH[31-12] is compared against the requested address.
11-8	RESERVED	R	0h	Reads return 0 and writes have no effect.
7	US	RW	0h	0 = This memory page is shared and coherence actions will be spawned on access to this page 1 = This memory page is not shared and coherence actions will not be spawned on access to this page
6-5	RESERVED	R	0h	Reads return 0 and writes have no effect.
4-0	SEGSZ	RW	0h	Value = 0-1Fh Segment Size — The Segment Size field defines the size of the controlled segment. Refer to Table 7-5 for segment size encoding details.

Table 7-110. Register Call Summary for MSMC_SMS_MPAXH_x_y

MSMC Registers • MSMC Registers: [0]
MPAX Segment Registers • MSMC_SMS_MPAXH_x_y Register (Offset = 204h + z*8) [reset = C000000h]: [0]

7.1.4.7.3 MSMC_SES_MPAXL_x_y Register (Offset = 600h + z*8) [reset = 0h]

The MSMC_SES_MPAXL_x_y is shown in Figure 7-38 and described in Table 7-112.
x = 0 to 15; y = 0 to 7; z = 0 to 127
(x;y;z) = (0;0;0, 0;1;1, 0;2;2, ... 15;5;125, 15;6;126, 15;7;127)

Table 7-111. MSMC_SES_MPAXL_x_y Instances

Instance	Physical Address
MSMC	0BC0 0600h + (z*8)

Figure 7-38. MSMC_SES_MPAXL_x_y Register

31	30	29	28	27	26	25	24
RADDR							
RW-0h							
23	22	21	20	19	18	17	16
RADDR							
RW-0h							
15	14	13	12	11	10	9	8
RADDR							
RW-0h							
7	6	5	4	3	2	1	0
RESERVED		SR	SW	SX	UR	UW	UX
RW-0h		RW-0h	RW-0h	RW-0h	RW-0h	RW-0h	RW-0h

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-112. MSMC_SES_MPAXL_x_y Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RADDR	RW	0h	Value = 0-FF FFFFh Replacement Address—Bits that replace and extend the upper address bits matched by BADDR.
7-6	RESERVED	RW	0h	Reads return 0 and writes have no effect.
5	SR	RW	0h	Supervisor read access type. • 0 = Normal operation. • 1 = Indicates a supervisor read request.
4	SW	RW	0h	Supervisor write access type. • 0 = Normal operation. • 1 = Indicates a supervisor write request.
3	SX	RW	0h	Supervisor execute access type. • 0 = Normal operation. • 1 = Indicates a supervisor execute request.
2	UR	RW	0h	User read access type. • 0 = Normal operation. • 1 = Indicates a user read request.
1	UW	RW	0h	User write access type. • 0 = Normal operation. • 1 = Indicates a user write request.
0	UX	RW	0h	User execute access type. • 0 = Normal operation. • 1 = Indicates a user execute request.

Table 7-113. Register Call Summary for MSMC_SES_MPAXL_x_y

MSMC Registers • MSMC Registers: [0]
MPAX Segment Registers • MSMC_SES_MPAXL_x_y Register (Offset = 600h + z*8) [reset = 0h]: [0]

7.1.4.7.4 MSMC_SES_MPAXH_x_y Register (Offset = 604h + z*8) [reset = 0h]

The MSMC_SES_MPAXH_x_y is shown in Figure 7-39 and described in Table 7-115.
x = 0 to 15; y = 0 to 7; z = 0 to 127
(x;y;z) = (0;0;0, 0;1;1, 0;2;2, ... 15;5;125, 15;6;126, 15;7;127)

Table 7-114. MSMC_SES_MPAXH_x_y Instances

Instance	Physical Address
MSMC	0BC0 0604h + (z*8)

Figure 7-39. MSMC_SES_MPAXH_x_y Register

31	30	29	28	27	26	25	24
BADDR							
RW-0h							
23	22	21	20	19	18	17	16
BADDR							
RW-0h							
15	14	13	12	11	10	9	8
BADDR				RESERVED			
RW-0h				R-0h			
7	6	5	4	3	2	1	0
US	RESERVED			SEGSZ			
RW-0h	R-0h			RW-0h			

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-115. MSMC_SES_MPAXH_x_y Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	BADDR	RW	0h	Value = 0-F FFFFh Base Address — The Segment Base Address field defines the placement of the controlled segment. This field is used to match against the incoming address on the system slave port to identify the addressed segment. For SMS and SES, MPAXH[31-12] is compared against the requested address.
11-8	RESERVED	R	0h	Reads return 0 and writes have no effect.
7	US	RW	0h	0 = This memory page is shared and coherence actions will be spawned on access to this page 1 = This memory page is not shared and coherence actions will not be spawned on access to this page
6-5	RESERVED	R	0h	Reads return 0 and writes have no effect.
4-0	SEGSZ	RW	0h	Value = 0-1Fh Segment Size — The Segment Size field defines the size of the controlled segment. Refer to Table 7-5 for segment size encoding details.

Table 7-116. Register Call Summary for MSMC_SES_MPAXH_x_y

MSMC Registers • MSMC Registers: [0]
MPAX Segment Registers • MSMC_SES_MPAXH_x_y Register (Offset = 604h + z*8) [reset = 0h]: [0]

7.2 DDR External Memory Interface (EMIF)

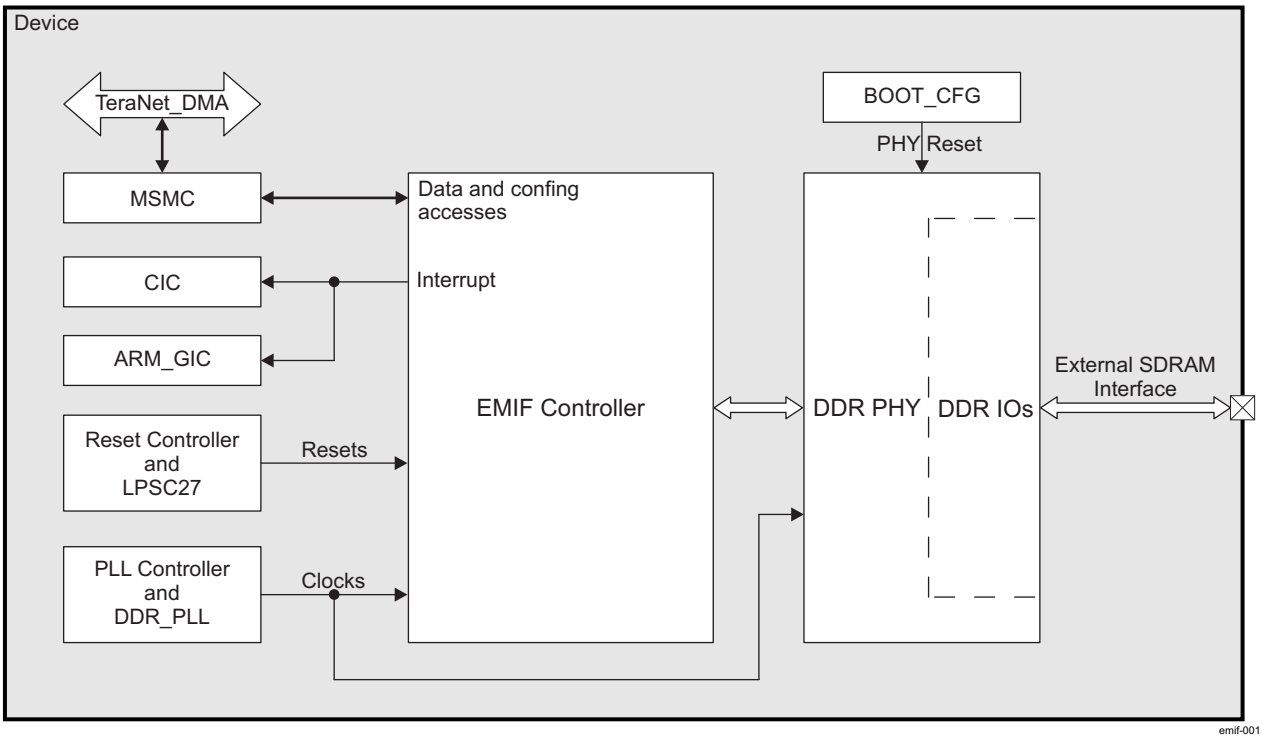
This section describes the DDR External Memory Interface (EMIF) for the device.

NOTE: The supported set of features and peripherals is device part number dependent. For more information, see the device Data Manual.

7.2.1 EMIF Overview

There is one EMIF controller in this device. It is used to provide an interface to external DDR3L SDRAM devices which can be utilized for storing program or data. Other memory types are not supported. The EMIF controller is accessed via the MSMC, and not directly through TeraNet. Figure 7-40 shows an overview of the EMIF controller and also connections to the other surrounding modules. The DDR PHY and DDR IOs are also used when performing data exchanges to and from external SDRAM.

Figure 7-40. EMIF Overview



The EMIF controller supports:

- Devices compliant to JEDEC JESD79-3F and JESD79-3-1 (DDR3L addendum) standards
- 16-bit and 32-bit SDRAM data bus width without ECC:
  - 16-bit: 2x8-bit SDRAMs
  - 16-bit: 1x16-bit SDRAM
  - 32-bit: 4x8-bit SDRAMs
  - 32-bit: 2x16-bit SDRAMs
- 32-bit SDRAM data bus width with 4-bit ECC:
  - 36-bit: 5x8-bit SDRAMs (32 bits for data and 4 bits for ECC, not used bits should be tied-off)
  - 36-bit: 3x16-bit SDRAMs (32 bits for data and 4 bits for ECC, not used bits should be tied-off)
- CAS latencies of 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 and 16
- CAS write latencies of 5, 6, 7, 8, 9, 10, 11, and 12
- 1, 2, 4, and 8 internal banks
- Burst length of 8
- Sequential burst type
- 8GB address space available over two chip selects
- 33-bit system address for address space of 8GB
- Page sizes with 256, 512, 1024, and 2048 words
- Self-refresh mode
- Power-down mode
- Output impedance calibration
- On-Die Termination (ODT)
- Prioritized refresh scheduling
- Programmable SDRAM refresh rate and backlog counter
- Programmable SDRAM timing parameters
- Only little endian mode
- ECC on SDRAM data bus:
  - 8-bit ECC per 64-bit data quanta without additional cycle latency
  - 1-bit correction and 2-bit detection
  - Statistics for 1-bit ECC and 2-bit ECC errors
  - Programmable address ranges to define ECC protected region
  - ECC calculated and stored on all writes to ECC protected address region
  - ECC verified on all reads to ECC protected address region
  - Two ECC modes supported:
    - Read-Modify-Write (RMW) ECC enabled to support sub quanta accesses to the ECC space.
    - RMW ECC disabled
- Class of service
- UDIMM address mirroring.

The EMIF controller does not support:

- Any memory types except DDR3L
- RDIMMs
- ECC for 16-bit mode
- Single ended DQS
- Mixed 8-bit and 16-bit SDRAM configurations
- 4-bit SDRAMs.

7.2.2 EMIF Environment

This section describes the external connections of the EMIF.

Figure 7-41 shows an example configuration of the EMIF for connection to 2x16-bit DDR3L memories per chip select without ECC.

Figure 7-42 shows an example configuration of the EMIF for connection to 3x16-bit DDR3L memories per chip select with ECC. In this case two 16-bit memories are used for data and one 16-bit (only 4 bits) memory is used for ECC.

For simplification the DDR PHY and DDR IOs are not shown. Only the IO signals for connection to external SDRAM are shown.

Figure 7-41. Example EMIF Configuration without ECC

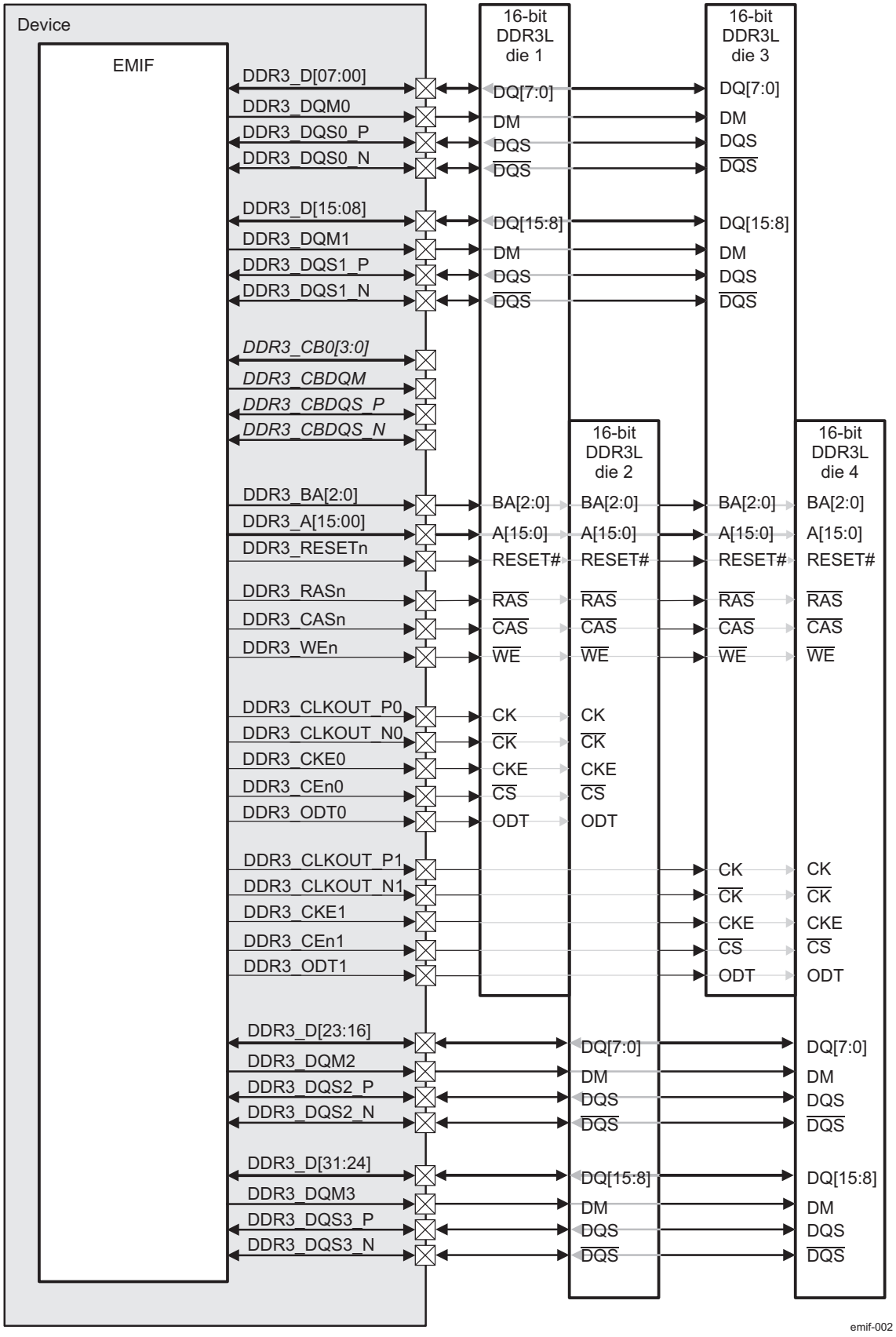


Figure 7-42. Example EMIF Configuration with ECC

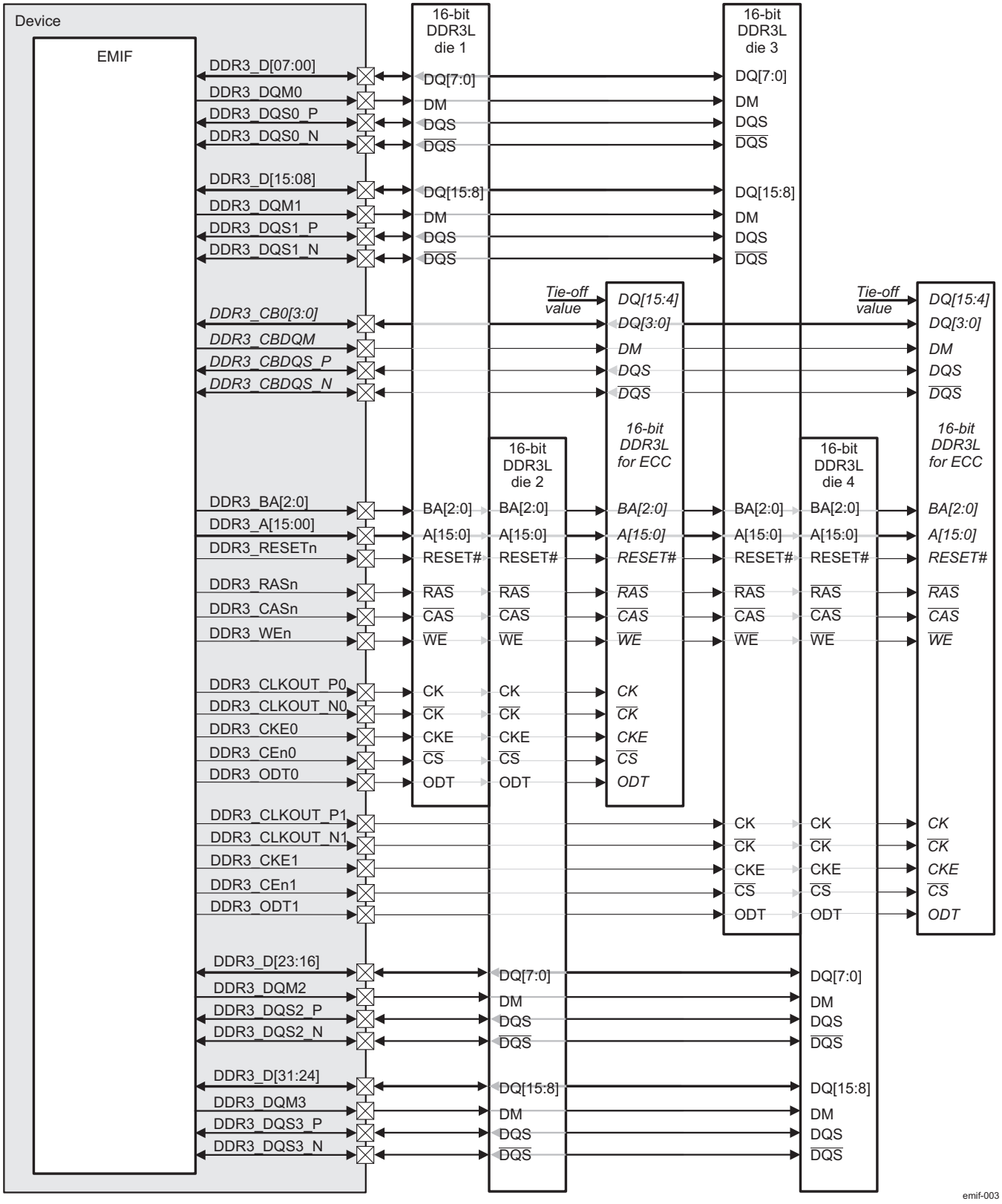


Table 7-117 describes the EMIF associated IO signals used for connection to DDR3L memories.

Table 7-117. EMIF IO signals

Signal Name	Direction	Description
Data Signals		
DDR3_D[31:00]	IOZ	Data bus
DDR3_DQM[3:0]	OZ	Data mask
DDR3_DQS[3:0]_P	IOZ	Data strobe
DDR3_DQS[3:0]_N	IOZ	Data strobe invert
DDR3_CB0[3:0]	*IOZ*	*Data bus used for ECC*
DDR3_CBDQM	*OZ*	*Data mask used for ECC*
DDR3_CBDQS_P	*IOZ*	*Data strobe used for ECC*
DDR3_CBDQS_N	*IOZ*	*Data strobe invert used for ECC*
Command and Control Signals		
DDR3_CEn0	OZ	Chip enable 0
DDR3_CEn1	OZ	Chip enable 1
DDR3_BA[2:0]	OZ	Bank address
DDR3_A[15:00]	OZ	Address bus
DDR3_CASn	OZ	Column address strobe
DDR3_RASn	OZ	Row address strobe
DDR3_WEn	OZ	Write enable
DDR3_CKE0	OZ	Clock enable 0
DDR3_CKE1	OZ	Clock enable 1
DDR3_CLKOUT_P0	OZ	Differential clock pair 0 used to drive the external SDRAM
DDR3_CLKOUT_N0	OZ	
DDR3_CLKOUT_P1	OZ	Differential clock pair 1 used to drive the external SDRAM
DDR3_CLKOUT_N1	OZ	
DDR3_ODT0	OZ	On Die Termination Output 0 used to set termination on the SDRAM
DDR3_ODT1	OZ	On Die Termination Output 1 used to set termination on the SDRAM
DDR3_RESETn	OZ	DDR3 reset signal

7.2.3 EMIF Integration

This section describes the EMIF integration in the device and includes information about clocks, resets, and hardware requests. Figure 7-43 shows the EMIF integration.

Figure 7-43. EMIF Integration

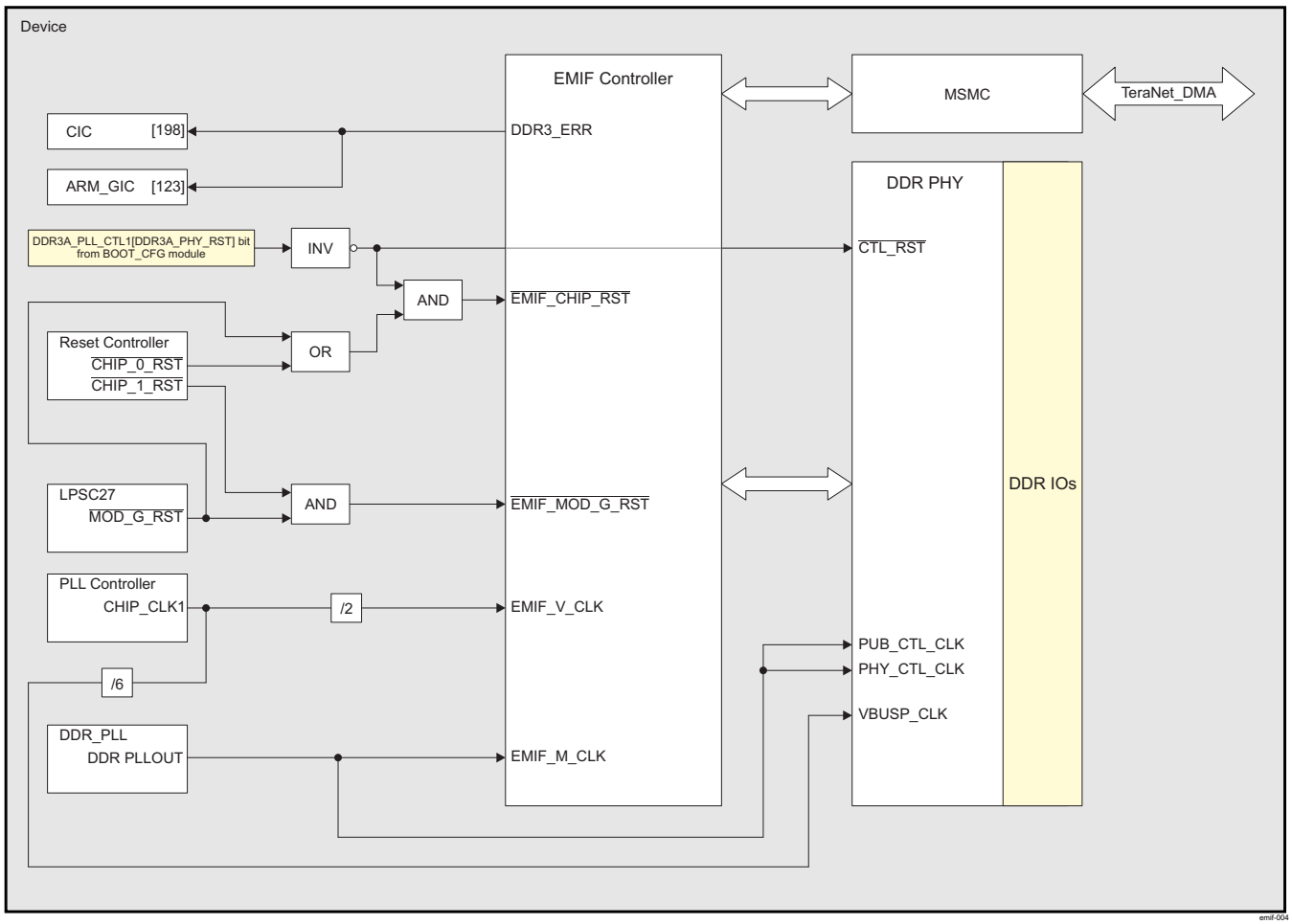


Table 7-118 through Table 7-120 summarize the integration of the EMIF module in the device.

Table 7-118. EMIF Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
EMIF	PD15	LPSC27	Accessed via MSMC, and not directly through TeraNet_DMA

Table 7-119. EMIF Clocks and Resets

Clocks				
Module Instance	Destination Signal Name	Source Signal Name	Source	Description

Table 7-119. EMIF Clocks and Resets (continued)

EMIF	EMIF_V_CLK	CHIP_CLK1/2	PLL Controller	Interface clock for EMIF. It is used for driving the interface logic for connection to the MSMC.
	EMIF_M_CLK	DDR PLLOUT	DDR_PLL	Functional and interface clock for EMIF. This clock runs at half the DDR3L clock rate and drives the EMIF state machine and configuration registers.
DDR PHY	PUB_CTL_CLK	DDR PLLOUT	DDR_PLL	Clocks for the DDR PHY with frequency and phase same as EMIF_M_CLK
	PHY_CTL_CLK	DDR PLLOUT	DDR_PLL	
	VBUSP_CLK	CHIP_CLK1/6	PLL Controller	Interface clock for the DDR PHY. This clock drives the PHY configuration registers.
Resets				
Module Instance	**Destination Signal Name**	**Source Signal Name**	**Source**	**Description**
EMIF	EMIF_CHIP_RST	MOD_G_RST	LPSC27	Resets the EMIF state machine, FIFOs and all EMIF configuration registers. The EMIF_CHIP_RST can be activated when: • Both LPSC27.MOD_G_RST and CHIP_0_RST are activated. • The DDR3A_PLL_CTL1[31] DDR3A_PHY_RST bit is set to 1h Access to the EMIF configuration registers cannot be performed while the EMIF_CHIP_RST is asserted (active).
		CHIP_0_RST	Reset Controller	
		DDR3A_PLL_CTL1[31] DDR3A_PHY_RST	BOOT_CFG	
	EMIF_MOD_G_RST	MOD_G_RST	LPSC27	Resets the EMIF state machine, FIFOs and only the interrupt registers. The EMIF_MOD_G_RST can be activated when: • LPSC27.MOD_G_RST is activated. • CHIP_1_RST is activated. Access to the EMIF configuration registers cannot be performed while the EMIF_MOD_G_RST is asserted (active).
		CHIP_1_RST	Reset Controller	
DDR PHY	CTL_RST	DDR3A_PLL_CTL1[31] DDR3A_PHY_RST	BOOT_CFG	Reset to the DDR PHY. The DDR PHY is reset when the DDR3A_PLL_CTL1[31] DDR3A_PHY_RST bit is set to 1h.

Table 7-120. EMIF Hardware Requests

Interrupt Requests				
Module Instance	**Event Name**	**Mapped To Input Event [Number]**		**Description**
		ARM GIC	**CIC**	
EMIF	DDR3_ERR	[123]	[198]	EMIF error interrupt

NOTE: For more information about interrupts, see Section 7.2.4.10.

7.2.4 EMIF Functional Description

7.2.4.1 Block Diagram

To move data efficiently from on-chip resources to an external SDRAM device, the EMIF controller makes use of:

- One command FIFO
- One write data FIFO
- One write status FIFO
- One read command FIFO
- SDRAM read data FIFO
- Register read data FIFO
- Command scheduler
- Data scheduler.

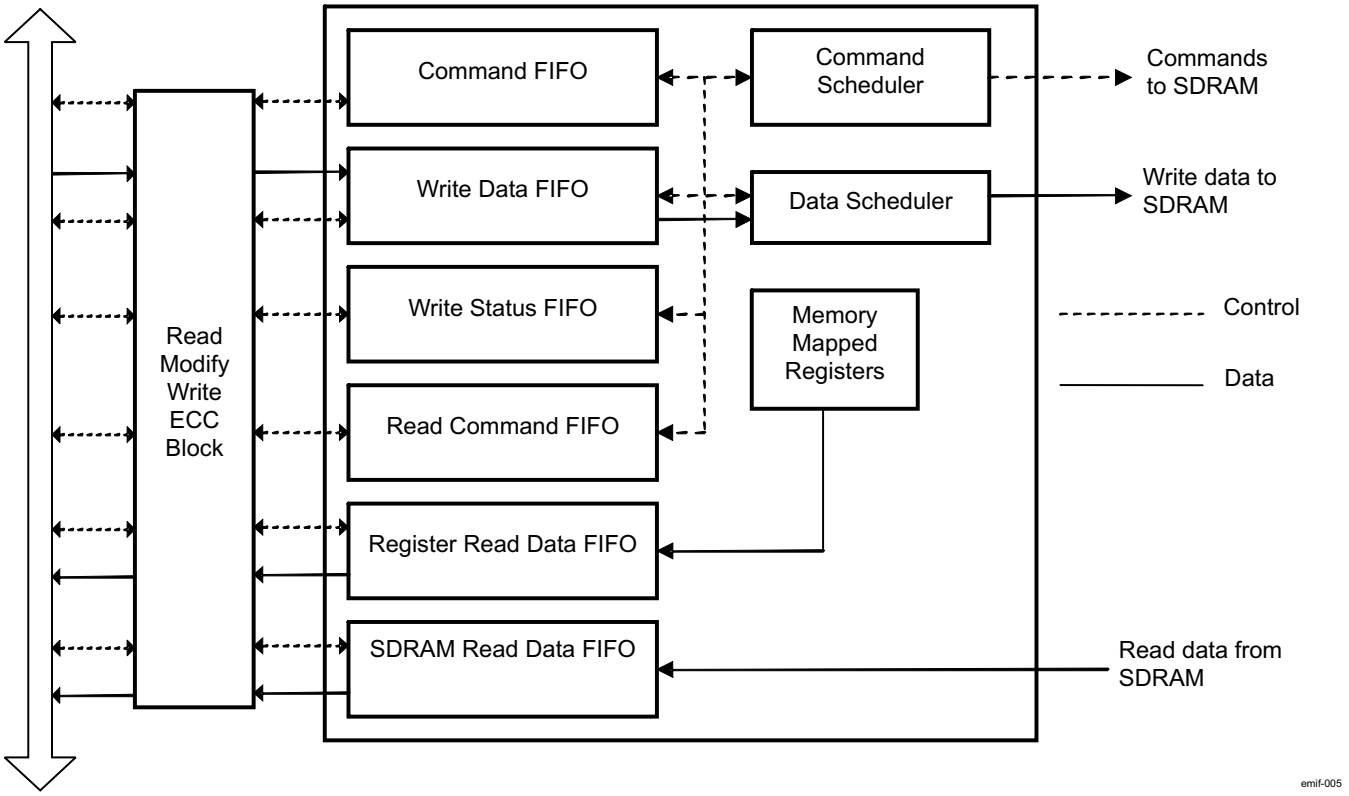
Table 7-121 describes the purpose of each FIFO.

Figure 7-44 shows a block diagram of the EMIF controller FIFOs. Commands, write data, and read data arrive at the EMIF controller parallel to each other. The same bus is used to write and read data to and from external SDRAM as well as EMIF configuration registers.

Table 7-121. EMIF controller FIFOs Description

FIFO	Description	Depth
Command	Stores all commands coming from on-chip requestors	16
Write Data	Stores write data coming from on-chip requestors to memory	20 (512-bit wide)
Write Status	Stores the write status information for each write transaction	7
Read Command	Stores all read transactions that have to be issued to on-chip requestors	28
SDRAM Read Data	Stores read data coming from SDRAM memory to on-chip requestors	28 (256-bit wide)
Register Read Data	Stores read data coming from EMIF configuration registers to on-chip requestors	2 (256-bit wide)

Figure 7-44. Block Diagram of EMIF FIFOs



7.2.4.1.1 Reordering of Commands in the Command FIFO

The EMIF controller performs command reordering and scheduling to achieve efficient transfers with maximum throughput. The goal is to maximize the usage of the data, address, and command buses while hiding the overhead of opening and closing SDRAM rows. Command reordering takes place within the command FIFO.

The EMIF controller examines all the commands stored in the command FIFO to schedule commands to the external memory. For each master, EMIF reorders the commands based on the following rules:

- The EMIF controller pushes a read command before an older write command from the same master if the read is to different address block (2048 bytes) and the read priority is equal to or greater than the write priority.
- The EMIF controller blocks a read command regardless of the master or priority if that read command is to the same address block (2048 bytes) as an older write command.

Thus, one pending read, write or both might exist for each master. Among all pending reads, the EMIF controller selects all reads that have their corresponding SDRAM banks already open. Among all pending writes, the EMIF controller selects all writes that have their corresponding SDRAM banks already open.

As a result of this command reordering several pending reads and writes may exist that have their corresponding banks open. The highest priority read is selected from the pending reads, and the highest priority write from the pending writes. If two or more commands have highest priority, the oldest command is selected. As a result, EMIF might have a final read and a final write command. It chooses either the read or the write command depending on the values programmed in the EMIF_RWTHRESH[12-8] WR_THRSH and EMIF_RWTHRESH[4-0] RD_THRSH fields. EMIF executes read commands until the read threshold is reached and then switches to write commands. EMIF executes write commands until the write threshold is reached and then it switches again to read commands.

The EMIF controller supports interleaving of commands for maximum efficiency. This means that the controller partially executes one command and switches to another higher priority command before finishing the first command.

Apart from reads and writes the EMIF controller also opens and closes SDRAM banks and maintains the refresh counts for an SDRAM. The priority of the SDRAM commands with respect to refresh levels is as follows:

1. (Highest priority) SDRAM refresh request due to Refresh Must level of refresh urgency reached.
2. Read request without a higher priority write (from the reordering algorithm already described).
3. Write request.
4. SDRAM Activate commands.
5. SDRAM Deactivate commands.
6. SDRAM Power-Down request.
7. SDRAM refresh request due to Refresh May or Release level of refresh urgency reached.
8. (Lowest priority) SDRAM self-refresh request.

7.2.4.1.2 Command Starvation

While performing the scheduling algorithm described in Section 7.2.4.1.1 the EMIF controller is subject to the following:

- A continuous stream of high priority commands can block lower priority commands.
- A continuous stream of SDRAM commands to a row in an open bank can block commands to another row in the same bank.

To avoid continuous blocking effect that can be caused by a continuous stream of high priority commands which thus block lower priority commands, the EMIF momentarily raises the priority of the oldest command over all other commands when the time for the oldest command configured through the EMIF_VBUSM_CONFIG[7-0] PR_OLD_COUNT bit field expires. In addition, the order of command accesses can also be adjusted by grouping commands in two categories or “classes” and assigning different latency expiration counters to each category. For more information see Section 7.2.4.1.4, *Class of Service of Commands in the Command FIFO*.

In addition, the highest priority condition is the rising edge of EMIF_CHIP_RST or EMIF_MOD_G_RST. If this occurs, the EMIF controller stops whatever it is currently doing and goes to idle state. In this case commands and data stored in the FIFOs are lost.

7.2.4.1.3 Possible Race Condition

A race condition may exist when certain masters write data to the EMIF controller. For example, if master A passes a software message via a buffer in the SDRAM and does not wait for indication that the write it complete, when master B attempts to read the software message it may read old data and therefore receive an incorrect message. In order to confirm that a write from master A has completed before a read from master B is performed, master A must wait for the write to complete before indicating to master B that the data is ready to be read. For example, an EDMA transfer controller should wait for the transfer completion event to occur before signaling a CPU to read the message from the SDRAM.

If master A does not wait for indication that a write is complete, it must perform the following:

1. Perform the required write.
2. Perform a dummy write to the EMIF_MIDR register.
3. Perform a dummy read from the EMIF_MIDR register.
4. Indicate to master B that the data is ready to be read after completion of the read in step 3. The completion of the read in step 3 ensures that the previous write has been done.

7.2.4.1.4 Class of Service of Commands in the Command FIFO

The commands in the Command FIFO can be mapped to two categories called also *classes of service*: 1 and 2. The mapping of commands to a particular class of service can be done based on priority or master ID. The mapping based on priority can be done by setting the appropriate values in the EMIF_PRI_COS_MAP register. The mapping based on master ID can be done by setting the appropriate values of master ID and masks in the EMIF_MSTID_COS_1_MAP and EMIF_MSTID_COS_2_MAP registers.

There are three master ID and mask values that can be set for each class of service. In conjunction with the masks each class of service can have a maximum of 144 master IDs mapped to it. For example, a master ID value of FFh along with a mask value of 3h maps all master IDs from F8h to FFh to that particular class of service. By default all commands are mapped to class of service 2.

Each class of service has an associated latency counter. The value of this counter can be set in the EMIF_VBUSM_CONFIG[23-16] COS_COUNT_1 and EMIF_VBUSM_CONFIG[15-8] COS_COUNT_2 fields. When the latency counter for a command expires, that is, reaches the value programmed for the class of service that the command belongs to, that command is executed next. If there is more than one command that has expired latency counters, the command with the highest priority is executed first. One exception to this rule is if the EMIF_VBUSM_CONFIG[7-0] PR_OLD_COUNT value expires for the oldest command in the queue. That command is executed first irrespective of priority or class of service. This is done to prevent the continuous blocking effect described in Section 7.2.4.1.2, *Command Starvation*.

7.2.4.2 Power Management

7.2.4.2.1 SDRAM Power-Down Mode

The EMIF controller supports SDRAM power-down mode. Automatic SDRAM power-down is enabled by setting the EMIF_PMCTL[10-8] LP_MODE field to 4h. The memory is put into power-down after the EMIF controller is idle for EMIF_PMCTL[15-12] PD_TIM number of DDR3CLKOUT cycles. In power-down mode, the EMIF controller does not stop the clocks to the memory. The controller maintains DDR3CKE low to maintain the power-down state. When the SDRAM is in power-down mode the EMIF controller services accesses to its configuration registers as usual.

If the SDRAM is in power-down mode and one of the following occurs, the EMIF brings the SDRAM out of power-down mode:

- The EMIF_PMCTL[10-8] LP_MODE field is set to value different than 4h.
- SDRAM access is requested.
- Refresh-must level is reached.

To exit power-down mode the EMIF does the following:

1. Drives DDR3CKE high after EMIF_SDTIM3[3-0] T_CKE + 1 cycles have elapsed since the power-down command has been issued.
2. Waits for EMIF_SDTIM3[30-28] T_XP + 1 cycles.
3. Enters its idle state and can issue any commands.

7.2.4.2.2 SDRAM Self-Refresh Mode

The EMIF controller supports self-refresh mode for low power. The controller maintains DDRCKE low to maintain the self-refresh state. In self-refresh the memory maintains valid data while consuming minimal amount of power.

Self-refresh mode is enabled by setting the EMIF_PMCTL[10-8] LP_MODE field to 2h. The EMIF controller automatically puts the SDRAM into self-refresh after the controller is idle for EMIF_PMCTL[7-4] SR_TIM number of DDR3CLKOUT cycles.

In self-refresh mode, the EMIF controller automatically stops the SDRAM clock (DDR3CLKOUT). The EMIF drives DDR3CKE low to maintain self-refresh mode. When the SDRAM is in self-refresh mode the EMIF controller services accesses to its configuration registers as usual.

If the SDRAM is in self-refresh mode and one of the following occurs, the EMIF brings the SDRAM out of selfrefresh mode:

- The EMIF_PMCTL[10-8] LP_MODE field is set to value different than 2h
- SDRAM access is requested
- The EMIF_PMCTL[7-4] SR_TIM field is cleared

To exit self-refresh the EMIF does the following:

1. Enables the SDRAM clock
2. Drives DDR3CKE high

3. Waits EMIF_SDTIM3[27-18] T_XSNR + 1 cycles
4. Starts a refresh cycle in the next cycle
5. Issues a ZQCL command, if the EMIF_ZQ_CONFIG[28] ZQ_SFEXITEN bit is set to 1
6. Enters its idle state and can issue any other command except write or read command. A write or a read command is issued only after EMIF_SDTIM3[17-8] T_XSRD + 1 cycles have elapsed since DDR3CKE has been driven high.

In a situation where memory accesses and a self-refresh command are sent to the EMIF controller, the controller always prioritizes the memory access. Thus, if a reset is triggered when memory accesses and a self-refresh command are queued in the controller, it is likely that self-refresh will not be entered.

The user must ensure that all memory accesses have been completed and verify that self-refresh is set by reading the EMIF_STATUS[27] SELF_REF bit before initiating a reset.

NOTE: The EMIF controller completes all pending memory accesses and refreshes before it puts SDRAM into self-refresh. If a request for a memory access is received, the EMIF controller services the memory access request then returns to the self-refresh state upon completion.

7.2.4.2.2.1 SDRAM Extended Temperature Range

The normal operating temperature range for DDR3 SDRAMs is typically 0 to 85°C. When operating in self-refresh mode within the extended temperature range (85°C to 95°C), the memory device must be refreshed at 2× the normal refresh rate. For this purpose either the auto self-refresh (bit DDR_PHY_MR2[6] ASR) or self-refresh temperature (bit DDR_PHY_MR2[7] SRT) feature should be used. Under normal operating conditions, both ASR and SRT should be disabled (equal to 0). If ASR is enabled, the internal refresh rate of the SDRAM automatically switches to 2× the refresh rate when the operating case temperature Tc is greater than 85°C in case of self-refresh mode. If SRT is enabled, the internal refresh rate of the SDRAM is forced to 2× the refresh rate regardless of the operating case temperature Tc. Both ASR and SRT cannot be enabled at the same time. One must be disabled if the other is enabled.

NOTE: ASR and SRT are used only in self-refresh mode (EMIF_PMCTL[10-8] LP_MODE = 2h). When operating in extended temperature range with LP_MODE = 0h (not in self-refresh), it is up to the user to program the manual refresh rate to 2× the normal refresh rate for proper operation.

If it is guaranteed that Tc will exceed 85°C, it is recommended to set the SRT bit to 1 to force the refresh rate to 2× regardless of the operating temperature.

7.2.4.3 SDRAM Refresh Scheduling

The EMIF controller uses two counters to schedule refresh (REF) commands:

- 13-bit decrementing refresh interval counter
- 4-bit refresh backlog counter.

The interval counter is loaded at reset with the value of the EMIF_SDRFC[15-0] REFRESH_RATE field. The interval counter decrements by one each cycle until it reaches zero. At this point it reloads from value EMIF_SDRFC[15-0] REFRESH_RATE and restarts decrementing. The counter also reloads and restarts decrementing whenever the EMIF_SDRFC[15-0] REFRESH_RATE field is updated.

The refresh backlog counter records the number of the outstanding REF commands which the EMIF controller currently has. The backlog counter increments by one (unless it has reached its maximum value of 15) each time the 13-bit interval counter reloads. The backlog counter decrements by one (unless it is already at zero) each time the EMIF controller issues a REF command.

For the range of values the backlog counter can take there are three levels of urgency with which the EMIF controller should perform refresh cycle in which it issues REF commands:

- *Refresh-may* level: reached whenever the backlog count is greater than 0. This indicates there is a refresh backlog, so if the EMIF controller is not busy and none of the SDRAM banks are open, it

should perform a refresh cycle.

- *Refresh-release* level: reached whenever the backlog count is greater than 4. This indicates the refresh backlog is getting higher, so if the EMIF controller is not busy it should perform a refresh cycle even if there is an open SDRAM bank.
- *Refresh-must* level: reached whenever the backlog count is greater than 7. This indicates the refresh backlog is getting excessive and the EMIF controller must perform a refresh cycle before servicing any new memory access requests.

The EMIF controller starts servicing new memory accesses after refresh-release level is cleared. If any of the commands in the Command FIFO have class-of-service latency counters that are expired, the EMIF controller does not wait for refresh-release level to be cleared. It only performs one refresh command and exits the refresh state.

The two counters do not operate when the SDRAM is in self-refresh mode. They start tracking the missed refreshes (the outstanding REF commands) only after the EMIF_SDRFC[31] INITREF_DIS bit is set to 0h.

The time between two REF commands is set through the EMIF_SDTIM4[13-4] T_RFC bit field.

7.2.4.4 Leveling

The EMIF controller supports leveling to compensate for the command and DQS skew as a result of the fly-by topology. Leveling compensates the skew for both reads and writes. The controller does not require the user to program initial leveling values to establish the starting point of the search window before triggering hardware leveling. The logic inside the DDR PHY is able to automatically search for and determine the optimal starting point when leveling and training sequence is triggered by writing to the DDR_PHY_PIR register.

7.2.4.5 Access Cycles

By default EMIF keeps its SDRAM chip select signals high (CSs are active-low). To direct a command to only one of the SDRAMs, EMIF asserts the chip select to that SDRAM for the duration of the command. If the EMIF_SDCFG[3] EBANK bit is set to 0, DDR3CE1z is always driven high except for Refresh, Power-Down and Self-Refresh commands.

The EMIF always performs burst accesses to the SDRAM. Multiple SDRAM bursts may be needed to service a single burst request to the port via which EMIF is connected to the system. Table 7-122 through Table 7-124 show a few examples how EMIF accesses the SDRAM for a linear incrementing transaction type. T0, T1, and so on are clock cycles. R0 is read starting at column 0, R8 is read starting at column 8, and R16 is read starting at column 16. D0-1 is the data from column 0 and 1, D2-3 is the data from column 2 and 3, and so on.

Table 7-122. 64-Byte Linear Read Starting at Address 0h

T0	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11
R0				R8							
				D0-1	D2-3	D4-5	D6-7	D8-9	D10-11	D12-13	D14-15

Table 7-123. 64-Byte Linear Read Starting at Address 10h

T0	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13
R4		R8				R16							
				D4-5	D6-7	D8-9	D10-11	D12-13	D14-15	D16-17	D18-19	Unused	Unused

Table 7-124. 64-Byte Linear Read Starting at Address 18h

T0	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13
R6		R8				R16							
				D6-7	Unused	D8-9	D10-11	D12-13	D14-15	D16-17	D18-19	D20-21	Unused

The EMIF uses the unused data phases in the preceding tables by issuing successive read commands if there are reads to open banks pending in the command FIFO.

7.2.4.6 Turnaround Time

Table 7-125 shows the turnaround time that the EMIF controller introduces on the data bus for various back-to-back accesses. Note that the EMIF controller takes advantage of the CAS latencies and packs the commands as close as possible on the control bus to introduce the following turn around time on the data bus.

Table 7-125. Turnaround Time

Previous Access	Next Access	Turnaround Time in SDRAM clock cycles (DDR3CLKOUT)
SDRAM Write	SDRAM Write to same chip select	0
SDRAM Write	SDRAM Write to different chip select	EMIF_SDTIM4[31-28] T_CSTA + 1
SDRAM Read	SDRAM Read to same chip select	0
SDRAM Read	SDRAM Read to different chip select	EMIF_SDTIM4[31-28] T_CSTA + 1
SDRAM Write	SDRAM Read	EMIF_SDTIM1[3-0] T_WTR + 1 + CL
SDRAM Read	SDRAM Write	EMIF_SDTIM1[3-0] T_RTW + 1

7.2.4.7 SDRAM Address Mapping

The EMIF controller views the external SDRAM as one continuous block of memory across the two chip selects. If smaller devices are used, the memory is seen to roll over. The EMIF controller receives SDRAM access requests along with 33-bit logical address from the rest of the system. The controller uses this logical address to generate a row (page), column, bank address, and chip selects for the DDR3L SDRAM. The number of bank and column address bits used is determined by the EMIF_SDCFG[6-5] IBANK and EMIF_SDCFG[1-0] PAGESIZE fields as shown in Table 7-126. The chip select lines used are determined by the EMIF_SDCFG[3] EBANK field.

Table 7-126. Bank Configuration Register Fields for Address Mapping

Bit Field in register EMIF_SDCFG	Bit Value	Bit Description
[6-5] IBANK		Defines the number of internal banks on external DDR3 memory
	0h	1 bank
	1h	2 banks
	2h	4 banks
	3h	8 banks
[3] EBANK		External chip select setup. Defines whether SDRAM accesses use 1 or 2 chip select lines
	0h	Use only chip enable 0 for all SDRAM accesses
	1h	Use chip enables 0 and 1 for SDRAM accesses
[1-0] PAGESIZE		Defines the page size of each page of the external DDR3 memory
	0h	256 words (requires 8 column address bits)
	1h	512 words (requires 9 column address bits)
	2h	1024 words (requires 10 column address bits)
	3h	2048 words (requires 11 column address bits)

NOTE: The EMIF_SDCFG[6-5] IBANK field should always be set to 3h since the DDR3 memory devices offer only 8-bank support unlike DDR2 with the option of 4-bank or 8-bank memories.

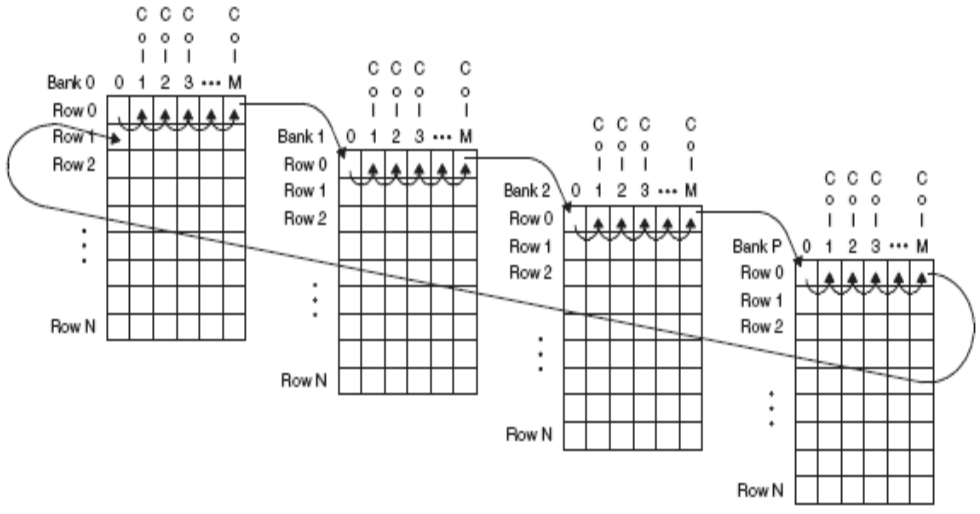
Table 7-127 shows the logical address-to-SDRAM address mapping. The effect of the address-mapping scheme is that as the source address increments across the SDRAM pages, EMIF moves to the same page in the next bank on the current chip select. This movement across the banks continues until the same page is accessed in all banks. Then EMIF moves to the next page in the first bank if the EMIF_SDCFG[3] EBANK bit is set to 0h. This process is illustrated on Figure 7-45. If EBANK is set to 1h the EMIF proceeds to the same page on the next chip select. Then it continues until the same page is accessed in all banks before accessing the next page on the first chip select.The EMIF uses this movement across chip selects and internal banks while remaining on the same page to maximize the number of the open SDRAM banks within the overall SDRAM space. Thus, the EMIF controller can keep a maximum of 16 banks (8 internal banks across two chip selects) open at a time and can interleave among all of them.

Table 7-127. Logical Address-to-SDRAM Address Mapping

Logical Address [32:N](1)							
Row Address		Chip Select		Bank Address		Column Address	
		EBANK	ncs(2)	IBANK	nbb(2)	PAGESIZE	ncb(2)
16 bits		0	0 bits	0	0 bits	0	8 bits
		1	1 bit	1	1 bit	1	9 bits
				2	2 bits	2	10 bits
				3	3 bits	3	11 bits
Logical address mapping for row address		Logical address mapping for chip select		Logical address mapping for bank address		Logical address mapping for column address	
MSB	LSB	MSB	LSB	MSB	LSB	MSB	LSB
M3 + 15	M3 + 1	M3 = M2 + ncs - 1	M2 + 1	M2 = M1 + nbb - 1	M1 + 1	M1 = N + ncb - 1	N

(1) N = 1 for 16-bit SDRAMs, N = 2 for 32-bit SDRAMs.
(2) nrb = number of row bits; ncs = number of chip select bits; nbb = number of bank address bits.

Figure 7-45. DDR3L SDRAM Column, Row, and Bank Access When EBANK = 0



A M is number of columns (as determined by PAGESIZE) minus 1, P is number of banks (as determined by IBANK) minus 1, and N is number of rows (as determined by both PAGESIZE and IBANK) minus 1.

7.2.4.8 DDR3L Output Impedance Calibration

The EMIF controller supports automatic output impedance (ZQ) calibration for DDR3L memories. The ZQ calibration can be enabled per chip select by setting the EMIF_ZQ_CONFIG[31] ZQ_CS1EN and EMIF_ZQ_CONFIG[30] ZQ_CS0EN bits. The EMIF supports the following types of ZQ calibration commands:

- ZQCS: ZQ calibration short command
- ZQCL: ZQ calibration long command.

The EMIF controller issues ZQCS command each time the EMIF_ZQ_CONFIG[15-0] ZQ_REFINTERVAL bit field expires. In other words, the ZQ_REFINTERVAL defines the interval between two ZQCS commands. When ZQCS command is issued, the EMIF waits and blocks any other command for EMIF_SDTIM4[23-16] ZQ_ZQCS + 1 number of DDR3CLKOUT clock cycles.

If the EMIF_ZQ_CONFIG[28] ZQ_SFEXITEN bit field is set to 1h, the EMIF issues ZQCL command every time it exits self-refresh mode. When ZQCL command is issued, the EMIF waits and blocks any other command for (EMIF_ZQ_CONFIG[18-16] ZQ_ZQCL_MULT + 1) × (EMIF_SDTIM4[23-16] ZQ_ZQCS + 1) number of DDR3CLKOUT clock cycles.

The ZQ calibration must be performed simultaneously over both chip selects. To enable ZQ calibration to be performed simultaneously over both chip selects the EMIF_ZQ_CONFIG[29] ZQ_DUALCALEN bit must be set to 1h. If ZQ_DUALCALEN is set to 0h, the EMIF performs ZQ calibration serially per chip select. ZQ_DUALCALEN = 0h is not supported and must not be used.

7.2.4.9 Error Correction And Detection

For data integrity, the EMIF controller supports ECC on the data written to or read from the ECC protected address ranges in memory. The ECC algorithm is a single-error-correct-double-error-detect (SECDED) algorithm and uses the (72,64) Hamming code. 8-bit ECC is calculated over 64-bit data quanta. ECC is enabled by setting to 1h both the EMIF_ECCCTL[31] ECC_EN and DDR_PHY_DX8GCR[0] DXEN bits. ECC is disabled by setting these two bits to 0h. The address ranges can be programmed in the EMIF_ECCADDR1 and EMIF_ECCADDR2 registers. Both registers have identical bits and functionality. This provides flexibility allowing two non-overlapping memory regions to be ECC protected. The system must ensure that any burst access with start address in the ECC protected region must not cross over to the unprotected region and vice-versa.

NOTE: The ECC is stored inside the SDRAM during writes. After enabling ECC and before performing any functional reads or writes, the SDRAM space configured as ECC should be first written with known data that is 64-bit aligned and with byte count multiple of 64-bit. This is to ensure the correct ECC values are stored in the ECC SDRAM before functional use.

NOTE: For both reads and writes the 64-bit data quanta is broken into two 32-bit words with 8-bit ECC calculated and stored over both 32-bit words.

The EMIF controller also has an additional read-modify-write (RMW) feature to support sub quanta accesses to the ECC protected memory. This feature can be enabled by setting to 1h the EMIF_ECCCTL[28] RMW_EN bit. If the RMW ECC feature is disabled (EMIF_ECCCTL[28] RMW_EN = 0h) and write access with byte count not multiple of 64-bit quanta or with non-64-bit-aligned address is performed within the address range protected by the ECC, the result is write ECC error interrupt. In this case the EMIF controller writes to the SDRAM but the ECC value written is corrupted. When RMW is enabled the controller performs a RMW operation if a write results in a sub-64-bit access. The details how EMIF performs RMW operations are described in Section 7.2.4.9.1.1.

NOTE: If ECC is used the RMW_EN bit must always be set to 1h to enable the RMW feature.

The ECC is read and verified during reads if the EMIF_ECCCTL[29] ECC_VERIFY_EN bit is set to 1h. ECC verification is disabled during reads if ECC_VERIFY_EN is set to 0h.

If there is 1-bit error, the EMIF controller corrects the data and sends it on the read interface. For 2-bit errors EMIF generates read ECC error interrupt. In both cases the data in the SDRAM is corrupted. It is software responsibility to correct that data.

NOTE: The user should note that the algorithm used by the ECC logic cannot detect more than 2-bit errors per 64-bit quanta. For these errors the output of the algorithm is unknown. It may erroneously detect as 1-bit, 2-bit or no error. More than 2-bit errors are expected to be very rare in a well designed system.

NOTE: To disable ECC, set to 0h both the EMIF_ECCCTL[31] ECC_EN and DDR_PHY_DX8GCR[0] DXEN bits before triggering the leveling sequence in the DDR_PHY_PIR register. By default DXEN is 1h, so the ECC byte lane is enabled from perspective of the PHY.

7.2.4.9.1 ECC Enhancements

7.2.4.9.1.1 Read-Modify-Write

NOTE: If ECC is used the EMIF_ECCCTL[28] RMW_EN bit must always be set to 1h to enable the RMW feature.

As shown in Figure 7-44, the Read-Modify-Write (RMW) module is placed before internal FIFO logic of the controller. The RMW module splits all SDRAM writes into a burst size of 64-bytes and converts any sub-quanta write into a RMW of aligned burst size (64 bytes). A quanta is defined as 8 bytes. Read bursts are not fragmented.

The RMW module has been designed such that normal aligned ECC block read or write bursts are unaffected. That is, they pass through without adding delay as well as maintaining all memory coherence. If RMW module detects write data that does not form a complete quanta, a read for that burst (64 bytes) is then issued by the controller. This return data burst is then merged with the sub-quanta write data and the modified 64-byte burst is then written to the SDRAM.

The controller maintains coherence across all outstanding commands in the command FIFO whether or not they are affected by the RMW module and also follows the normal command arbitration described in Section 7.2.4.1.1. The RMW process for sub-quanta writes means the execution of the write command will be delayed compared to if the command was not sub-quanta and was 64-bit aligned.

7.2.4.9.1.2 Logging 1-bit ECC Error Address

For 1-bit ECC error the controller logs the start address of the SDRAM burst in an internal 2-level deep address FIFO. The EMIF_ONE_BIT_ECC_ERR_ADDR_LOG register displays the address on top of the internal FIFO. Software must write 1h to the EMIF_ONE_BIT_ECC_ERR_ADDR_LOG register to pop the FIFO and display the next address stored. The FIFO is loaded with the address for the next 1-bit ECC error if it is not full. It must be noted that no address comparison is performed, that is, if a single address has ECC errors back-to-back, that address is logged twice.

7.2.4.9.1.3 Counting the Number of 1-bit ECC Errors

The number of 1-bit ECC errors can be counted using the EMIF_ONE_BIT_ECC_ERR_CNT register. The controller also supports programming a threshold and a window in the EMIF_ONE_BIT_ECC_ERR_THRSH register. The window is programmed in number of refresh periods. When the programmed window value is 0, that is, window is disabled and the internal error count exceeds the programmed threshold, the controller generates a 1-bit ECC error interrupt. When servicing the interrupt software has to set the EMIF_ONE_BIT_ECC_ERR_CNT register with a value less than the

threshold otherwise the interrupt will not be triggered again. When the programmed window value is non-zero, that is, window is enabled, the controller generates a 1-bit ECC error interrupt only if the internal error count exceeds the programmed threshold in that window. The EMIF_ONE_BIT_ECC_ERR_CNT register is reset every time the window expires. Software can use this to measure the degree of 1-bit ECC errors occurring in the system.

7.2.4.9.1.4 Diagnosing the Data Bus Bit on Which 1-bit ECC Errors Occur

For diagnosis, the controller has the EMIF_ONE_BIT_ECC_ERR_DIST_1 register that represent whether an error has occurred in a given bit location on the data. This is useful to detect whether the errors are random or systemic. The EMIF_ONE_BIT_ECC_ERR_DIST_1 register overlays all 1-bit ECC errors until software clears the register. Therefore, multiple bits could be set as a result of multiple 1-bit ECC errors occurring over multiple read accesses.

For 2-bit ECC errors the controller generate a 2-bit ECC error interrupt. It must be noted that the controller detects but does not correct 2-bit ECC errors. The controller generates a 2-bit ECC error interrupt and sends the resultant data from the ECC correction logic. The read data received from the memory may have further been corrupted by the ECC correction logic since it will have attempted to correct the read data but failed due to uncorrectable error.

7.2.4.9.1.5 Logging 2-bit ECC Error Address

For all uncorrectable ECC errors the controller logs the start address of the SDRAM burst in the EMIF_TWO_BIT_ECC_ERR_ADDR_LOG register. This register shows the address of the first uncorrectable error. After software clears the register, it will be loaded with the address for the next uncorrectable error.

7.2.4.10 Interrupt Requests

The EMIF controller generates one interrupt request, the DDR3_ERR, associated with the following registers:

- EMIF_IRQSTATUS_RAW_SYS — interrupt raw status register
- EMIF_IRQSTATUS_SYS — interrupt status register
- EMIF_IRQENABLE_SET_SYS — interrupt enable register
- EMIF_IRQENABLE_CLR_SYS — interrupt disable register

For the interrupt behavior of the EMIF controller the following applies:

- EMIF asserts its interrupt line DDR3_ERR only if the interrupts are enabled by setting to 1h the corresponding bits in the EMIF_IRQENABLE_SET_SYS register. These interrupts can be disabled by setting to 1h the corresponding bits in the EMIF_IRQENABLE_CLR_SYS register.
- After an interrupt has been serviced, software must clear the corresponding status flag. This is done by setting to 1h the corresponding bit in the EMIF_IRQSTATUS_SYS register which also clears the corresponding bit in the EMIF_IRQSTATUS_RAW_SYS register. The status flags in the EMIF_IRQSTATUS_RAW_SYS register are set even if the corresponding interrupt is disabled unlike those in the EMIF_IRQSTATUS_SYS register, which are set only if the corresponding interrupt is enabled.
- An interrupt is also generated by the EMIF, if certain bit in the EMIF_IRQSTATUS_RAW_SYS register is set to 1h and the corresponding interrupt is enabled through the EMIF_IRQENABLE_SET_SYS register. This feature is useful during user software debugging. In addition, even if interrupts are not enabled the corresponding raw flag in the EMIF_IRQSTATUS_RAW_SYS register is set to 1h when an interrupt condition occurs.
- Even if interrupts are not enabled, a status bit in the EMIF_IRQSTATUS_RAW_SYS register can also be cleared by setting to 1h the corresponding bit in the EMIF_IRQSTATUS_SYS register.

Table 7-128 lists the interrupt events which can assert the DDR3_ERR interrupt line.

Table 7-128. EMIF Events

Event Flag	Event Mask	Description
EMIF_IRQSTATUS_RAW_SYS[5] 1B_ECC_ERR_SYS EMIF_IRQSTATUS_SYS[5] 1B_ECC_ERR_SYS	EMIF_IRQENABLE_SET_SYS[5] 1B_ECC_ERR_SYS EMIF_IRQENABLE_CLR_SYS[5] 1B_ECC_ERR_SYS	1-bit ECC error interrupt. See Section 7.2.4.9.1.3 for more information.
EMIF_IRQSTATUS_RAW_SYS[4] 2B_ECC_ERR_SYS EMIF_IRQSTATUS_SYS[4] 2B_ECC_ERR_SYS	EMIF_IRQENABLE_SET_SYS[4] 2B_ECC_ERR_SYS EMIF_IRQENABLE_CLR_SYS[4] 2B_ECC_ERR_SYS	2-bit ECC error interrupt. See Section 7.2.4.9.1.5 for more information.
EMIF_IRQSTATUS_RAW_SYS[3] WR_ECC_ERR_SYS EMIF_IRQSTATUS_SYS[3] WR_ECC_ERR_SYS	EMIF_IRQENABLE_SET_SYS[3] WR_ECC_ERR_SYS EMIF_IRQENABLE_CLR_SYS[3] WR_ECC_ERR_SYS	Interrupt for memory access made to a non-quanta aligned location or done with byte count not multiple of the ECC quanta.
EMIF_IRQSTATUS_RAW_SYS[0] ERR_SYS EMIF_IRQSTATUS_SYS[0] ERR_SYS	EMIF_IRQENABLE_SET_SYS[0] EN_ERR_SYS EMIF_IRQENABLE_CLR_SYS[0] EN_ERR_SYS	Interrupt for command or address error.

7.2.4.11 Performance Monitoring

The EMIF_PERF_CNT_1 and EMIF_PERF_CNT_2 registers are used to monitor or calculate the EMIF Controller bandwidth and efficiency. These counters are able to count events such as accesses made to EMIF, Activate (ACT) commands sent to SDRAM, read and write accesses made to EMIF, and other events. Each counter counts independent of the other. In addition to the ability of events counting, the counters are also able to filter the events from a particular master or address space. The events counting and filter enabling are configured using the EMIF_PERF_CNT_CFG register. The actual value of the filter is configured through the EMIF_PERF_CNT_SEL register. Each counter can be configured independently.

Table 7-129 lists all the events that can be counted and whether a filter can be applied to a particular event. A filter is applied to an event if the following bits are set to 1h for that event:

- For Performance Counter 1: EMIF_PERF_CNT_CFG[15] CNTR1_MSTID_EN and EMIF_PERF_CNT_CFG[14] CNTR1_REGION_EN
- For Performance Counter 2: EMIF_PERF_CNT_CFG[31] CNTR2_MSTID_EN and EMIF_PERF_CNT_CFG[30] CNTR2_REGION_EN

The CNTR*n*_CFG (*n* = 1 or 2) fields in the EMIF_PERF_CNT_CFG register are used to select the event to count.

Table 7-129. Performance Counter Filter Configuration

CNTR*n*_CFG[(1)]	CNTR*n*_REGION_EN	CNTR*n*_MSTID_EN	Event Selected for Counting
0h	0h	0h or 1h	Count the accesses made to EMIF
1h	0h	0h or 1h	Count the Activate (ACT) commands sent to SDRAM
2h	0h or 1h	0h or 1h	Count the read accesses made to EMIF
3h	0h or 1h	0h or 1h	Count the write accesses made to EMIF
4h	0h	0h	Count the number of EMIF_M_CLK clock cycles during which the Command FIFO is full
5h	0h	0h	Count the number of EMIF_M_CLK clock cycles during which the Write Data FIFO is full
6h	0h	0h	Count the number of EMIF_M_CLK clock cycles during which the Read Data FIFO is full
7h	0h	0h	Count the number of EMIF_M_CLK clock cycles during which the Write Status FIFO is full
8h	0h or 1h	0h or 1h	Count the number of priority elevations
9h	0h	0h	Count the number of EMIF_M_CLK clock cycles that a command was pending

(1) *n* = 1 or 2

Table 7-129. Performance Counter Filter Configuration (continued)

CNTR*n*_CFG[(1)]	CNTR*n*_REGION_EN	CNTR*n*_MSTID_EN	Event Selected for Counting
Ah	0h	0h	Count the number of EMIF_M_CLK cycles for which the SDRAM data bus was transferring data.
Bh	0h	0h	Count the number of SDRAM read bursts resulting from RMW and wRMW accesses
Ch	0h	0h	Count the number of SDRAM write bursts resulting from wRMW accesses
Dh-Fh	0h	0h	Reserved for future use.

NOTE: The EMIF performance counters cannot distinguish between single access and burst access. In both cases they are incremented by 1. If the actual SDRAM bandwidth of an initiator has to be measured the EMIF performance counters may not be sufficient.

7.2.4.11.1 Performance Counters General Examples

- **General Example for Counting Total Accesses made to EMIF**
  If the EMIF_PERF_CNT_2 register is used to count total accesses made to EMIF regardless of the address space or master the following steps should be performed:
  - To enable counting of all accesses to the SDRAM, the EMIF_PERF_CNT_CFG[19-16] CNTR2_CFG bit field must be set to 0h.
  - To disable filtering, both the EMIF_PERF_CNT_CFG[31] CNTR2_MSTID_EN and EMIF_PERF_CNT_CFG[30] CNTR2_REGION_EN bits must be set to 0h.

    With this configuration EMIF_PERF_CNT_2 counts every access made to the EMIF. This includes all accesses from all masters and to any address space.
- **General Example for Counting All Read Accesses made to EMIF**
  If the EMIF_PERF_CNT_1 register is used to count all read accesses made to EMIF from master with Master ID equal to 41 (that is, EDMA_0_TC0_R) to the SDRAM space the following steps should be performed:
  - To enable counting reads, the EMIF_PERF_CNT_CFG[3-0] CNTR1_CFG bit field must be set to 2h.
  - The EMIF_PERF_CNT_SEL[15-8] MSTID1 bit field must be set to 29h
  - The EMIF_PERF_CNT_SEL[3-0] REGION_SEL1 bit field must be set to 0h for SDRAM space.
  - To enable filtering, both the EMIF_PERF_CNT_CFG[15] CNTR1_MSTID_EN and the EMIF_PERF_CNT_CFG[14] CNTR1_REGION_EN bits must be set to 1h.

    With this configuration EMIF_PERF_CNT_1 counts every read made to the EMIF from master with ID 41 to the SDRAM space. This does not include accesses from other masters or to other address spaces (for example, EMIF configuration registers) and does not include commands other than reads.

7.2.4.12 Emulation Considerations

The EMIF controller remains fully functional during emulation halts to allow emulation access to external memory.

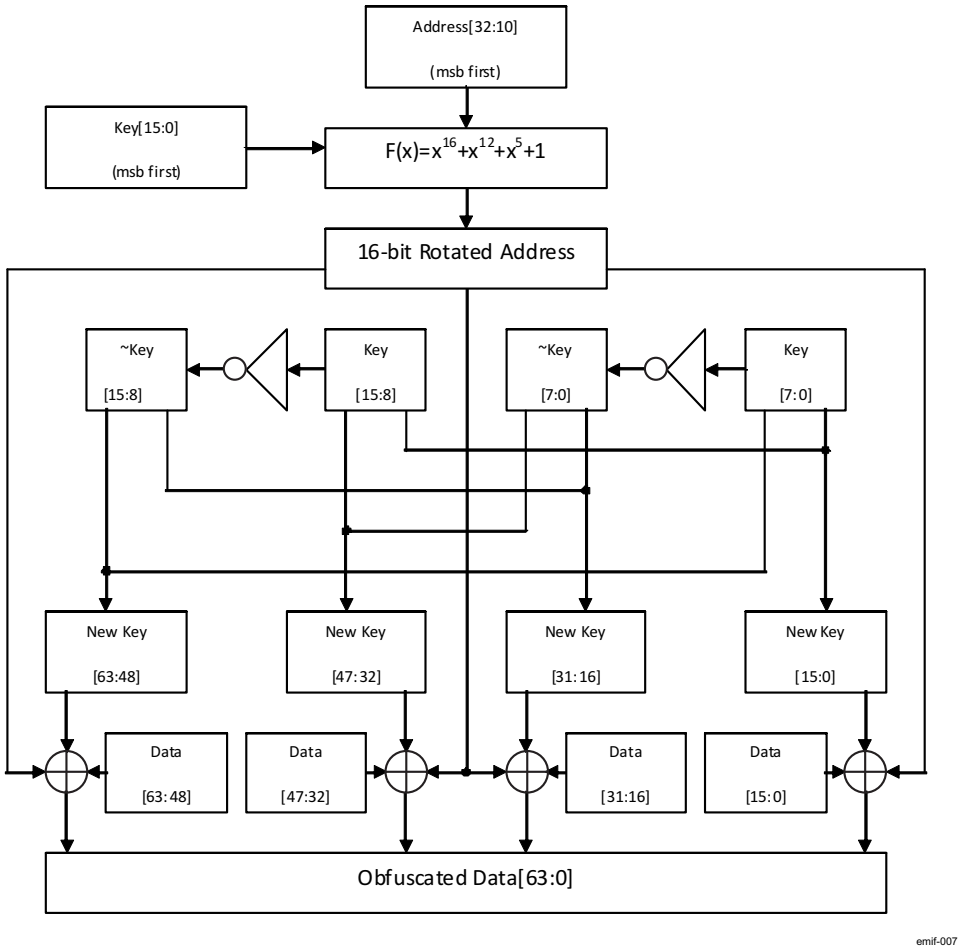
7.2.4.13 Data Bus Obfuscation

For security, the EMIF controller supports obfuscation for data written to the SDRAM. Data written to the EMIF configuration registers is not obfuscated.

NOTE: Obfuscation is available only on secure TI devices. Contact a TI sales office for additional information.

Figure 7-46 shows the obfuscation scheme.

Figure 7-46. Data Bus Obfuscation



Obfuscation is enabled by setting to 1 the OBFUSCATION_ENABLE bit inside the Security Manager. This causes the obfuscation key to be latched inside the controller. The 16-bit key should be programmed by software inside the Security Manager.

NOTE: Since the key is latched when obfuscation is enabled, the key should not be modified further.

The key is further modified to create a 64-bit new key. Data going out from the controller during writes is obfuscated and data coming back during reads is de-obfuscated. As seen from Figure 7-46, the obfuscation logic makes use of the upper address bits to ensure that obfuscation is different for different regions of memory.

Further, the data is also XORed with four copies of the 16-bit rotated address and the new key to ensure that obfuscation is also unique for each 1KB page of memory.

Enabling obfuscation does not increase the latency on command execution.

7.2.5 EMIF Registers

Table 7-131 lists the EMIF configuration registers and also the DDR_PHY configuration registers. All other register offset addresses not listed in Table 7-131 should be considered as reserved locations and the register contents should not be modified.

Table 7-130. EMIF and DDR_PHY Instances

Instance	Base Address
EMIF	2101 0000h
DDR_PHY	0232 9000h

Table 7-131. EMIF and DDR_PHY Registers

Offset	Acronym	Register Description	EMIF Physical Address	DDR_PHY Physical Address	Section
0h	EMIF_MIDR	Module ID and Revision Register	2101 0000h	-	Section 7.2.5.1
4h	EMIF_STATUS	EMIF controller Status Register	2101 0004h	-	Section 7.2.5.2
8h	EMIF_SDCFG	SDRAM Configuration Register	2101 0008h	-	Section 7.2.5.3
10h	EMIF_SDRFC	SDRAM Refresh Control Register	2101 0010h	-	Section 7.2.5.4
18h	EMIF_SDTIM1	SDRAM Timing 1 Register	2101 0018h	-	Section 7.2.5.5
1Ch	EMIF_SDTIM2	SDRAM Timing 2 Register	2101 001Ch	-	Section 7.2.5.6
20h	EMIF_SDTIM3	SDRAM Timing 3 Register	2101 0020h	-	Section 7.2.5.7
28h	EMIF_SDTIM4	SDRAM Timing 4 Register	2101 0028h	-	Section 7.2.5.8
38h	EMIF_PMCTL	Power Management Control Register	2101 0038h	-	Section 7.2.5.9
54h	EMIF_VBUSM_CONFIG	Latency Configuration Register	2101 0054h	-	Section 7.2.5.10
80h	EMIF_PERF_CNT_1	Performance Counter 1 Register	2101 0080h	-	Section 7.2.5.11
84h	EMIF_PERF_CNT_2	Performance Counter 2 Register	2101 0084h	-	Section 7.2.5.12
88h	EMIF_PERF_CNT_CFG	Performance Counter Config Register	2101 0088h	-	Section 7.2.5.13
8Ch	EMIF_PERF_CNT_SEL	Performance Counter Master Region Select Register	2101 008Ch	-	Section 7.2.5.14
90h	EMIF_PERF_CNT_TIM	Performance Counter Time Register	2101 0090h	-	Section 7.2.5.15
A0h	EMIF_IRQ_EOI	End of Interrupt Register	2101 00A0h	-	Section 7.2.5.16
A4h	EMIF_IRQSTATUS_RAW_SYS	Interrupt Raw Status Register	2101 00A4h	-	Section 7.2.5.17
ACh	EMIF_IRQSTATUS_SYS	Interrupt Status Register	2101 00ACh	-	Section 7.2.5.18
B4h	EMIF_IRQENABLE_SET_SYS	Interrupt Enable Set Register	2101 00B4h	-	Section 7.2.5.19
BCh	EMIF_IRQENABLE_CLR_SYS	Interrupt Enable Clear Register	2101 00BCh	-	Section 7.2.5.20
C8h	EMIF_ZQ_CONFIG	SDRAM Output Impedance Calibration Configuration Register	2101 00C8h	-	Section 7.2.5.21
100h	EMIF_PRI_COS_MAP	Priority To Class-Of-Service Mapping Register	2101 0100h	-	Section 7.2.5.22

Table 7-131. EMIF and DDR_PHY Registers (continued)

Offset	Acronym	Register Description	EMIF Physical Address	DDR_PHY Physical Address	Section
104h	EMIF_MSTID_COS_1_MAP	Master ID to Class-Of-Service 1 Mapping Register	2101 0104h	-	Section 7.2.5.23
108h	EMIF_MSTID_COS_2_MAP	Master ID to Class-Of-Service 2 Mapping Register	2101 0108h	-	Section 7.2.5.24
110h	EMIF_ECCCTL	ECC Control Register	2101 0110h	-	Section 7.2.5.25
114h	EMIF_ECCADDR1	ECC Address Range 1 Register	2101 0114h	-	Section 7.2.5.26
118h	EMIF_ECCADDR2	ECC Address Range 2 Register	2101 0118h	-	Section 7.2.5.27
120h	EMIF_RWTHRESH	Read Write Execution Threshold Register	2101 0120h	-	Section 7.2.5.28
130h	EMIF_ONE_BIT_ECC_ERR_CNT	1-Bit ECC Error Count Register	2101 0130h	-	Section 7.2.5.29
134h	EMIF_ONE_BIT_ECC_ERR_THRSH	1-Bit ECC Error Threshold Register	2101 0134h	-	Section 7.2.5.30
138h	EMIF_ONE_BIT_ECC_ERR_DIST_1	1-Bit ECC Error Distribution 1 Register	2101 0138h	-	Section 7.2.5.31
13Ch	EMIF_ONE_BIT_ECC_ERR_ADDR_LOG	1-Bit ECC Error Address Log Register	2101 013Ch	-	Section 7.2.5.32
140h	EMIF_TWO_BIT_ECC_ERR_ADDR_LOG	2-Bit ECC Error Address Log Register	2101 0140h	-	Section 7.2.5.33
144h	EMIF_ONE_BIT_ECC_ERR_DIST_2	1-Bit ECC Error Distribution 2 Register	2101 0144h	-	Section 7.2.5.34
4h	DDR_PHY_PIR	PHY Initialization Register	-	0232 9004h	Section 7.2.5.35
8h	DDR_PHY_PGCR0	PHY General Configuration Register 0	-	0232 9008h	Section 7.2.5.36
Ch	DDR_PHY_PGCR1	PHY General Configuration Register 1	-	0232 900Ch	Section 7.2.5.37
10h	DDR_PHY_PGSR0	PHY General Status Register 0	-	0232 9010h	Section 7.2.5.39
14h	DDR_PHY_PGSR1	PHY General Status Register 1	-	0232 9014h	Section 7.2.5.40
18h	DDR_PHY_PLLCR	PLL Control Register	-	0232 9018h	Section 7.2.5.41
1Ch	DDR_PHY_PTR0	PHY Timing Register 0	-	0232 901Ch	Section 7.2.5.42
20h	DDR_PHY_PTR1	PHY Timing Register 1	-	0232 9020h	Section 7.2.5.43
24h	DDR_PHY_PTR2	PHY Timing Register 2	-	0232 9024h	Section 7.2.5.44
28h	DDR_PHY_PTR3	PHY Timing Register 3	-	0232 9028h	Section 7.2.5.45
2Ch	DDR_PHY_PTR4	PHY Timing Register 4	-	0232 902Ch	Section 7.2.5.46
38h	DDR_PHY_ACIOCR	AC I/O Configuration Register	-	0232 9038h	Section 7.2.5.47
3Ch	DDR_PHY_DXCCR	DATX8 Common Configuration Register	-	0232 903Ch	Section 7.2.5.48
44h	DDR_PHY_DCR	DRAM Configuration Register	-	0232 9044h	Section 7.2.5.49
48h	DDR_PHY_DTPR0	DRAM Timing Parameters Register 0	-	0232 9048h	Section 7.2.5.50

Table 7-131. EMIF and DDR_PHY Registers (continued)

Offset	Acronym	Register Description	EMIF Physical Address	DDR_PHY Physical Address	Section
4Ch	DDR_PHY_DTPR1	DRAM Timing Parameters Register 1	-	0232 904Ch	Section 7.2.5.51
50h	DDR_PHY_DTPR2	DRAM Timing Parameters Register 2	-	0232 9050h	Section 7.2.5.52
54h	DDR_PHY_MR0	Mode Register 0	-	0232 9054h	Section 7.2.5.53
58h	DDR_PHY_MR1	Mode Register 1	-	0232 9058h	Section 7.2.5.54
5Ch	DDR_PHY_MR2	Mode Register 2	-	0232 905Ch	Section 7.2.5.55
60h	DDR_PHY_MR3	Mode Register 3	-	0232 9060h	Section 7.2.5.56
64h	DDR_PHY_ODTCR	ODT Configuration Register	-	0232 9064h	Section 7.2.5.57
68h	DDR_PHY_DTCR	Data Training Configuration Register	-	0232 9068h	Section 7.2.5.58
8Ch	DDR_PHY_PGCR2	PHY General Configuration Register 2	-	0232 908Ch	Section 7.2.5.38
180h	DDR_PHY_ZQ0CR0	ZQ 0 Impedance Control Register 0	-	0232 9180h	Section 7.2.5.59
184h	DDR_PHY_ZQ0CR1	ZQ 0 Impedance Control Register 1	-	0232 9184h	Section 7.2.5.60
188h	DDR_PHY_ZQ0SR0	ZQ 0 Impedance Status Register 0	-	0232 9188h	Section 7.2.5.61
18Ch	DDR_PHY_ZQ0SR1	ZQ 0 Impedance Status Register 1	-	0232 918Ch	Section 7.2.5.62
190h	DDR_PHY_ZQ1CR0	ZQ 1 Impedance Control Register 0	-	0232 9190h	Section 7.2.5.63
194h	DDR_PHY_ZQ1CR1	ZQ 1 Impedance Control Register 1	-	0232 9194h	Section 7.2.5.64
198h	DDR_PHY_ZQ1SR0	ZQ 1 Impedance Status Register 0	-	0232 9198h	Section 7.2.5.65
19Ch	DDR_PHY_ZQ1SR1	ZQ 1 Impedance Status Register 1	-	0232 919Ch	Section 7.2.5.66
1A0h	DDR_PHY_ZQ2CR0	ZQ 2 Impedance Control Register 0	-	0232 91A0h	Section 7.2.5.67
1A4h	DDR_PHY_ZQ2CR1	ZQ 2 Impedance Control Register 1	-	0232 91A4h	Section 7.2.5.68
1A8h	DDR_PHY_ZQ2SR0	ZQ 2 Impedance Status Register 0	-	0232 91A8h	Section 7.2.5.69
1ACh	DDR_PHY_ZQ2SR1	ZQ 2 Impedance Status Register 1	-	0232 91ACh	Section 7.2.5.70
1C0h	DDR_PHY_DX0GCR	DATX8 0 General Configuration Register	-	0232 91C0h	Section 7.2.5.71
1C4h	DDR_PHY_DX0GSR0	DATX8 0 General Status Register 0	-	0232 91C4h	Section 7.2.5.72
1E0h	DDR_PHY_DX0LCDLR0	DATX8 0 Local Calibrated Delay Line Register 0	-	0232 91E0h	Section 7.2.5.74
1E4h	DDR_PHY_DX0LCDLR1	DATX8 0 Local Calibrated Delay Line Register 1	-	0232 91E4h	Section 7.2.5.75
1E8h	DDR_PHY_DX0LCDLR2	DATX8 0 Local Calibrated Delay Line Register 2	-	0232 91E8h	Section 7.2.5.76
1ECh	DDR_PHY_DX0MDLR	DATX8 0 Master Delay Line Register	-	0232 91ECh	Section 7.2.5.77

Table 7-131. EMIF and DDR_PHY Registers (continued)

Offset	Acronym	Register Description	EMIF Physical Address	DDR_PHY Physical Address	Section
1F0h	DDR_PHY_DX0GTR	DATX8 0 General Timing Register	-	0232 91F0h	Section 7.2.5.78
1F4h	DDR_PHY_DX0GSR2	DATX8 0 General Status Register 2	-	0232 91F4h	Section 7.2.5.73
200h	DDR_PHY_DX1GCR	DATX8 1 General Configuration Register	-	0232 9200h	Section 7.2.5.79
204h	DDR_PHY_DX1GSR0	DATX8 1 General Status Register 0	-	0232 9204h	Section 7.2.5.80
220h	DDR_PHY_DX1LCDLR0	DATX8 1 Local Calibrated Delay Line Register 0	-	0232 9220h	Section 7.2.5.82
224h	DDR_PHY_DX1LCDLR1	DATX8 1 Local Calibrated Delay Line Register 1	-	0232 9224h	Section 7.2.5.83
228h	DDR_PHY_DX1LCDLR2	DATX8 1 Local Calibrated Delay Line Register 2	-	0232 9228h	Section 7.2.5.84
22Ch	DDR_PHY_DX1MDLR	DATX8 1 Master Delay Line Register	-	0232 922Ch	Section 7.2.5.85
230h	DDR_PHY_DX1GTR	DATX8 1 General Timing Register	-	0232 9230h	Section 7.2.5.86
234h	DDR_PHY_DX1GSR2	DATX8 1 General Status Register 2	-	0232 9234h	Section 7.2.5.81
240h	DDR_PHY_DX2GCR	DATX8 2 General Configuration Register	-	0232 9240h	Section 7.2.5.79
244h	DDR_PHY_DX2GSR0	DATX8 2 General Status Register 0	-	0232 9244h	Section 7.2.5.88
260h	DDR_PHY_DX2LCDLR0	DATX8 2 Local Calibrated Delay Line Register 0	-	0232 9260h	Section 7.2.5.90
264h	DDR_PHY_DX2LCDLR1	DATX8 2 Local Calibrated Delay Line Register 1	-	0232 9264h	Section 7.2.5.91
268h	DDR_PHY_DX2LCDLR2	DATX8 2 Local Calibrated Delay Line Register 2	-	0232 9268h	Section 7.2.5.92
26Ch	DDR_PHY_DX2MDLR	DATX8 2 Master Delay Line Register	-	0232 926Ch	Section 7.2.5.93
270h	DDR_PHY_DX2GTR	DATX8 2 General Timing Register	-	0232 9270h	Section 7.2.5.94
274h	DDR_PHY_DX2GSR2	DATX8 2 General Status Register 2	-	0232 9274h	Section 7.2.5.89
280h	DDR_PHY_DX3GCR	DATX8 3 General Configuration Register	-	0232 9280h	Section 7.2.5.87
284h	DDR_PHY_DX3GSR0	DATX8 3 General Status Register 0	-	0232 9284h	Section 7.2.5.96
2A0h	DDR_PHY_DX3LCDLR0	DATX8 3 Local Calibrated Delay Line Register 0	-	0232 92A0h	Section 7.2.5.98
2A4h	DDR_PHY_DX3LCDLR1	DATX8 3 Local Calibrated Delay Line Register 1	-	0232 92A4h	Section 7.2.5.99
2A8h	DDR_PHY_DX3LCDLR2	DATX8 3 Local Calibrated Delay Line Register 2	-	0232 92A8h	Section 7.2.5.100
2ACh	DDR_PHY_DX3MDLR	DATX8 3 Master Delay Line Register	-	0232 92ACh	Section 7.2.5.101
2B0h	DDR_PHY_DX3GTR	DATX8 3 General Timing Register	-	0232 92B0h	Section 7.2.5.102
2B4h	DDR_PHY_DX3GSR2	DATX8 3 General Status Register 2	-	0232 92B4h	Section 7.2.5.97
3C0h	DDR_PHY_DX8GCR	DATX8 8 General Configuration Register	-	0232 93C0h	Section 7.2.5.95

Table 7-131. EMIF and DDR_PHY Registers (continued)

Offset	Acronym	Register Description	EMIF Physical Address	DDR_PHY Physical Address	Section
3C4h	DDR_PHY_DX8GSR0	DATX8 8 General Status Register 0	-	0232 93C4h	Section 7.2.5.104
3E0h	DDR_PHY_DX8LCDLR0	DATX8 8 Local Calibrated Delay Line Register 0	-	0232 93E0h	Section 7.2.5.106
3E4h	DDR_PHY_DX8LCDLR1	DATX8 8 Local Calibrated Delay Line Register 1	-	0232 93E4h	Section 7.2.5.107
3E8h	DDR_PHY_DX8LCDLR2	DATX8 8 Local Calibrated Delay Line Register 2	-	0232 93E8h	Section 7.2.5.108
3ECh	DDR_PHY_DX8MDLR	DATX8 8 Master Delay Line Register	-	0232 93ECh	Section 7.2.5.109
3F0h	DDR_PHY_DX8GTR	DATX8 8 General Timing Register	-	0232 93F0h	Section 7.2.5.110
3F4h	DDR_PHY_DX8GSR2	DATX8 8 General Status Register 2	-	0232 93F4h	Section 7.2.5.105

7.2.5.1 EMIF_MIDR Register (Offset = 0h) [reset = 40461C02h]

The Module ID and Revision register contains the revision number and identification data of the DDR3 peripheral, and is described in the following figure and table.

Table 7-132. EMIF_MIDR Instances

Instance	Physical Address
EMIF	2101 0000h

Figure 7-47. EMIF_MIDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-40461C02h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 7-133. EMIF_MIDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	40461C02h	TI internal data. Identifies revision of peripheral.

Table 7-134. Register Call Summary for EMIF_MIDR

EMIF Functional Description • Possible Race Condition: [0][1]
EMIF Registers • EMIF Registers: [0]

7.2.5.2 EMIF_STATUS Register (Offset = 4h) [reset = 0h]

This register contains the status of the DDR3 module and is described in the following figure and table.

Table 7-135. EMIF_STATUS Instances

Instance	Physical Address
EMIF	2101 0004h

Figure 7-48. EMIF_STATUS Register

31	30	29	28	27	26	25	24
BE	RESERVED		OBF_STAT	SELF_REF	PWRDN	RESERVED	
R-0h	R-0h		R-0h	R-0h	R-0h	R-0h	
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					IFRDY	RESERVED	
R-0h					R-0h	R-0h	

LEGEND: R = Read Only; *-n* = value after reset

Table 7-136. EMIF_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31	BE	R	0h	Indicates whether the EMIF is in big or little-endian mode. **NOTE: Only little-endian mode is supported on this device.**
30-29	RESERVED	R	0h	This field is tied off to 1h.
28	OBF_STAT	R	0h	Obfuscation Status. • 0 = Obfuscation disabled. • 1 = Obfuscation enabled.
27	SELF_REF	R	0h	Self refresh mode status. • 0 = SDRAM is not in self refresh mode. • 1 = SDRAM is in self refresh mode.
26	PWRDN	R	0h	Power down status. • 0 = SDRAM is not in power down mode. • 1 = SDRAM is in power down mode.
25-3	RESERVED	R	0h	Reserved. A value written to this field has no effect
2	IFRDY	R	0h	EMIF controller interface logic ready bit. The interface logic controls the signals used to communicate with DDR3 SDRAM devices. This bit displays the status of the interface logic. • 0 = Interface logic is not ready; either powered down, not ready, or not locked. • 1 = Interface logic is powered up, locked and ready for operation.
1-0	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

Table 7-137. Register Call Summary for EMIF_STATUS

EMIF Functional Description • SDRAM Self-Refresh Mode: [0]
EMIF Registers • EMIF Registers: [0]

7.2.5.3 EMIF_SDCFG Register (Offset = 8h) [reset = 200h]

The SDRAM Configuration Register (EMIF_SDCFG) contains field that program the EMIF controller to meet the specifications of the DDR3 memory. These fields configure the EMIF controller to match the data bus width, CAS latency, number of internal banks, and page size of the external DDR3 memory.

Table 7-138. EMIF_SDCFG Instances

Instance	Physical Address
EMIF	2101 0008h

Figure 7-49. EMIF_SDCFG Register

31	30	29	28	27	26	25	24
SDRAM_TYPE			RESERVED	DDR_TERM			RESERVED
RW-0h			R-0h	RW-0h			R-0h
23	22	21	20	19	18	17	16
DYN_ODT		RESERVED					CWL
RW-0h		R-0h					RW-0h
15	14	13	12	11	10	9	8
CWL		NM		CL			
RW-0h		RW-0h		RW-2h			
7	6	5	4	3	2	1	0
RESERVED	IBANK		RESERVED	EBANK	RESERVED	PAGESIZE	
R-0h	RW-0h		R-0h	RW-0h	R-0h	RW-0h	

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-139. EMIF_SDCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	SDRAM_TYPE	RW	0h	SDRAM type selection. Set to 3 for DDR3. All other values reserved.
28	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
27-25	DDR_TERM	RW	0h	Defines termination resistor value. • 0 = Disables termination • 1 = RZQ/4 • 2 = RZQ/2 • 3 = RZQ/6 • 4 = RZQ/12 • 5 = RZQ/8
24	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
23-22	DYN_ODT	RW	0h	Dynamic On-Die Termination • 0 = Turn off dynamic ODT. • 1 = RZQ/4 • 2 = RZQ/2 All other values reserved.
21-17	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

Table 7-139. EMIF_SDCFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16-14	CWL	RW	0h	CAS Write Latency. Lower value gives better performance. • 0 = CAS write latency of 5 • 1 = CAS write latency of 6 • 2 = CAS write latency of 7 • 3 = CAS write latency of 8 • 4 = CAS write latency of 9 • 5 = CAS write latency of 10 • 6 = CAS write latency of 11 • 7 = CAS write latency of 12
13-12	NM	RW	0h	DDR3 data bus width. • 1 = 32-bit bus width. • 2 = 16-bit bus width. All other values reserved.
11-8	CL	RW	2h	CAS Latency. The value of this field defines the CAS latency, to be used when accessing connected SDRAM devices. • 2 = CAS latency of 5. • 4 = CAS latency of 6. • 6 = CAS latency of 7. • 8 = CAS latency of 8. • 10 = CAS latency of 9. • 12 = CAS latency of 10. • 14 = CAS latency of 11. • 1 = CAS latency of 12. • 3 = CAS latency of 13. • 5 = CAS latency of 14. • 7 = CAS latency of 15. • 9 = CAS latency of 16. All other values are reserved.
7	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
6- 5	IBANK	RW	0h	Internal SDRAM bank setup bits. Defines number of banks inside connected SDRAM devices. Values 4-7 are reserved for this field. A word is equal to the bus width of the individual SDRAM devices used (example, 1 byte for x8 and 2bytes for x16 devices) • 0 = 1 bank SDRAM devices. • 1 = 2 bank SDRAM devices. • 2 = 4 bank SDRAM devices. • 3 = 8 bank SDRAM devices.
4	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
3	EBANK	RW	0h	External chip select setup. Defines whether SDRAM accesses use 1 or 2 chip select lines as follows: • 0 = Use DDR3_CEn0 for all SDRAM accesses. • 1 = Use DDR3_CEn0 and DDR3_CEn1 for SDRAM accesses.
2	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
1-0	PAGESIZE	RW	0h	Page size bits. Defines internal page size of the external DDR3 memory. Values 4-7 are reserved for this field. A word is equal to the bus width of the individual SDRAM devices used (example, 1 byte for x8 and 2 bytes for x16 devices) • 0 = 256-word page requiring 8 column address bits. • 1 = 512-word page requiring 9 column address bits. • 2 = 1024-word page requiring 10 column address bits. • 3 = 2048-word page requiring 11 column address bits.

Table 7-140. Register Call Summary for EMIF_SDCFG

EMIF Functional Description • SDRAM Address Mapping: [0][1][2][3][4][5] • Access Cycles: [0]
EMIF Registers • EMIF_SDCFG Register (Offset = 8h) [reset = 200h]: [0] • EMIF Registers: [0]

7.2.5.4 EMIF_SDRFC Register (Offset = 10h) [reset = 80000000h]

The SDRAM Refresh Control Register (EMIF_SDRFC) is used to configure the EMIF controller to:

- Enter and Exit the self-refresh state.
- Meet the refresh requirements of the attached SDRAM device by programming a rate at which the EMIF controller issues autorefresh commands.

The EMIF_SDRFC register is described in the following figure and table.

Table 7-141. EMIF_SDRFC Instances

Instance	Physical Address
EMIF	2101 0010h

Figure 7-50. EMIF_SDRFC Register

31	30	29	28	27	26	25	24
INITREF_DIS	RESERVED						
RW-1h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
REFRESH_RATE							
RW-0h							
7	6	5	4	3	2	1	0
REFRESH_RATE							
RW-0h							

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 7-142. EMIF_SDRFC Register Field Descriptions

Bit	Field	Type	Reset	Description
31	INITREF_DIS	RW	1h	Refresh Disable. • 0 = Normal operation • 1 = Disables SDRAM refreshes, but carries out SDRAM write/read transactions
30-16	RESERVED	R	0h	Reserved.
15-0	REFRESH_RATE	RW	0h	The value in this field is used to define the rate at which connected SDRAM devices will be refreshed. REFRESH_RATE = Refresh period * DDR3 clock frequency.

Table 7-143. Register Call Summary for EMIF_SDRFC

EMIF Functional Description • SDRAM Refresh Scheduling: [0][1][2][3]
EMIF Registers • EMIF_SDRFC Register (Offset = 10h) [reset = 80000000h]: [0][1] • EMIF_ZQ_CONFIG Register (Offset = C8h) [reset = 0h]: [0] • EMIF_ONE_BIT_ECC_ERR_THRSH Register (Offset = 134h) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.5 EMIF_SDTIM1 Register (Offset = 18h) [reset = 3FFFFFFFh]

SDRAM Timing 1 register (EMIF_SDTIM1) configures the EMIF controller to meet many of the AC timing specifications of the DDR3 memory. Note that DDR3_CLKOUT is equal to the period of the DDR3_CLKOUT signal. See the DDR3 memory data sheet for information on appropriate values to program each field. The EMIF_SDTIM1 register is described in the following figure and table.

Table 7-144. EMIF_SDTIM1 Instances

Instance	Physical Address
EMIF	2101 0018h

Figure 7-51. EMIF_SDTIM1 Register

31	30	29	28	27	26	25	24
RESERVED		T_WR					T_RAS
R-0h		RW-1Fh					RW-7Fh
23	22	21	20	19	18	17	16
T_RAS						T_RC	
RW-7Fh						RW-FFh	
15	14	13	12	11	10	9	8
T_RC						T_RRD	
RW-FFh						RW-3Fh	
7	6	5	4	3	2	1	0
T_RRD				T_WTR			
RW-3Fh				RW-Fh			

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 7-145. EMIF_SDTIM1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect
29-25	T_WR	RW	1Fh	These bits specify the minimum number of DDR3_CLKOUT cycles from the last write transfer to a precharge command, minus 1. The value of this parameter can be derived from the t_{WR} AC timing parameter in the DDR3 memory data sheet. Calculate using the formula: $T\_WR = (t_{WR}/t_{CK}) - 1$
24-18	T_RAS	RW	7Fh	These bits specify the minimum number of DDR3_CLKOUT cycles from an activate command to precharge command, minus 1. The value of this parameter can be derived from the minimum value of the t_{RAS} AC timing parameter in the DDR3 memory data sheet. Calculate using the formula: $T\_RAS = (t_{RAS}/t_{CK}) - 1$
17-10	T_RC	RW	FFh	These bits specify the minimum number of DDR3_CLKOUT cycles from an activate command to an activate command, minus 1. The value of this parameter can be derived from the t_{RC}AC timing parameter in the DDR3 memory data sheet. Calculate using the formula: $T\_RC = (t_{RC}/t_{CK}) - 1$
9-4	T_RRD	RW	3Fh	These bits specify the minimum number of DDR3_CLKOUT cycles from an activate to an activate in a different bank, minus 1. The value of this parameter can be derived from the t_{FAW} AC timing parameter in the DDR3 memory data sheet. Calculate using the formula: $T\_RRD = (t_{FAW}/(4*t_{CK})) - 1$

Table 7-145. EMIF_SDTIM1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	T_WTR	RW	Fh	These bits specify the minimum number of DDR3_CLKOUT cycles from the last write to a read command, minus 1. The value of this parameter can be derived from the t_{WTR} AC timing parameter in the DDR3 memory data sheet. Convert the t_{WTR} value from the memory data sheet into ns before using the formula below. Calculate using the formula: $T\_WTR = (t_{WTR}/t_{CK}) - 1$

Table 7-146. Register Call Summary for EMIF_SDTIM1

EMIF Functional Description • Turnaround Time: [0][1]
EMIF Registers • EMIF_SDTIM2 Register (Offset = 1Ch) [reset = 1FFFh]: [0] • EMIF_SDTIM3 Register (Offset = 20h) [reset = FFFFFFFFh]: [0] • EMIF_SDTIM1 Register (Offset = 18h) [reset = 3FFFFFFFh]: [0][1] • EMIF Registers: [0]

7.2.5.6 EMIF_SDTIM2 Register (Offset = 1Ch) [reset = 1FFFh]

Like the SDRAM Timing 1 register (EMIF_SDTIM1), the SDRAM Timing 2 register (EMIF_SDTIM2) also configures the EMIF controller to meet many of the AC timing specifications of the DDR3 memory. See the DDR3 memory data sheet for information on appropriate values to program each field. The EMIF_SDTIM2 register is described in the following figure and table.

Table 7-147. EMIF_SDTIM2 Instances

Instance	Physical Address
EMIF	2101 001Ch

Figure 7-52. EMIF_SDTIM2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			T_RTW			T_RP	
R-0h			RW-7h			RW-1Fh	
7	6	5	4	3	2	1	0
T_RP			T_RCD				
RW-1Fh			RW-1Fh				

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 7-148. EMIF_SDTIM2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Value = 0 Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
12-10	T_RTW	RW	7h	Minimum number of DDR3_CLKOUT cycles between read and write date phases, minus 1. This is not an SDRAM timing parameter. The value depends on the board topology supported. For single rank: reg_t_RTW= (((2*dqs_delay_for_cs0)+tDQSCK+ wr_leveling_tolerance)/clock_period)-1 For dual rank: reg_t_RTW= ((dqs_delay_for_cs0+dqs_delay_for_cs1+ absolute(command_delay_for_cs0- command_delay_for_cs1)+tDQSCK+ wr_leveling_tolerance)/clock_period)-1
9-5	T_RP	RW	1Fh	These bits specify the minimum number of DDR3_CLKOUT cycles from a precharge command to a refresh or activate command, minus 1. The value of this parameter can be derived from the t_{RP} AC timing parameter in the DDR3 memory data sheet. Calculate using the formula: $T\_RP = (t_{RP}/t_{CK}) - 1$
4-0	T_RCD	RW	1Fh	These bits specify the minimum number of DDR3_CLKOUT cycles from an activate command to a read or write command, minus 1. The value of this parameter can be derived from the t_{RCD} AC timing parameter in the DDR3 memory data sheet. Calculate using the formula: $T\_RCD = (t_{RCD}/t_{CK}) - 1$

Table 7-149. Register Call Summary for EMIF_SDTIM2

EMIF Registers
• EMIF_SDTIM2 Register (Offset = 1Ch) [reset = 1FFFh]: [0][1] • EMIF_SDTIM3 Register (Offset = 20h) [reset = FFFFFFFFh]: [0] • EMIF Registers: [0]

7.2.5.7 EMIF_SDTIM3 Register (Offset = 20h) [reset = FFFFFFFFh]

Like the SDRAM Timing 1 and 2 registers (EMIF_SDTIM1 & EMIF_SDTIM2), the SDRAM Timing 3 register (EMIF_SDTIM3) also configures the EMIF controller to meet many of the AC timing specifications of the DDR3 memory. See the DDR3 memory data sheet for information on appropriate values to program each field. The EMIF_SDTIM3 register is described in the following figure and table.

Table 7-150. EMIF_SDTIM3 Instances

Instance	Physical Address
EMIF	2101 0020h

Figure 7-53. EMIF_SDTIM3 Register

31	30	29	28	27	26	25	24
T_XP				T_XSNR			
RW-Fh				RW-3FFh			
23	22	21	20	19	18	17	16
T_XSNR						T_XSRD	
RW-3FFh						RW-3FFh	
15	14	13	12	11	10	9	8
T_XSRD							
RW-3FFh							
7	6	5	4	3	2	1	0
T_RTP				T_CKE			
RW-Fh				RW-Fh			

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-151. EMIF_SDTIM3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	T_XP	RW	Fh	These bits specify the minimum number of DDR3_CLKOUT cycles from Power down exit to any command other than a read command, minus 1. The value of this parameter can be derived from the t_{XP} AC timing parameter in the DDR3 memory data sheet. Convert the t_{XP} data sheet parameter value into ns before using the formula below. Calculate using the formula: $T\_XP = (t_{XP}/t_{CK}) - 1$
27-18	T_XSNR	RW	3FFh	These bits specify the minimum number of DDR3_CLKOUT cycles from a self-refresh exit to a command that does not require a locked DLL, minus 1. The value of this parameter can be derived from the t_{XS} AC timing parameter in the DDR3 memory data sheet. Calculate using the formula: $T\_XSNR = (t_{XS}/t_{CK}) - 1$
17-8	T_XSRD	RW	3FFh	These bits specify the minimum number of DDR3_CLKOUT cycles from a self-refresh exit to a command that requires a locked DLL, minus 1. The value of this parameter can be derived from the t_{XSDLL} AC timing parameter in the DDR3 memory data sheet. This parameter typically appears in the data sheet in terms of t_{CK} clock cycles. Calculate using the formula: $T\_XSRD = t_{XSDLL} - 1$
7-4	T_RTP	RW	Fh	These bits specify the minimum number of DDR3_CLKOUT cycles from the last read to precharge command, minus 1. The value of this parameter can be derived from the t_{RTP} AC timing parameter in the DDR3 memory data sheet. Convert the t_{XP} data sheet parameter value into ns before using formula below. Calculate using the formula: $T\_RTP = (t_{RTP}/t_{CK}) - 1$

Table 7-151. EMIF_SDTIM3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	T_CKE	RW	Fh	These bits specify the minimum number of DDR3_CLKOUT cycles between transitions on the DDR3_CKE pin, minus 1. The value of this parameter can be derived from the t_{CKE} AC timing parameter in the DDR3 memory data sheet. Calculate using the formula: $T\_CKE = (t_{CKE}/t_{CK}) - 1$

Table 7-152. Register Call Summary for EMIF_SDTIM3

EMIF Functional Description • SDRAM Self-Refresh Mode: [0][1] • SDRAM Power-Down Mode: [0][1]
EMIF Registers • EMIF_SDTIM3 Register (Offset = 20h) [reset = FFFFFFFFh]: [0][1] • EMIF Registers: [0]

7.2.5.8 EMIF_SDTIM4 Register (Offset = 28h) [reset = FFFF3FFFh]

Like the SDRAM Timing 1, 2 and 3 registers, the SDRAM Timing 4 register (EMIF_SDTIM4) also configures the EMIF controller to meet many of the AC timing specifications of the DDR3 memory. See the DDR3 memory data sheet for information on appropriate values to program each field. The EMIF_SDTIM4 register is described in the following figure and table.

Table 7-153. EMIF_SDTIM4 Instances

Instance	Physical Address
EMIF	2101 0028h

Figure 7-54. EMIF_SDTIM4 Register

31	30	29	28	27	26	25	24
T_CSTA				T_CKESR			
RW-Fh				RW-Fh			
23	22	21	20	19	18	17	16
ZQ_ZQCS							
RW-FFh							
15	14	13	12	11	10	9	8
RESERVED		T_RFC					
R-0h		RW-3FFh					
7	6	5	4	3	2	1	0
T_RFC				T_RAS_MAX			
RW-3FFh				RW-Fh			

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-154. EMIF_SDTIM4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	T_CSTA	RW	Fh	Minimum DDR3_CLKOUT cycles between write-to-write or read-to-read data phases to different chip selects, minus 1.
27-24	T_CKESR	RW	Fh	Value = 0 Minimum DDR3_CLKOUT cycles for which DDR3 should remain in self-refresh. This parameter typically appears as number of t_{CK} clock cycles. T_CKESR = (t_{CKESR}/t_{CK}) – 1
23-16	ZQ_ZQCS	RW	FFh	These bits specify the minimum number of DDR3_CLKOUT cycles for a ZQCS command, minus 1. The value of this parameter can be derived from the t_{ZQCS} AC timing parameter in the DDR3 memory data sheet. This parameter typically appears as number of t_{CK} clock cycles. Calculate using the formula : ZQ_ZQCS = t_{ZQCS} – 1
15-14	RESERVED	R	0h	Value = 0 Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
13-4	T_RFC	RW	3FFh	These bits specify the minimum number of DDR3_CLKOUT cycles from a refresh or load mode command to a refresh or activate command, minus 1. The value of this parameter can be derived from the t_{RFC} AC timing parameter in the DDR3 memory data sheet. Calculate using the formula: T_RFC = (t_{RFC}/t_{CK}) – 1
3-0	T_RAS_MAX	RW	Fh	This field must always be programmed to Fh.

Table 7-155. Register Call Summary for EMIF_SDTIM4

EMIF Functional Description • SDRAM Refresh Scheduling: [0] • Turnaround Time: [0][1] • DDR3L Output Impedance Calibration: [0][1]
EMIF Registers • EMIF_ZQ_CONFIG Register (Offset = C8h) [reset = 0h]: [0] • EMIF_SDTIM4 Register (Offset = 28h) [reset = FFFF3FFFh]: [0][1] • EMIF Registers: [0]

7.2.5.9 EMIF_PMCTL Register (Offset = 38h) [reset = 0h]

The EMIF_PMCTL register is described in the following figure and table.

Table 7-156. EMIF_PMCTL Instances

Instance	Physical Address
EMIF	2101 0038h

Figure 7-55. EMIF_PMCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
PD_TIM				RESERVED	LP_MODE		
RW-0h				R-0h	RW-0h		
7	6	5	4	3	2	1	0
SR_TIM				RESERVED			
RW-0h				R-0h			

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-157. EMIF_PMCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect
15-12	PD_TIM	RW	0h	Power Management timer for power-down. The EMIF controller puts the SDRAM in power-down mode after the EMIF controller is idle for PD_TIM number of DDR3_CLKOUT cycles and if LP_MODE is set to 4. • 0 = Immediately enter power-down • 1 = Enter power-down after 16 clocks • 2 = Enter power-down after 32 clocks • 3 = Enter power-down after 64 clocks • 4 = Enter power-down after 128 clocks • 5 = Enter power-down after 256 clocks • 6 = Enter power-down after 512 clocks • 7 = Enter power-down after 1024 clocks • 8 = Enter power-down after 2048 clocks • 9 = Enter power-down after 4096 clocks • 10 = Enter power-down after 8912 clocks • 11 = Enter power-down after 16384 clocks • 12 = Enter power-down after 32768 clocks • 13 = Enter power-down after 65536 clocks • 14 = Enter power-down after 131072 clocks • 15 = Enter power-down after 262144 clocks
11	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect
10-8	LP_MODE	RW	0h	Automatic power management enable. • 2 = Self-refresh mode • 4 = Power-down mode All other values disable automatic power management.

Table 7-157. EMIF_PMCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-4	SR_TIM	RW	0h	Power management timer for self-refresh. The EMIF controller will put the external SDRAM in self-refresh mode after EMIF controller has been idle for these number of DDR3_CLKOUT cycles and LP_MODE is set to 2. • 0 = Immediately enter self-refresh • 1 = Enter self-refresh after 16 clocks • 2 = Enter self-refresh after 32 clocks • 3 = Enter self-refresh after 64 clocks • 4 = Enter self-refresh after 128 clocks • 5 = Enter self-refresh after 256 clocks • 6 = Enter self-refresh after 512 clocks • 7 = Enter self-refresh after 1024 clocks • 8 = Enter self-refresh after 2048 clocks • 9 = Enter self-refresh after 4096 clocks • 10 = Enter self-refresh after 8912 clocks • 11 = Enter self-refresh after 16384 clocks • 12 = Enter self-refresh after 32768 clocks • 13 = Enter self-refresh after 65536 clocks • 14 = Enter self-refresh after 131072 clocks • 15 = Enter self-refresh after 262144 clocks
3-0	RESERVED	R	0h	Reserved

Table 7-158. Register Call Summary for EMIF_PMCTL

EMIF Functional Description • SDRAM Self-Refresh Mode: [0][1][2][3] • SDRAM Power-Down Mode: [0][1][2] • SDRAM Extended Temperature Range: [0]
EMIF Registers • EMIF_PMCTL Register (Offset = 38h) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.10 EMIF_VBUSM_CONFIG Register (Offset = 54h) [reset = FFFFFFh]

The EMIF_VBUSM_CONFIG register is described in the following figure and table.

Table 7-159. EMIF_VBUSM_CONFIG Instances

Instance	Physical Address
EMIF	2101 0054h

Figure 7-56. EMIF_VBUSM_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
COS_COUNT_1							
RW-FFh							
15	14	13	12	11	10	9	8
COS_COUNT_2							
RW-FFh							
7	6	5	4	3	2	1	0
PR_OLD_COUNT							
RW-FFh							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-160. EMIF_VBUSM_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
23-16	COS_COUNT_1	RW	FFh	Priority raise counter for Class of Service 1. Number of DDR3_CLKOUT cycles after which the EMIF controller momentarily raises the priority of Class of Service 1 commands in Command FIFO. Number of clock cycles = COS_COUNT_1 x 16 clocks
15-8	COS_COUNT_2	RW	FFh	Priority raise counter for Class of Service 2. Number of DDR3_CLKOUT cycles after which the EMIF controller momentarily raises the priority of Class of Service 2 commands in Command FIFO. Number of clock cycles = COS_COUNT_2 x 16 clocks
7-0	PR_OLD_COUNT	RW	FFh	Priority raise old counter. Number of DDR3_CLKOUT cycles after which EMIF controller momentarily raises the priority of the oldest command in Command FIFO. Number of clock cycles = PR_OLD_COUNT_2 x 16 clocks

Table 7-161. Register Call Summary for EMIF_VBUSM_CONFIG

EMIF Functional Description • Command Starvation: [0] • Class of Service of Commands in the Command FIFO: [0][1][2]
EMIF Registers • EMIF_VBUSM_CONFIG Register (Offset = 54h) [reset = FFFFFFh]: [0] • EMIF Registers: [0]

7.2.5.11 EMIF_PERF_CNT_1 Register (Offset = 80h) [reset = 0h]

The EMIF_PERF_CNT_1 register is described in the following figure and table.

Table 7-162. EMIF_PERF_CNT_1 Instances

Instance	Physical Address
EMIF	2101 0080h

Figure 7-57. EMIF_PERF_CNT_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTER_1																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 7-163. EMIF_PERF_CNT_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNTER_1	R	0h	32-bit counter can be programmed as specified in the Performance Counter Config and Performance Counter Master Region Select registers.

Table 7-164. Register Call Summary for EMIF_PERF_CNT_1

EMIF Functional Description • Performance Monitoring: [0] • : [2][3]
EMIF Registers • EMIF_PERF_CNT_1 Register (Offset = 80h) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.12 EMIF_PERF_CNT_2 Register (Offset = 84h) [reset = 0h]

The EMIF_PERF_CNT_2 register is described in the following figure and table.

Table 7-165. EMIF_PERF_CNT_2 Instances

Instance	Physical Address
EMIF	2101 0084h

Figure 7-58. EMIF_PERF_CNT_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTER_2																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 7-166. EMIF_PERF_CNT_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNTER_2	R	0h	32-bit counter can be programmed as specified in the Performance Counter Config and Performance Counter Master Region Select registers.

Table 7-167. Register Call Summary for EMIF_PERF_CNT_2

EMIF Functional Description • Performance Monitoring: [0] • Performance Counters General Examples: [0][1]
EMIF Registers • EMIF_PERF_CNT_2 Register (Offset = 84h) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.13 EMIF_PERF_CNT_CFG Register (Offset = 88h) [reset = 10000h]

The EMIF_PERF_CNT_CFG register is described in the following figure and table.

Table 7-168. EMIF_PERF_CNT_CFG Instances

Instance	Physical Address
EMIF	2101 0088h

Figure 7-59. EMIF_PERF_CNT_CFG Register

31	30	29	28	27	26	25	24
CNTR2_MSTID_EN	CNTR2_REGION_EN	RESERVED					
RW-0h	RW-0h	R-0h					
23	22	21	20	19	18	17	16
RESERVED				CNTR2_CFG			
R-0h				RW-1h			
15	14	13	12	11	10	9	8
CNTR1_MSTID_EN	CNTR1_REGION_EN	RESERVED					
RW-0h	RW-0h	R-0h					
7	6	5	4	3	2	1	0
RESERVED				CNTR1_CFG			
R-0h				RW-0h			

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-169. EMIF_PERF_CNT_CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CNTR2_MSTID_EN	RW	0h	Master ID filter enable for Performance Counter 2 Register. Set to 1 to enable filtering of events for counter 2 by Master ID.
30	CNTR2_REGION_EN	RW	0h	Memory space region enable for Performance Counter 2 Register. Set to 1 to enable filtering of events for counter 2 by the accessed memory region.
29-20	RESERVED	R	0h	Reserved
19-16	CNTR2_CFG	RW	1h	Filter configuration selected for Performance Counter 2. This field selects the type of event to count for Counter 2. Refer to Table 7-129 for various configuration options.
15	CNTR1_MSTID_EN	RW	0h	Master ID filter enable for Performance Counter 1 Register. Set to 1 to enable filtering of events for counter 1 by Master ID.
14	CNTR1_REGION_EN	RW	0h	Memory space region enable for Performance Counter 1 Register. Set to 1 to enable filtering of events for counter 1 by the accessed memory region.
13-4	RESERVED	R	0h	Reserved
3-0	CNTR1_CFG	RW	0h	Filter configuration selected for Performance Counter 1. This field selects the type of event to count for Counter 1. Refer to Table 7-129 for various configuration options.

Table 7-170. Register Call Summary for EMIF_PERF_CNT_CFG

EMIF Functional Description • Performance Monitoring: [0][1][2][3][4][5] • : [2][3][4][5][6][7]
EMIF Registers • EMIF_PERF_CNT_CFG Register (Offset = 88h) [reset = 10000h]: [0] • EMIF Registers: [0]

7.2.5.14 EMIF_PERF_CNT_SEL Register (Offset = 8Ch) [reset = 0h]

For events that can be configured to enable master ID, memory region filters or both, the value of the master ID and the region select options for the counters are programmed in the EMIF_PERF_CNT_SEL register. The EMIF_PERF_CNT_SEL register is described in the following figure and table.

Table 7-171. EMIF_PERF_CNT_SEL Instances

Instance	Physical Address
EMIF	2101 008Ch

Figure 7-60. EMIF_PERF_CNT_SEL Register

31	30	29	28	27	26	25	24
MSTID2							
RW-0h							
23	22	21	20	19	18	17	16
RESERVED				REGION_SEL2			
R-0h				RW-0h			
15	14	13	12	11	10	9	8
MSTID1							
RW-0h							
7	6	5	4	3	2	1	0
RESERVED				REGION_SEL1			
R-0h				RW-0h			

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-172. EMIF_PERF_CNT_SEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	MSTID2	RW	0h	Master ID for Performance Counter 2 Register.
23-20	RESERVED	R	0h	Reserved
19-16	REGION_SEL2	RW	0h	Region select for Performance Counter 2. • 0h - DDR3 memory space • 7h - EMIF configuration registers All other values are reserved.
15-8	MSTID1	RW	0h	Master ID for Performance Counter 1 Register.
7-4	RESERVED	R	0h	Reserved
3-0	REGION_SEL1	RW	0h	Region select for Performance Counter 1. • 0h - DDR3 memory space • 7h - EMIF configuration registers All other values are reserved.

Table 7-173. Register Call Summary for EMIF_PERF_CNT_SEL

EMIF Functional Description • Performance Monitoring: [0] • : [1][2]
EMIF Registers • EMIF_PERF_CNT_SEL Register (Offset = 8Ch) [reset = 0h]: [0][1] • EMIF Registers: [0]

7.2.5.15 EMIF_PERF_CNT_TIM Register (Offset = 90h) [reset = 0h]

The EMIF_PERF_CNT_TIM register is described in the following figure and table.

Table 7-174. EMIF_PERF_CNT_TIM Instances

Instance	Physical Address
EMIF	2101 0090h

Figure 7-61. EMIF_PERF_CNT_TIM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOTAL_TIME																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 7-175. EMIF_PERF_CNT_TIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TOTAL_TIME	R	0h	32-bit counter continuously counts number of DDR/2 clock cycles elapsed after the controller is brought out of reset.

Table 7-176. Register Call Summary for EMIF_PERF_CNT_TIM

EMIF Registers
- EMIF_PERF_CNT_TIM Register (Offset = 90h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.16 EMIF_IRQ_EOI Register (Offset = A0h) [reset = 0h]

The EMIF_IRQ_EOI register is described in the following figure and table.

Table 7-177. EMIF_IRQ_EOI Instances

Instance	Physical Address
EMIF	2101 00A0h

Figure 7-62. EMIF_IRQ_EOI Register

31	30	29	28	27	26	25	24
RESERVED							
RW-0h							
23	22	21	20	19	18	17	16
RESERVED							
RW-0h							
15	14	13	12	11	10	9	8
RESERVED							
RW-0h							
7	6	5	4	3	2	1	0
RESERVED							EOI
RW-0h							RW-0h

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-178. EMIF_IRQ_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	RW	0h	Reserved.
0	EOI	RW	0h	Software End Of Interrupt (EOI) control. Write 0 for system VBUSM interrupt. This field always reads 0 (no EOI memory).

Table 7-179. Register Call Summary for EMIF_IRQ_EOI

EMIF Registers
- EMIF_IRQ_EOI Register (Offset = A0h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.17 EMIF_IRQSTATUS_RAW_SYS Register (Offset = A4h) [reset = 0h]

The EMIF_IRQSTATUS_RAW_SYS register is described in the following figure and table.

Table 7-180. EMIF_IRQSTATUS_RAW_SYS Instances

Instance	Physical Address
EMIF	2101 00A4h

Figure 7-63. EMIF_IRQSTATUS_RAW_SYS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		1B_ECC_ERR_SYS	2B_ECC_ERR_SYS	WR_ECC_ERR_SYS	RESERVED		ERR_SYS
R-0h		WOS-0h	WOS-0h	WOS-0h	R-0h		WOS-0h

LEGEND: R = Read Only; WOS = Write 1 to Set Bit; *-n* = value after reset

Table 7-181. EMIF_IRQSTATUS_RAW_SYS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Value = 0h Reserved
5	1B_ECC_ERR_SYS	WOS	0h	Value = 0h Raw status of 1-bit ECC error interrupt. Writing a 1 sets the raw status. Writing a 0 has no effect.
4	2B_ECC_ERR_SYS	WOS	0h	Value = 0h Raw status of 2-bit ECC error interrupt. Writing a 1 sets the raw status. Writing a 0 has no effect.
3	WR_ECC_ERR_SYS	WOS	0h	Value = 0h Raw status of write ECC error interrupt. Writing a 1 sets the raw status. Writing a 0 has no effect.
2-1	RESERVED	R	0h	Value = 0h Reserved
0	ERR_SYS	WOS	0h	Value = 0h Raw status of system VBUSM interrupt for command or address error. Writing a 1 sets the raw status. Writing a 0 has no effect.

Table 7-182. Register Call Summary for EMIF_IRQSTATUS_RAW_SYS

EMIF Functional Description • Interrupt Requests: [0][1][2][3][4][5][6][7][8][9]
EMIF Registers • EMIF_IRQSTATUS_RAW_SYS Register (Offset = A4h) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.18 EMIF_IRQSTATUS_SYS Register (Offset = ACh) [reset = 0h]

The EMIF_IRQSTATUS_SYS register is described in the following figure and table.

Table 7-183. EMIF_IRQSTATUS_SYS Instances

Instance	Physical Address
EMIF	2101 00ACh

Figure 7-64. EMIF_IRQSTATUS_SYS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		1B_ECC_ERR_SYS	2B_ECC_ERR_SYS	WR_ECC_ERR_SYS	RESERVED		ERR_SYS
R-0h		WOC-0h	WOC-0h	WOC-0h	R-0h		WOC-0h

LEGEND: R = Read Only; WOC = Write 1 to Clear Bit; *-n* = value after reset

Table 7-184. EMIF_IRQSTATUS_SYS Register Field Descriptions

Bit	Field	Type	Reset	Description
31- 6	RESERVED	R	0h	Value = 0h Reserved.
5	1B_ECC_ERR_SYS	WOC	0h	Value = 0h Enabled status of 1-bit ECC error interrupt. Writing a 1 clears the status as well the raw status. Writing a 0 has no effect.
4	2B_ECC_ERR_SYS	WOC	0h	Value = 0h Enabled status of 2-bit ECC error interrupt. Writing a 1 clears the status as well the raw status. Writing a 0 has no effect.
3	WR_ECC_ERR_SYS	WOC	0h	Value = 0h Enabled status of write ECC error interrupt. Writing a 1 clears the status as well the raw status. Writing a 0 has no effect.
2-1	RESERVED	R	0h	Value = 0h Reserved.
0	ERR_SYS	WOC	0h	Value = 0h Enabled status of system VBUSM interrupt for command or address error. Writing a 1 clears the status as well the raw status. Writing a 0 has no effect.

Table 7-185. Register Call Summary for EMIF_IRQSTATUS_SYS

EMIF Functional Description • Interrupt Requests: [0][1][2][3][4][5][6][7]
EMIF Registers • EMIF_IRQSTATUS_SYS Register (Offset = ACh) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.19 EMIF_IRQENABLE_SET_SYS Register (Offset = B4h) [reset = 0h]

The IRQSTATUS_SET_SYS register is described in the following figure and table.

Table 7-186. EMIF_IRQENABLE_SET_SYS Instances

Instance	Physical Address
EMIF	2101 00B4h

Figure 7-65. EMIF_IRQENABLE_SET_SYS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		1B_ECC_ERR_SYS	2B_ECC_ERR_SYS	WR_ECC_ERR_SYS	RESERVED		ERR_SYS
R-0h		WOS-0h	WOS-0h	WOS-0h	R-0h		WOS-0h

LEGEND: R = Read Only; WOS = Write 1 to Set Bit; *-n* = value after reset

Table 7-187. EMIF_IRQENABLE_SET_SYS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Value = 0h Reserved.
5	1B_ECC_ERR_SYS	WOS	0h	Value = 0h Enabled set for 1-bit ECC error interrupt. Writing a 1 will enable the read ECC error interrupt, set this bit as well as the bit in Interrupt Enable Clear Register. Writing a 0 has no effect.
4	2B_ECC_ERR_SYS	WOS	0h	Value = 0h Enabled set for 2-bit ECC error interrupt. Writing a 1 will enable the read ECC error interrupt, set this bit as well as the bit in Interrupt Enable Clear Register. Writing a 0 has no effect.
3	WR_ECC_ERR_SYS	WOS	0h	Value = 0h Enabled set for write ECC error interrupt. Writing a 1 will enable the write ECC error interrupt, set this bit as well as the bit in Interrupt Enable Clear Register. Writing a 0 has no effect.
2-1	RESERVED	R	0h	Value = 0h Reserved.
0	ERR_SYS	WOS	0h	Value = 0h Enable set for system VBUSM interrupt for command or address error. Writing a 1 will enable the interrupt, and set this bit as well as the corresponding Interrupt Enable Clear Register. Writing a 0 has no effect.

Table 7-188. Register Call Summary for EMIF_IRQENABLE_SET_SYS

EMIF Functional Description • Interrupt Requests: [0][1][2][3][4][5][6]
EMIF Registers • EMIF Registers: [0]

7.2.5.20 EMIF_IRQENABLE_CLR_SYS Register (Offset = BCh) [reset = 0h]

The IRQSTATUS_CLR_SYS register is described in the following figure and table.

Table 7-189. EMIF_IRQENABLE_CLR_SYS Instances

Instance	Physical Address
EMIF	2101 00BCh

Figure 7-66. EMIF_IRQENABLE_CLR_SYS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		1B_ECC_ERR_SYS	2B_ECC_ERR_SYS	WR_ECC_ERR_SYS	RESERVED		ERR_SYS
R-0h		WOC-0h	WOC-0h	WOC-0h	R-0h		WOC-0h

LEGEND: R = Read Only; WOC = Write 1 to Clear Bit; *-n* = value after reset

Table 7-190. EMIF_IRQENABLE_CLR_SYS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Value = 0h Reserved.
5	1B_ECC_ERR_SYS	WOC	0h	Value = 0h Enabled clear for 1-bit ECC error interrupt. Writing a 1 will disable the read ECC error interrupt, clear this bit as well as the bit in Interrupt Enable Clear Register. Writing a 0 has no effect.
4	2B_ECC_ERR_SYS	WOC	0h	Value = 0h Enabled clear for 2-bit ECC error interrupt. Writing a 1 will disable the read ECC error interrupt, clear this bit as well as the bit in Interrupt Enable Clear Register. Writing a 0 has no effect.
3	WR_ECC_ERR_SYS	WOC	0h	Value = 0h Enabled clear for write ECC error interrupt. Writing a 1 will disable the write ECC error interrupt, clear this bit as well as the bit in Interrupt Enable Clear Register. Writing a 0 has no effect.
2-1	RESERVED	R	0h	Value = 0h Reserved.
0	ERR_SYS	WOC	0h	Value = 0h Enable clear for system VBUSM interrupt for command or address error. Writing a 1 will disable the interrupt, and clear this bit as well as the corresponding Interrupt Enable Set Register. Writing a 0 has no effect.

Table 7-191. Register Call Summary for EMIF_IRQENABLE_CLR_SYS

EMIF Functional Description • Interrupt Requests: [0][1][2][3][4][5]
EMIF Registers • EMIF Registers: [0]

7.2.5.21 EMIF_ZQ_CONFIG Register (Offset = C8h) [reset = 0h]

The EMIF_ZQ_CONFIG register is described in the following figure and table.

Table 7-192. EMIF_ZQ_CONFIG Instances

Instance	Physical Address
EMIF	2101 00C8h

Figure 7-67. EMIF_ZQ_CONFIG Register

31	30	29	28	27	26	25	24
ZQ_CS1EN	ZQ_CS0EN	ZQ_DUALCALEN	ZQ_SFEXITEN	RESERVED			
RW-0h	RW-0h	RW-0h	RW-0h	R-0h			
23	22	21	20	19	18	17	16
RESERVED					ZQ_ZQCL_MULT		
R-0h					RW-0h		
15	14	13	12	11	10	9	8
ZQ_REFINTERVAL							
RW-0h							
7	6	5	4	3	2	1	0
ZQ_REFINTERVAL							
RW-0h							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-193. EMIF_ZQ_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ZQ_CS1EN	RW	0h	ZQ calibration for CS1 • 0 = Disable ZQ calibration for CS1 • 1 = Enable ZQ calibration for CS1
30	ZQ_CS0EN	RW	0h	ZQ calibration for CS0 • 0 = Disable ZQ calibration for CS0 • 1 = Enable ZQ calibration for CS0
29	ZQ_DUALCALEN	RW	0h	ZQ Dual Calibration enable. Allows both ranks to be calibrated simultaneously. • 0 = Dual ZQ calibration disable. NOTE: This is not supported. This bit must be set 1 if calibration is needed. • 1 = Both chip selects have a separate calibration resistor per device.
28	ZQ_SFEXITEN	RW	0h	ZQCL on Self-refresh, Active power-down and precharge power-down exit enable. • 0 = Disable ZQCL on Self-refresh, Active power-down and precharge power-down exit enable • 1 = Enable ZQCL on Self-refresh, Active power-down and precharge power-down exit enable. Set this value to 1 to issue a ZQCL command upon self-refresh exit.
27-19	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
18-16	ZQ_ZQCL_MULT	RW	0h	Number of ZQCS intervals that make up a ZQCL interval, minus one. ZQCS interval is defined by the EMIF_SDTIM4[23-16] ZQ_ZQCS field. The value of this parameter can be derived using the formula: T_ZQ_ZQCL_MULT = (t_{ZQoper}/t_{ZQCS} – 1)

Table 7-193. EMIF_ZQ_CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-0	ZQ_REFINTERVAL	RW	0h	Number of refresh periods between ZQCS commands, minus one. This field supports between one refresh period to 256 ms between ZQCS calibration commands. Refresh period is defined by the EMIF_SDRFC[15-0] REFRESH_RATE field. ZQ_REFINTERVAL = number of refresh periods between ZQCS commands. The interval is calculated as = 0.5% / [(Tsens x Tdriftrate) + (Vsens x Vdriftrate)]. Tsens = max (dRTTdT, dRONdTM) from the memory device data sheet. Vsens = max(dRTTdV, dRONdVM) from the memory device data sheet Tdriftrate = drift rate in ^{o} C/second. This is the temperature drift rate that the SDRAM is subject to in the application. Vdriftrate = drift rate in mV/second. This is the voltage drift rate that the SDRAM is subject to in the application. Example: If Tsens= 1.5%/ ^{o} C, Vsens = 0.15%/mV, Tdriftrate = 1.2 ^{o} C/second and Vdriftrate = 10mV/second, Interval = 0.5/[(1.5 x 1.2) + (0.15 x 10)] = 152ms. Since refresh interval = 7.8µs, ZQ_REFINTERVAL = 152ms/7.8µs = 4C1Fh

Table 7-194. Register Call Summary for EMIF_ZQ_CONFIG

EMIF Functional Description • SDRAM Self-Refresh Mode: [0] • DDR3L Output Impedance Calibration: [0][1][2][3][4][5]
EMIF Registers • EMIF_ZQ_CONFIG Register (Offset = C8h) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.22 EMIF_PRI_COS_MAP Register (Offset = 100h) [reset = 0h]

The EMIF_PRI_COS_MAP register is described in the following figure and table.

Table 7-195. EMIF_PRI_COS_MAP Instances

Instance	Physical Address
EMIF	2101 0100h

Figure 7-68. EMIF_PRI_COS_MAP Register

31	30	29	28	27	26	25	24
PRI_COS_MAP_EN	RESERVED						
RW-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
PRI_7_COS		PRI_6_COS		PRI_5_COS		PRI_4_COS	
RW-0h		RW-0h		RW-0h		RW-0h	
7	6	5	4	3	2	1	0
PRI_3_COS		PRI_2_COS		PRI_1_COS		PRI_0_COS	
RW-0h		RW-0h		RW-0h		RW-0h	

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-196. EMIF_PRI_COS_MAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31	PRI_COS_MAP_EN	RW	0h	Priority to Class-of-service mapping • 0 = Disable Priority to Class-of-service mapping • 1 = Enable Priority to Class-of-service mapping
30-16	RESERVED	R	0h	Reserved. The reserved location is always read as 0. A value written to this field has no effect.
15-14	PRI_7_COS	RW	0h	Class-of-service for commands with priority of 7 (lowest priority). • 1= Map to Class-of-service 1 • 2= Map to Class-of-service 2 • 0 or 3 does not assign any class of service.
13-12	PRI_6_COS	RW	0h	Class-of-service for commands with priority of 6. • 1= Map to Class-of-service 1 • 2= Map to Class-of-service 2 • 0 or 3 does not assign any class of service.
11-10	PRI_5_COS	RW	0h	Class-of-service for commands with priority of 5. • 1= Map to Class-of-service 1 • 2= Map to Class-of-service 2 • 0 or 3 does not assign any class of service.
9-8	PRI_4_COS	RW	0h	Class-of-service for commands with priority of 4. • 1= Map to Class-of-service 1 • 2= Map to Class-of-service 2 • 0 or 3 does not assign any class of service.
7-6	PRI_3_COS	RW	0h	Class-of-service for commands with priority of 3. • 1= Map to Class-of-service 1 • 2= Map to Class-of-service 2 • 0 or 3 does not assign any class of service.

Table 7-196. EMIF_PRI_COS_MAP Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	PRI_2_COS	RW	0h	Class-of-service for commands with priority of 2. • 1= Map to Class-of-service 1 • 2= Map to Class-of-service 2 • 0 or 3 does not assign any class of service.
3-2	PRI_1_COS	RW	0h	Class-of-service for commands with priority of 1. • 1= Map to Class-of-service 1 • 2= Map to Class-of-service 2 • 0 or 3 does not assign any class of service.
1-0	PRI_0_COS	RW	0h	Class-of-service for commands with priority of 0 (highest priority). • 1= Map to Class-of-service 1 • 2= Map to Class-of-service 2 • 0 or 3 does not assign any class of service.

Table 7-197. Register Call Summary for EMIF_PRI_COS_MAP

EMIF Functional Description • Class of Service of Commands in the Command FIFO: [0]
EMIF Registers • EMIF_PRI_COS_MAP Register (Offset = 100h) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.23 EMIF_MSTID_COS_1_MAP Register (Offset = 104h) [reset = 0h]

The EMIF_MSTID_COS_1_MAP register is described in the following figure and table.

Table 7-198. EMIF_MSTID_COS_1_MAP Instances

Instance	Physical Address
EMIF	2101 0104h

Figure 7-69. EMIF_MSTID_COS_1_MAP Register

31	30	29	28	27	26	25	24
MSTID_COS_1_MAP_EN	MSTID_1_COS_1						
RW-0h	RW-0h						
23	22	21	20	19	18	17	16
MSTID_1_COS_1	MSK_1_COS_1			MSTID_2_COS_1			
RW-0h	RW-0h			RW-0h			
15	14	13	12	11	10	9	8
MSTID_2_COS_1				MSK_2_COS_1		MSTID_3_COS_1	
RW-0h				RW-0h		RW-0h	
7	6	5	4	3	2	1	0
MSTID_3_COS_1						MSK_3_COS_1	
RW-0h						RW-0h	

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-199. EMIF_MSTID_COS_1_MAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MSTID_COS_1_MAP_EN	RW	0h	Master ID to Class-of-service 1 mapping. • 0 = Disable Master ID to Class-of-service 1 mapping • 1 = Enable Master ID to Class-of-service 1 mapping
30-23	MSTID_1_COS_1	RW	0h	Master ID value 1 for Class-of-service 1.
22-20	MSK_1_COS_1	RW	0h	Mask for master ID value 1 for Class-of-service 1. • 0 = Disable masking • 1 = Mask master ID bit 0 • 2 = Mask master ID bits 1-0 • 3 = Mask master ID bits 2-0
19-12	MSTID_2_COS_1	RW	0h	Master ID value 2 for Class-of-service 1.
11-10	MSK_2_COS_1	RW	0h	Mask for master ID value 2 for Class-of-service 1. • 0 = Disable masking • 1 = Mask master ID bit 0 • 2 = Mask master ID bits 1-0 • 3 = Mask master ID bits 2-0
9-2	MSTID_3_COS_1	RW	0h	Master ID value 3 for Class-of-service 1.
1-0	MSK_3_COS_1	RW	0h	Mask for master ID value 3 for Class-of-service 1. • 0 = Disable masking • 1 = Mask master ID bit 0 • 2 = Mask master ID bits 1-0 • 3 = Mask master ID bits 2-0

Table 7-200. Register Call Summary for EMIF_MSTID_COS_1_MAP

EMIF Functional Description • Class of Service of Commands in the Command FIFO: [0]
EMIF Registers • EMIF_MSTID_COS_1_MAP Register (Offset = 104h) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.24 EMIF_MSTID_COS_2_MAP Register (Offset = 108h) [reset = 0h]

The EMIF_MSTID_COS_2_MAP register is described in the following figure and table.

Table 7-201. EMIF_MSTID_COS_2_MAP Instances

Instance	Physical Address
EMIF	2101 0108h

Figure 7-70. EMIF_MSTID_COS_2_MAP Register

31	30	29	28	27	26	25	24
MSTID_COS_2_MAP_EN	MSTID_1_COS_2						
RW-0h	RW-0h						
23	22	21	20	19	18	17	16
MSTID_1_COS_2	MSK_1_COS_2			MSTID_2_COS_2			
RW-0h	RW-0h			RW-0h			
15	14	13	12	11	10	9	8
MSTID_2_COS_2				MSK_2_COS_2		MSTID_3_COS_2	
RW-0h				RW-0h		RW-0h	
7	6	5	4	3	2	1	0
MSTID_3_COS_2						MSK_3_COS_2	
RW-0h						RW-0h	

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-202. EMIF_MSTID_COS_2_MAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MSTID_COS_2_MAP_EN	RW	0h	Master ID to Class-of-service 2 mapping. • 0 = Disable Master ID to Class-of-service 2 mapping • 1 = Enable Master ID to Class-of-service 2 mapping
30-23	MSTID_1_COS_2	RW	0h	Master ID value 1 for Class-of-service 2.
22-20	MSK_1_COS_2	RW	0h	Mask for master ID value 1 for Class-of-service 2. • 0 = Disable masking • 1 = Mask master ID bit 0 • 2 = Mask master ID bits 1-0 • 3 = Mask master ID bits 2-0
19-12	MSTID_2_COS_2	RW	0h	Master ID value 2 for Class-of-service 2.
11-10	MSK_2_COS_2	RW	0h	Mask for master ID value 2 for Class-of-service 2. • 0 = Disable masking • 1 = Mask master ID bit 0 • 2 = Mask master ID bits 1-0 • 3 = Mask master ID bits 2-0
9-2	MSTID_3_COS_2	RW	0h	Master ID value 3 for Class-of-service 2.
1-0	MSK_3_COS_2	RW	0h	Mask for master ID value 3 for Class-of-service 2. • 0 - Disable masking • 1 = Mask master ID bit 0 • 2 = Mask master ID bits 1-0 • 3 = Mask master ID bits 2-0

Table 7-203. Register Call Summary for EMIF_MSTID_COS_2_MAP

EMIF Functional Description • Class of Service of Commands in the Command FIFO: [0]
EMIF Registers • EMIF_MSTID_COS_2_MAP Register (Offset = 108h) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.25 EMIF_ECCCTL Register (Offset = 110h) [reset = 0h]

The EMIF_ECCCTL register is described in the following figure and table.

Table 7-204. EMIF_ECCCTL Instances

Instance	Physical Address
EMIF	2101 0110h

Figure 7-71. EMIF_ECCCTL Register

31	30	29	28	27	26	25	24
ECC_EN	ECC_ADDR_RNG_PROT	ECC_VERIFY_EN	RMW_EN	RESERVED			
RW-0h	RW-0h	RW-0h	RW-0h	R-0h			
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ECC_ADDR_RNG_2_EN	ECC_ADDR_RNG_1_EN
R-0h						RW-0h	RW-0h

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-205. EMIF_ECCCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ECC_EN	RW	0h	ECC enable. • 0 = Disable ECC • 1 = Enable ECC **NOTE:** If ECC is used this bit must be set to 1h.
30	ECC_ADDR_RNG_PROT	RW	0h	This bit is used to determine whether ECC calculation is allowed within address ranges described by ECC Address Range 1 and 2 Registers, provided ECC_EN is set to enable ECC. • 0 = Disable ECC calculation within address ranges defined in ECC Address Range 1 and 2 registers and enable calculation for accesses outside of these address ranges • 1 = Enable ECC calculation within address ranges defined in ECC Address Range 1 and 2 registers and disable calculation for accesses outside of these address ranges
29	ECC_VERIFY_EN	RW	0h	• 0 - Disable ECC verification for read accesses when ECC_EN=1. • 1- Enable ECC verification for read accesses when ECC_EN=1. This field is ignored if ECC_EN=0.
28	RMW_EN	RW	0h	• 0 - Disable read-modify-write functionality for sub-quanta accesses when ECC_EN=1. • 1 - Enable read-modify-write functionality for sub-quanta accesses when ECC_EN=1. This field is ignored if ECC_EN=0. **NOTE:** If ECC is used this bit must be set to 1h.
27-2	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
1	ECC_ADDR_RNG_2_EN	RW	0h	ECC Address Range 2 enable. • 0 = Disable ECC Address Range 2 • 1 = Enable ECC Address Range 2

Table 7-205. EMIF_ECCCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	ECC_ADDR_RNG_1_EN	RW	0h	ECC Address Range 1 enable. • 0 = Disable ECC Address Range 1 • 1 = Enable ECC Address Range 1

Table 7-206. Register Call Summary for EMIF_ECCCTL

EMIF Functional Description • Read-Modify-Write: [0] • Error Correction And Detection: [0][1][2][3][4]
EMIF Registers • EMIF_ECCCTL Register (Offset = 110h) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.26 EMIF_ECCADDR1 Register (Offset = 114h) [reset = 0h]

The EMIF_ECCADDR1 register is described in the following figure and table.

Table 7-207. EMIF_ECCADDR1 Instances

Instance	Physical Address
EMIF	2101 0114h

Figure 7-72. EMIF_ECCADDR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_END_ADDR_1																ECC_STRT_ADDR_1															
RW-0h																RW-0h															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-208. EMIF_ECCADDR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	ECC_END_ADDR_1	RW	0h	End address [32-17] of 33-bit address for ECC address range 1. If this bit field is set to 1000h, this indicates that the SDRAM physical end address on which the ECC applies is 0_2001_FFFFh. If this bit field is set to 0FFFh the physical end address on which the ECC applies is 0_1FFF_FFFFh. This bit field controls only the 16 MSbs of the physical end address of the ECC protected range. The other 17 LSbs are always 1_FFFFh.
15-0	ECC_STRT_ADDR_1	RW	0h	Start address [32-17] of 33-bit address for ECC address range 1. If this bit field is set to 0000h, this indicates that the SDRAM physical start address on which the ECC applies is 0_0000_0000h. This bit field controls only the 16 MSbs of the physical start address of the ECC protected range. The other 17 LSbs are always 0_0000h.

NOTE: The range is inclusive of start and end addresses.

Table 7-209. Register Call Summary for EMIF_ECCADDR1

EMIF Functional Description • Error Correction And Detection: [0]
EMIF Registers • EMIF_ECCADDR1 Register (Offset = 114h) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.27 EMIF_ECCADDR2 Register (Offset = 118h) [reset = 0h]

The EMIF_ECCADDR2 register is described in the following figure and table.

Table 7-210. EMIF_ECCADDR2 Instances

Instance	Physical Address
EMIF	2101 0118h

Figure 7-73. EMIF_ECCADDR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_END_ADDR_2																ECC_STRT_ADDR_2															
RW-0h																RW-0h															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-211. EMIF_ECCADDR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	ECC_END_ADDR_2	RW	0h	End address [32-17] of 33-bit address for ECC address range 2. If this bit field is set to 1000h, this indicates that the SDRAM physical end address on which the ECC applies is 0_2001_FFFFh. If this bit field is set to 0FFFh the physical end address on which the ECC applies is 0_1FFF_FFFFh. This bit field controls only the 16 MSbs of the physical end address of the ECC protected range. The other 17 LSbs are always 1_FFFFh.
15-0	ECC_STRT_ADDR_2	RW	0h	Start address [32-17] of 33-bit address for ECC address range 2. If this bit field is set to 0000h, this indicates that the SDRAM physical start address on which the ECC applies is 0_0000_0000h. This bit field controls only the 16 MSbs of the physical start address of the ECC protected range. The other 17 LSbs are always 0_0000h.

NOTE: The range is inclusive of start and end addresses.

Table 7-212. Register Call Summary for EMIF_ECCADDR2

EMIF Functional Description • Error Correction And Detection: [0]
EMIF Registers • EMIF_ECCADDR2 Register (Offset = 118h) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.28 EMIF_RWTHRESH Register (Offset = 120h) [reset = 305h]

The EMIF_RWTHRESH register is described in the following figure and table.

Table 7-213. EMIF_RWTHRESH Instances

Instance	Physical Address
EMIF	2101 0120h

Figure 7-74. EMIF_RWTHRESH Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			WR_THRSH				
R-0h			RW-3h				
7	6	5	4	3	2	1	0
RESERVED			RD_THRSH				
R-0h			RW-5h				

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-214. EMIF_RWTHRESH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Value = 0h Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
12-8	WR_THRSH	RW	3h	Write Threshold. Number of SDRAM write bursts after which the arbitration will switch to executing read commands. The value programmed is always minus 1 the required number.
7-5	RESERVED	R	0h	Value = 0h Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
4-0	RD_THRSH	RW	5h	Read Threshold. Number of SDRAM read bursts after which the arbitration will switch to executing write commands. The value programmed is always minus 1 the required number.

Table 7-215. Register Call Summary for EMIF_RWTHRESH

EMIF Functional Description • Reordering of Commands in the Command FIFO: [0][1]
EMIF Registers • EMIF_RWTHRESH Register (Offset = 120h) [reset = 305h]: [0] • EMIF Registers: [0]

7.2.5.29 EMIF_ONE_BIT_ECC_ERR_CNT Register (Offset = 130h) [reset = 0h]

The EMIF_ONE_BIT_ECC_ERR_CNT register is described in the following figure and table.

Table 7-216. EMIF_ONE_BIT_ECC_ERR_CNT Instances

Instance	Physical Address
EMIF	2101 0130h

Figure 7-75. EMIF_ONE_BIT_ECC_ERR_CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1B_ECC_ERR_CNT																															
WS-0h																															

LEGEND: WS = Write Value to Subtract Field; -*n* = value after reset

Table 7-217. EMIF_ONE_BIT_ECC_ERR_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	1B_ECC_ERR_CNT	WS	0h	32-bit counter that displays the number of 1-bit ECC errors. Writing a value will decrement the count by that value.

Table 7-218. Register Call Summary for EMIF_ONE_BIT_ECC_ERR_CNT

EMIF Functional Description • Counting the Number of 1-bit ECC Errors: [0][1][2]
EMIF Registers • EMIF_ONE_BIT_ECC_ERR_CNT Register (Offset = 130h) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.30 EMIF_ONE_BIT_ECC_ERR_THRSH Register (Offset = 134h) [reset = 0h]

The EMIF_ONE_BIT_ECC_ERR_THRSH register is described in the following figure and table.

Table 7-219. EMIF_ONE_BIT_ECC_ERR_THRSH Instances

Instance	Physical Address
EMIF	2101 0134h

Figure 7-76. EMIF_ONE_BIT_ECC_ERR_THRSH Register

31	30	29	28	27	26	25	24
1B_ECC_ERR_THRSH							
RW-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
1B_ECC_ERR_WIN							
RW-0h							
7	6	5	4	3	2	1	0
1B_ECC_ERR_WIN							
RW-0h							

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 7-220. EMIF_ONE_BIT_ECC_ERR_THRSH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	1B_ECC_ERR_THRSH	RW	0h	1-bit ECC error threshold. The EMIF controller generates an interrupt when the 1-bit ECC error count is greater than this threshold. A value of 0 disables the generation of interrupt.
23-16	RESERVED	R	0h	Reserved for future use
15-0	1B_ECC_ERR_WIN	RW	0h	1-bit ECC error window in number of refresh periods. The controller generates an interrupt when the 1-bit ECC error count is equal to or greater than the threshold within this window. A value of 0 disables the window. Refresh period is defined by the EMIF_SDRFC[15-0] REFRESH_RATE field.

Table 7-221. Register Call Summary for EMIF_ONE_BIT_ECC_ERR_THRSH

EMIF Functional Description • Counting the Number of 1-bit ECC Errors: [0]
EMIF Registers • EMIF_ONE_BIT_ECC_ERR_THRSH Register (Offset = 134h) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.31 EMIF_ONE_BIT_ECC_ERR_DIST_1 Register (Offset = 138h) [reset = 0h]

The EMIF_ONE_BIT_ECC_ERR_DIST_1 register is described in the following figure and table.

Table 7-222. EMIF_ONE_BIT_ECC_ERR_DIST_1 Instances

Instance	Physical Address
EMIF	2101 0138h

Figure 7-77. EMIF_ONE_BIT_ECC_ERR_DIST_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1B_ECC_ERR_DST_1																															
WOC-0h																															

LEGEND: WOC = Write 1 to Clear Bit; *-n* = value after reset

Table 7-223. EMIF_ONE_BIT_ECC_ERR_DIST_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	1B_ECC_ERR_DST_1	WOC	0h	1-bit ECC error distribution over data bus bits 31:0. A value of 1 on a bit indicates 1-bit error on the corresponding bit on the data bus. Writing a 1 to any bit clears that bit. Writing a 0 has no effect.

Table 7-224. Register Call Summary for EMIF_ONE_BIT_ECC_ERR_DIST_1

EMIF Functional Description • Diagnosing the Data Bus Bit on Which 1-bit ECC Errors Occur: [0][1]
EMIF Registers • EMIF_ONE_BIT_ECC_ERR_DIST_1 Register (Offset = 138h) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.32 EMIF_ONE_BIT_ECC_ERR_ADDR_LOG Register (Offset = 13Ch) [reset = 0h]

The EMIF_ONE_BIT_ECC_ERR_ADDR_LOG register is described in the following figure and table.

Table 7-225. EMIF_ONE_BIT_ECC_ERR_ADDR_LOG Instances

Instance	Physical Address
EMIF	2101 013Ch

Figure 7-78. EMIF_ONE_BIT_ECC_ERR_ADDR_LOG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1B_ECC_ERR_ADDR																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-226. EMIF_ONE_BIT_ECC_ERR_ADDR_LOG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	1B_ECC_ERR_ADDR	RW	0h	1-bit ECC error address. Most significant bits of the starting address of the first two SDRAM bursts that had the 1-bit ECC error. This field displays the first address logged in the 2 deep address logging FIFO. Writing a 1h will pop one element of the FIFO. Writing a 2h will pop both the elements of the FIFO. Writing any other value has no effect..

Table 7-227. Register Call Summary for EMIF_ONE_BIT_ECC_ERR_ADDR_LOG

EMIF Functional Description • Logging 1-bit ECC Error Address: [0][1]
EMIF Registers • EMIF_ONE_BIT_ECC_ERR_ADDR_LOG Register (Offset = 13Ch) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.33 EMIF_TWO_BIT_ECC_ERR_ADDR_LOG Register (Offset = 140h) [reset = 0h]

The EMIF_TWO_BIT_ECC_ERR_ADDR_LOG register is show in the figure and table below.

Table 7-228. EMIF_TWO_BIT_ECC_ERR_ADDR_LOG Instances

Instance	Physical Address
EMIF	2101 0140h

Figure 7-79. EMIF_TWO_BIT_ECC_ERR_ADDR_LOG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
2B_ECC_ERR_ADDR																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-229. EMIF_TWO_BIT_ECC_ERR_ADDR_LOG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	2B_ECC_ERR_ADDR	RW	0h	2-bit ECC error address. Most significant bits of the start address of the first SDRAM burst that had the 2-bit ECC error. Writing 1h clears this field. Writing any other value has no effect.

Table 7-230. Register Call Summary for EMIF_TWO_BIT_ECC_ERR_ADDR_LOG

EMIF Functional Description • Logging 2-bit ECC Error Address: [0]
EMIF Registers • EMIF_TWO_BIT_ECC_ERR_ADDR_LOG Register (Offset = 140h) [reset = 0h]: [0] • EMIF Registers: [0]

7.2.5.34 EMIF_ONE_BIT_ECC_ERR_DIST_2 Register (Offset = 144h) [reset = 0h]

The EMIF_ONE_BIT_ECC_ERR_DIST_2 register is described in the figure and table below.

Table 7-231. EMIF_ONE_BIT_ECC_ERR_DIST_2 Instances

Instance	Physical Address
EMIF	2101 0144h

Figure 7-80. EMIF_ONE_BIT_ECC_ERR_DIST_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1B_ECC_ERR_DST_2																															
WOC-0h																															

LEGEND: WOC = Write 1 to Clear Bit; *-n* = value after reset

Table 7-232. EMIF_ONE_BIT_ECC_ERR_DIST_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	1B_ECC_ERR_DST_2	WOC	0h	1-bit ECC error distribution over data bus bits 63:32. A value of 1 on a bit indicates 1-bit error on the corresponding bit on the data bus. Writing a 1 to any bit will clear that bit. Writing a 0 has no effect.

Table 7-233. Register Call Summary for EMIF_ONE_BIT_ECC_ERR_DIST_2

EMIF Registers
- EMIF_ONE_BIT_ECC_ERR_DIST_2 Register (Offset = 144h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.35 DDR_PHY_PIR Register (Offset = 4h) [reset = 0h]

The PHY Initialization register is used to configure and control initialization of the DDR PHY. This includes the triggering of certain initialization routines, as well as reset of the PHY and/or the PLLs used in the PHY. Any DDR_PHY_PIR register bit used to trigger an initialization routine (bits 0 to 15) is self clearing and is set to 0 once the initialization is done. Any configuration register write that sets the DDR_PHY_PIR[INIT] register bit triggers initialization as selected by the other DDR_PHY_PIR register bits. The completion status of this initialization can be checked by polling the PHY General Status Register 0.

Table 7-234. DDR_PHY_PIR Instances

Instance	Physical Address
DDR_PHY	0232 9004h

Figure 7-81. DDR_PHY_PIR Register

31	30	29	28	27	26	25	24
INITBYP	ZCALBYP	DCALBYP	LOCKBYP	CLRSR	RESERVED		
RW-0h	RW-0h	RW-0h	RW-0h	RW-0h	R-0h		
23	22	21	20	19	18	17	16
RESERVED					CTLDINIT	PLLBYP	ICPC
R-0h					RW-0h	RW-0h	RW-0h
15	14	13	12	11	10	9	8
WREYE	RDEYE	WRDSKW	RDDSKW	WLADJ	QSGATE	WL	DRAMINIT
RW-0h	RW-0h	RW-0h	RW-0h	RW-0h	RW-0h	RW-0h	RW-0h
7	6	5	4	3	2	1	0
DRAMRST	PHYRST	DCAL	PLLINIT	RESERVED		ZCAL	INIT
RW-0h	RW-0h	RW-0h	RW-0h	R-0h		RW-0h	RW-0h

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-235. DDR_PHY_PIR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	INITBYP	RW	0h	Initialization Bypass: Bypasses or stops, if set, all initialization routines currently running, including PHY initialization, DRAM initialization, and PHY training. Initialization may be triggered manually using INIT and the other relevant bits of the DDR_PHY_PIR register. This bit is self-clearing.
30	ZCALBYP	RW	0h	Impedance Calibration Bypass: Bypasses or stops, if set, impedance calibration of all ZQ control blocks that automatically triggers after reset. Impedance calibration may be triggered manually using INIT and ZCAL bits of the DDR_PHY_PIR register. This bit is self-clearing.
29	DCALBYP	RW	0h	Digital Delay Line (DDL) Calibration Bypass: Bypasses or stops, if set, DDL calibration that automatically triggers after reset. DDL calibration may be triggered manually using INIT and DCAL bits of the DDR_PHY_PIR register. This bit is self-clearing.
28	LOCKBYP	RW	0h	PLL Lock Bypass: Bypasses or stops, if set, the waiting of PLLs to lock. PLL lock wait is automatically triggered after reset. PLL lock wait may be triggered manually using INIT and PLLLOCK bits of the DDR_PHY_PIR register. This bit is self-clearing.

Table 7-235. DDR_PHY_PIR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	CLRSR	RW	0h	Clear Status Registers: A write of '1' to this bit will clear (reset to '0') all status registers, including PGSR and DXnGSR. The clear status register bit is self-clearing. Note, this bit does not clear the PGSR.IDONE bit. If the IDONE bit is set it remains at 1'b1 to indicate the PUB has competed its task. This bit is primarily for debug purposes and is typically not needed during normal functional operation. It can be used when PGSR.IDONE=1, to manually clear the PGSR status bits, although starting a new init process will automatically clear the PGSR status bits. Or it can be used to manually clear the DXnGSR status bits, although starting a new data training process will automatically clear the DXnGSR status bits.
26-19	RESERVED	R	0h	Returns zeros on reads.
18	CTLDINIT	RW	0h	Controller DRAM Initialization: Indicates if set that DRAM initialization will be performed by the controller. Otherwise if not set it indicates that DRAM initialization will be performed using the built-in initialization sequence or using software through the configuration port.
17	PLLBYP	RW	0h	PLL Bypass: A setting of 1 on this bit will put all PHY PLLs in bypass mode.
16	ICPC	RW	0h	Initialization Complete Pin Configuration: Specifies how the DFI initialization complete output pin (dfi_init_complete) should be used to indicate the status of initialization. Valid value are: • 0 = Asserted after PHY initialization (DLL locking and impedance calibration) is complete. • 1 = Asserted after PHY initialization is complete and the triggered the PUB initialization (DRAM initialization, data training, or initialization trigger with no selected initialization) is complete.
15	WREYE	RW	0h	Write Data Eye Training: Executes a PUB training routine to maximize the write data eye.
14	RDEYE	RW	0h	Read Data Eye Training: Executes a PUB training routine to maximize the read data eye.
13	WRDSKW	RW	0h	Write Data Bit Deskew: Executes a PUB training routine to deskew the DQ bits during write
12	RDDSKW	RW	0h	Read Data Bit Deskew: Executes a PUB training routine to deskew the DQ bits during read
11	WLADJ	RW	0h	Write Leveling Adjust (DDR3 Only): Executes a PUB training routine that readjusts the write latency used during write in case the write leveling routine changed the expected latency.
10	QSGATE	RW	0h	Read DQS Gate Training: Executes a PUB training routine to determine the optimum position of the read data DQS strobe for maximum system timing margins
9	WL	RW	0h	Write Leveling (DDR3 Only): Executes a PUB write leveling routine.
8	DRAMINIT	RW	0h	DRAM Initialization: Executes the DRAM initialization sequence
7	DRAMRST	RW	0h	DRAM Reset: Issues a reset to the DRAM (by driving the DRAM reset pin low) and wait 200us. This can be triggered in isolation or with the full DRAM initialization (DRAMINIT). For the later case, the reset is issued and 200us is waited before starting the full initialization sequence.
6	PHYRST	RW	0h	PHY Reset: Resets the AC and DATX8 modules by asserting the AC/DATX8 reset pin.
5	DCAL	RW	0h	Digital Delay Line (DDL) Calibration: Performs PHY delay line calibration.
4	PLLINIT	RW	0h	PLL Initialization: Executes the PLL initialization sequence which includes correct driving of PLL power-down, reset and gear shift pins, and then waiting for the PHY PLLs to lock.

Table 7-235. DDR_PHY_PIR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	RESERVED	R	0h	Returns zeros on reads
1	ZCAL	RW	0h	Impedance Calibration: Performs PHY impedance calibration. When set the impedance calibration will be performed in parallel with PHY initialization (PLL initialization + DDL calibration + PHY reset).
0	INIT	RW	0h	Initialization Trigger: A write of '1' to this bit triggers the DDR system initialization, including PHY initialization, DRAM initialization, and PHY training. The exact initialization steps to be executed are specified in bits 1 to 15 of this register. A bit setting of 1 means the step will be executed as part of the initialization sequence, while a setting of '0' means the step will be bypassed. The initialization trigger bit is self-clearing.

Table 7-236. Register Call Summary for DDR_PHY_PIR

EMIF Functional Description
- Leveling: [0]
- Error Correction And Detection: [0]

EMIF Registers
- DDR_PHY_ZQ0CR0 Register (Offset = 180h) [reset = 4000014Ah]: [0]
- DDR_PHY_ZQ1CR0 Register (Offset = 190h) [reset = 4000014Ah]: [0]
- DDR_PHY_ZQ2CR0 Register (Offset = 1A0h) [reset = 4000014Ah]: [0]
- DDR_PHY_PIR Register (Offset = 4h) [reset = 0h]: [0][1][2][3][4][5][6]
- EMIF Registers: [0]
- DDR_PHY_PGSR0 Register (Offset = 10h) [reset = 0h]: [0]
- DDR_PHY_PGCR1 Register (Offset = Ch) [reset = 300C461h]: [0]

7.2.5.36 DDR_PHY_PGCR0 Register (Offset = 8h) [reset = A8003E3Fh]

DDR_PHY_PGCR0-2 are used for miscellaneous PHY configurations such as enabling VT drift compensation and setting up write-leveling.

Table 7-237. DDR_PHY_PGCR0 Instances

Instance	Physical Address
DDR_PHY	0232 9008h

Figure 7-82. DDR_PHY_PGCR0 Register

31	30	29	28	27	26	25	24
CKEN						RESERVED	
RW-2Ah						R-0h	
23	22	21	20	19	18	17	16
RESERVED						DTOSEL	
R-0h						RW-0h	
15	14	13	12	11	10	9	8
DTOSEL		OSCWDL		OSCDIV			OSCEN
RW-0h		RW-3h		RW-7h			RW-0h
7	6	5	4	3	2	1	0
DLTST	DLTMODE	RDBVT	WDBVT	RGLVT	RDLVT	WDLVT	WLLVT
RW-0h	RW-0h	RW-1h	RW-1h	RW-1h	RW-1h	RW-1h	RW-1h

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-238. DDR_PHY_PGCR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	CKEN	RW	2Ah	CK Enable: Controls whether the CK going to the SDRAM is enabled (toggling) or disabled (static value) and whether the CK is inverted. Two bits for each of the up to three CK pairs. Valid values for the two bits are: • 00 = CK disabled (Driven to constant 0) • 01 = CK toggling with inverted polarity • 10 = CK toggling with normal polarity (This should be the default setting) • 11 = CK disabled (Driven to constant 1) TI recommends that this field be always programmed to 101010b.
25-19	RESERVED	R	0h	Returns zeros on reads.

Table 7-238. DDR_PHY_PGCR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-14	DTOSEL	RW	0h	Digital Test Output Select: Selects the PHY digital test output that should be driven onto PHY digital test output (phy_dto) pin: Valid values are: • 00000 = DATX8 0 PLL digital test output • 00001 = DATX8 1 PLL digital test output • 00010 = DATX8 2 PLL digital test output • 00011 = DATX8 3 PLL digital test output • 00100 = DATX8 4 PLL digital test output • 00101 = DATX8 5 PLL digital test output • 00110 = DATX8 6 PLL digital test output • 00111 = DATX8 7 PLL digital test output • 01000 = DATX8 8 PLL digital test output • 01001 = AC PLL digital test output • 01010 – 01111 = Reserved • 10000 = DATX8 0 delay line digital test output • 10001 = DATX8 1 delay line digital test output • 10010 = DATX8 2 delay line digital test output • 10011 = DATX8 3 delay line digital test output • 10100 = DATX8 4 delay line digital test output • 10101 = DATX8 5 delay line digital test output • 10110 = DATX8 6 delay line digital test output • 10111 = DATX8 7 delay line digital test output • 11000 = DATX8 8 delay line digital test output • 11001 = AC delay line digital test output • 11010 – 11111 = Reserved
13-12	OSCWDL	RW	3h	Oscillator Mode Write-Leveling Delay Line Select: Selects which of the two write leveling LCDLs is active. The delay select value of the inactive LCDL is set to zero while the delay select value of the active LCDL can be varied by the input write leveling delay select pin. Valid values are: • 00 = No WL LCDL is active • 01 = DDR WL LCDL is active • 10 = SDR WL LCDL is active • 11 = Both LCDLs are active
11-9	OSCDIV	RW	7h	Oscillator Mode Division: Specifies the factor by which the delay line oscillator mode output is divided down before it is output on the delay line digital test output pin dl_dto. Valid values are: • 000 = Divide by 1 • 001 = Divide by 256 • 010 = Divide by 512 • 011 = Divide by 1024 • 100 = Divide by 2048 • 101 = Divide by 4096 • 110 = Divide by 8192 • 111 = Divide by 65536
8	OSCEN	RW	0h	Oscillator Enable: Enables, if set, the delay line oscillation.
7	DLTST	RW	0h	Delay Line Test Start: A write of '1' to this bit will trigger delay line oscillator mode period measurement. This bit is not self clearing and needs to be reset to '0' before the measurement can be re-triggered.
6	DLTMODE	RW	0h	Delay Line Test Mode: Selects, if set, the delay line oscillator test mode.
5	RDBVT	RW	1h	Read Data BDL VT Compensation: Enables, if set the VT drift compensation of the read data bit delay lines
4	WDBVT	RW	1h	Write Data BDL VT Compensation: Enables, if set the VT drift compensation of the write data bit delay lines.

Table 7-238. DDR_PHY_PGCR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	RGLVT	RW	1h	Read DQS Gating LCDL Delay VT Compensation: Enables, if set the VT drift compensation of the read DQS gating LCDL
2	RDLVT	RW	1h	Read DQS LCDL Delay VT Compensation: Enables, if set the VT drift compensation of the read DQS LCDL.
1	WDLVT	RW	1h	Write DQ LCDL Delay VT Compensation: Enables, if set the VT drift compensation of the write DQ LCDL.
0	WLLVT	RW	1h	Write Leveling LCDL Delay VT Compensation: Enables, if set, the VT drift compensation of the write leveling LCDL

Table 7-239. Register Call Summary for DDR_PHY_PGCR0

EMIF Registers

- DDR_PHY_PGCR2 Register (Offset = 8Ch) [reset = F12480h]: [0]
- DDR_PHY_PGCR0 Register (Offset = 8h) [reset = A8003E3Fh]: [0]
- EMIF Registers: [0]
- DDR_PHY_PGCR1 Register (Offset = Ch) [reset = 300C461h]: [0]

7.2.5.37 DDR_PHY_PGCR1 Register (Offset = Ch) [reset = 300C461h]

DDR_PHY_PGCR0-2 are used for miscellaneous PHY configurations such as enabling VT drift compensation and setting up write-leveling.

Table 7-240. DDR_PHY_PGCR1 Instances

Instance	Physical Address
DDR_PHY	0232 900Ch

Figure 7-83. DDR_PHY_PGCR1 Register

31	30	29	28	27	26	25	24
LBMODE	LBGDQS		LBDQSS	IOLB	INHVT	PHYHRST	ZCKSEL
RW-0h	RW-0h		RW-0h	RW-0h	RW-0h	RW-1h	RW-2h
23	22	21	20	19	18	17	16
ZCKSEL	DLDLMT						
RW-2h	RW-1h						
15	14	13	12	11	10	9	8
DLDLMT	FDEPTH		LPFDEPTH		LPFEN	MDLEN	IODDRM
RW-1h	RW-2h		RW-0h		RW-1h	RW-0h	RW-0h
7	6	5	4	3	2	1	0
IODDRM	WLSELT	RESERVED			WLSTEP	WLMODE	PDDISDX
RW-0h	RW-1h	R-4h			RW-0h	RW-0h	RW-1h

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-241. DDR_PHY_PGCR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	LBMODE	RW	0h	Loopback Mode: Indicates, if set, that the PHY/PUB is in loopback mode
30-29	LBGDQS	RW	0h	Loopback DQS Gating: Selects the DQS gating mode that should be used when the PHY is in loopback mode, including BIST loopback mode. Valid values are: • 00 = DQS gate is always on • 01 = DQS gate training will be triggered on the PUB • 10 = DQS gate is set manually using software • 11 = Reserved
28	LBDQSS	RW	0h	Loopback DQS Shift: Selects how the read DQS is shifted during loopback to ensure that the read DQS is centered into the read data eye. Valid values are: • 0 = PUB sets the read DQS LCDL to 0; DQS is already shifted 90 degrees by write path • 1 = The read DQS shift is set manually through software
27	IOLB	RW	0h	I/O Loop-Back Select: Selects where inside the I/O the loop-back of signals happens. Valid values are: • 0 = Loopback is after output buffer; output enable must be asserted • 1 = Loopback is before output buffer; output enable is don't care
26	INHVT	RW	0h	VT Calculation Inhibit: Inhibits calculation of the next VT compensated delay line values. A value of 1 will inhibit the VT calculation. This bit should be set to 1 during writes to the delay line registers. This bit should NOT be set to 1 until after PHY initialization completes

Table 7-241. DDR_PHY_PGCR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	PHYHRST	RW	1h	PHY High-Speed Reset: A write of '0' to this bit resets the AC and DATX8 macros without resetting PUB RTL logic. This bit is not self-clearing and a '1' must be written to de-assert the reset
24-23	ZCKSEL	RW	2h	Impedance Clock Divider Select: Selects the divide ratio for the clock used by the impedance control logic relative to the clock used by the memory controller and SDRAM. Valid values are: • 00 = Divide by 2 • 01 = Divide by 8 • 10 = Divide by 32 • 11 = Divide by 64 TI recommends that this field be always programmed to 01b.
22-15	DLDLMT	RW	1h	Delay Line VT Drift Limit: Specifies the minimum change in the delay line VT drift in one direction which should result in the assertion of the delay line VT drift status signal (vt_drift). The limit is specified in terms of delay select values. A value of 0 disables the assertion of delay line VT drift status signal.
14-13	FDEPTH	RW	2h	Filter Depth: Specifies the number of measurements over which all AC and DATX8 initial period measurements, that happen after reset or when calibration is manually triggered, are averaged. Valid values are: • 00 = 2 • 01 = 4 • 10 = 8 • 11 = 16
12-11	LPFDEPTH	RW	0h	Low-Pass Filter Depth: Specifies the number of measurements over which MDL period measurements are filtered. This determines the time constant of the low pass filter. Valid values are: • 00 = 2 • 01 = 4 • 10 = 8 • 11 = 16
10	LPFEN	RW	1h	Low-Pass Filter Enable: Enables, if set, the low pass filtering of MDL period measurements.
9	MDLEN	RW	0h	Master Delay Line Enable: Enables, if set, the AC master delay line calibration to perform subsequent period measurements following the initial period measurements that are performed after reset or on when calibration is manually triggered. These additional measurements are accumulated and filtered as long as this bit remains high.
8-7	IODDRM	RW	0h	This field should be programmed to 1h if using DDR3 and 2h if using DDR3L. • 1 = DDR3 • 2 = DDR3L
6	WLSELT	RW	1h	Write Leveling Select Type: Selects the encoding type for the write leveling select signal depending on the desired setup/hold margins for the internal pipelines. Valid values are: • 0 = Setup margin of 90 degrees and hold margin of 90 degrees • 1 = Setup margin of 135 degrees and hold margin of 45 degrees
5-3	RESERVED	R	4h	Return zeros on reads.
2	WLSTEP	RW	0h	Write Leveling Step: Specifies the number of delay step-size increments during each step of write leveling. Valid values are: • 0 = 32 step sizes • 1 = 1 step size

Table 7-241. DDR_PHY_PGCR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	WLMODE	RW	0h	Write Leveling (Software) Mode: Indicates if set that the PHY is in software write leveling mode in which software executes single steps of DQS pulsing by writing '1' to DDR_PHY_PIR.WL. The write leveling DQ status from the DRAM is captured in DXnGSR0.WLDQ
0	PDDISDX	RW	1h	Power Down Disabled Byte: Indicates if set that the PLL and I/Os of a disabled byte should be powered down

Table 7-242. Register Call Summary for DDR_PHY_PGCR1

EMIF Registers
- EMIF Registers: [0]

7.2.5.38 DDR_PHY_PGCR2 Register (Offset = 8Ch) [reset = F12480h]

DDR_PHY_PGCR0-2 are used for miscellaneous PHY configurations such as enabling VT drift compensation and setting up write-leveling.

Table 7-243. DDR_PHY_PGCR2 Instances

Instance	Physical Address
DDR_PHY	0232 908Ch

Figure 7-84. DDR_PHY_PGCR2 Register

31	30	29	28	27	26	25	24
DYNACPDD	RESERVED			DTPMXTMR			
RW-0h	R-0h			RW-Fh			
23	22	21	20	19	18	17	16
DTPMXTMR				FXDLAT	NOBUB	TREFPRD	
RW-Fh				RW-0h	RW-0h	RW-12480h	
15	14	13	12	11	10	9	8
TREFPRD							
RW-12480h							
7	6	5	4	3	2	1	0
TREFPRD							
RW-12480h							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-244. DDR_PHY_PGCR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DYNACPDD	RW	0h	Dynamic AC Power Down Driver: Powers down, when set, the output driver on I/O for the address and bank address lines
30-28	RESERVED	R	0h	These bits should be always programmed to 0.
27-20	DTPMXTMR	RW	Fh	Data Training PUB Mode Timer Exit: Specifies the number of controller clocks to wait when entering and exiting pub mode data training. The default value ensures controller refreshes do not cause memory model errors when entering and exiting data training. The value should be increased if controller initiated SDRAM ZQ short or long operation may occur just before or just after the execution of data training.
19	FXDLAT	RW	0h	Fixed Latency: Specified whether all reads should be returned to the controller with a fixed read latency. Enabling fixed read latency increases the read latency. Valid values are: • 0 = Disable fixed read latency • 1 = Enable fixed read latency Fixed read latency is calculated as (12 + (maximum DXnGTR.RxDGSL)/2) half data rate clock cycles.
18	NOBUB	RW	0h	No Bubbles: Specified whether reads should be returned to the controller with no bubbles. Enabling no-bubble reads increases the read latency. Valid values are: • 0 = Bubbles are allowed during reads • 1 = Bubbles are not allowed during reads

Table 7-244. DDR_PHY_PGCR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17-0	TREFPRD	RW	12480h	Refresh Period: Indicates the period in clock cycles after which the PUB has to issue a refresh command to the SDRAM. This is derived from the maximum refresh interval from the data sheet, tRFC(max) or REFI, divided by the clock cycle time. A further 400 clocks must be subtracted from the derived number to account for command flow and missed slots of refreshes in the internal PUB blocks. The default corresponds to DDR3 9*7.8us at 1066MHz (DDR3-2133) when a burst of 9 refreshes are issued at every refresh interval. TI recommends tREFPRD be programmed to 5*tREFI.

Table 7-245. Register Call Summary for DDR_PHY_PGCR2

EMIF Registers
- EMIF Registers: [0]

7.2.5.39 DDR_PHY_PGSR0 Register (Offset = 10h) [reset = 0h]

DDR_PHY_PGSR0-1 are general status registers for the PHY. They indicate, among other things, whether initialization, write leveling or period measurement calibrations are done.

Table 7-246. DDR_PHY_PGSR0 Instances

Instance	Physical Address
DDR_PHY	0232 9010h

Figure 7-85. DDR_PHY_PGSR0 Register

31	30	29	28	27	26	25	24
APLOCK	RESERVED			WEERR	REERR	WDERR	RDERR
R-0h	R-0h			R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
WLAERR	QSGERR	WLERR	ZCERR	RESERVED			
R-0h	R-0h	R-0h	R-0h	R-0h			
15	14	13	12	11	10	9	8
RESERVED				WEDONE	REDONE	WDDONE	RDDONE
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
WLADONE	QSGDONE	WLDONE	DIDONE	ZCDONE	DCDONE	PLDONE	IDONE
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 7-247. DDR_PHY_PGSR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	APLOCK	R	0h	AC PLL Lock: Indicates, if set, that AC PLL has locked. This is a direct status of the AC PLL lock pin
30-28	RESERVED	R	0h	Returns zeros on reads.
27	WEERR	R	0h	Write Eye Training Error: Indicates if set that there is an error in write eye training.
26	REERR	R	0h	Read Eye Training Error: Indicates if set that there is an error in read eye training
25	WDERR	R	0h	Write Bit Deskew Error: Indicates if set that there is an error in write bit deskew
24	RDERR	R	0h	Read Bit Deskew Error: Indicates if set that there is an error in read bit deskew
23	WLAERR	R	0h	Write Leveling Adjustment Error: Indicates if set that there is an error in write leveling adjustment
22	QSGERR	R	0h	DQS Gate Training Error: Indicates if set that there is an error in DQS gate training.
21	WLERR	R	0h	Write Leveling Error: Indicates if set that there is an error in write leveling.
20	ZCERR	R	0h	Impedance Calibration Error: Indicates if set that there is an error in impedance calibration.
19-12	RESERVED	R	0h	Returns zeros on reads.
11	WEDONE	R	0h	Write Eye Training Done: Indicates if set that write eye training has completed
10	REDONE	R	0h	Read Eye Training Done: Indicates if set that read eye training has completed
9	WDDONE	R	0h	Write Bit Deskew Done: Indicates if set that write bit deskew has completed.

Table 7-247. DDR_PHY_PGSR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	RDDONE	R	0h	Read Bit Deskew Done: Indicates if set that read bit deskew has completed
7	WLADONE	R	0h	Write Leveling Adjustment Done: Indicates if set that write leveling adjustment has completed
6	QSGDONE	R	0h	DQS Gate Training Done: Indicates if set that DQS gate training has completed.
5	WLDONE	R	0h	Write Leveling Done: Indicates if set that write leveling has completed.
4	DIDONE	R	0h	DRAM Initialization Done: Indicates if set that DRAM initialization has completed
3	ZCDONE	R	0h	Impedance Calibration Done: Indicates if set that impedance calibration has completed
2	DCDONE	R	0h	Digital Delay Line (DDL) Calibration Done: Indicates if set that DDL calibration has completed
1	PLDONE	R	0h	PLL Lock Done: Indicates if set that PLL locking has completed.
0	IDONE	R	0h	Initialization Done: Indicates if set that the DDR system initialization has completed. This bit is set after all the selected initialization routines in DDR_PHY_PIR register have completed.

Table 7-248. Register Call Summary for DDR_PHY_PGSR0

EMIF Registers

- DDR_PHY_DX3GSR2 Register (Offset = 2B4h) [reset = 0h]: [0]
- DDR_PHY_DX3GSR0 Register (Offset = 284h) [reset = 0h]: [0]
- DDR_PHY_DX1GSR0 Register (Offset = 204h) [reset = 0h]: [0]
- DDR_PHY_PGSR0 Register (Offset = 10h) [reset = 0h]: [0]
- DDR_PHY_DX1GSR2 Register (Offset = 234h) [reset = 0h]: [0]
- DDR_PHY_DX0GSR0 Register (Offset = 1C4h) [reset = 0h]: [0]
- DDR_PHY_DX0GSR2 Register (Offset = 1F4h) [reset = 0h]: [0]
- DDR_PHY_DX2GSR2 Register (Offset = 274h) [reset = 0h]: [0]
- DDR_PHY_DX2GSR0 Register (Offset = 244h) [reset = 0h]: [0]
- DDR_PHY_DX8GSR0 Register (Offset = 3C4h) [reset = 0h]: [0]
- DDR_PHY_DX8GSR2 Register (Offset = 3F4h) [reset = 0h]: [0]
- DDR_PHY_PGSR1 Register (Offset = 14h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.40 DDR_PHY_PGSR1 Register (Offset = 14h) [reset = 0h]

DDR_PHY_PGSR0-1 are general status registers for the PHY. They indicate, among other things, whether initialization, write leveling or period measurement calibrations are done.

Table 7-249. DDR_PHY_PGSR1 Instances

Instance	Physical Address
DDR_PHY	0232 9014h

Figure 7-86. DDR_PHY_PGSR1 Register

31	30	29	28	27	26	25	24
RESERVED	VTSTOP	RESERVED					DLTCODE
R-0h	R-0h	R-0h					R-0h
23	22	21	20	19	18	17	16
DLTCODE							
R-0h							
15	14	13	12	11	10	9	8
DLTCODE							
R-0h							
7	6	5	4	3	2	1	0
DLTCODE							DLTDONE
R-0h							R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 7-250. DDR_PHY_PGSR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Returns zeros on reads.
30	VTSTOP	R	0h	VT Stop: Indicates if set that the VT calculation logic has stopped computing the next values for the VT compensated delay line values. After assertion of the PGCR.INHVT, the VTSTOP bit should be read to ensure all VT compensation logic has stopped computations before writing to the delay line registers
29-25	RESERVED	R	0h	Returns zeros on reads.
24-1	DLTCODE	R	0h	Delay Line Test Code: Returns the code measured by the PHY control block that corresponds to the period of the AC delay line digital test output.
0	DLTDONE	R	0h	Delay Line Test Done: Indicates, if set, that the PHY control block has finished doing period measurement of the AC delay line digital test output

Table 7-251. Register Call Summary for DDR_PHY_PGSR1

EMIF Registers
- EMIF Registers: [0]

7.2.5.41 DDR_PHY_PLLCR Register (Offset = 18h) [reset = 1C000h]

The DDR_PHY_PLLCR register provides miscellaneous controls of the PLLs used in the AC and DATX8 macros, including PLL test modes and PLL bypass.

Table 7-252. DDR_PHY_PLLCR Instances

Instance	Physical Address
DDR_PHY	0232 9018h

Figure 7-87. DDR_PHY_PLLCR Register

31	30	29	28	27	26	25	24
BYP	PLLRST	PLLPD	RESERVED				
RW-0h	RW-0h	RW-0h	R-0h				
23	22	21	20	19	18	17	16
RESERVED				FRQSEL		QPMODE	CPPC
R-0h				RW-0h		RW-0h	RW-Eh
15	14	13	12	11	10	9	8
CPPC			CPIC		GSHIFT	RESERVED	
RW-Eh			RW-0h		R-0h	R-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-253. DDR_PHY_PLLCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	BYP	RW	0h	PLL Bypass: Bypasses the PLL if set to 1.
30	PLLRST	RW	0h	PLL Rest: Resets the PLLs by driving the PLL reset pin. This bit is not self-clearing and a '0' must be written to de-assert the reset.
29	PLLPD	RW	0h	PLL Power Down: Puts the PLLs in power down mode by driving the PLL power down pin. This bit is not self-clearing and a '0' must be written to de-assert the power-down
28-20	RESERVED	R	0h	Returns zeros on reads.
19-18	FRQSEL	RW	0h	PHY PLL reference frequency select: Selects the operating range of the PHY PLL. Valid values are: • 00 = PHY PLL reference clock ranges from 335 MHz to 533 MHz • 01 = PHY PLL reference clock ranges from 225 MHz to 385 MHz • 10 = Reserved • 11 = PHY PLL reference clock ranges from 166 MHz to 275 MHz
17	QPMODE	RW	0h	PLL Quadrature Phase Mode: Enables, if set, the quadrature phase clock outputs. This mode is not used in this version of the PHY
16-13	CPPC	RW	Eh	Charge Pump Proportional Current Control
12-11	CPIC	RW	0h	Charge Pump Integrating Current Control
10	GSHIFT	R	0h	Gear Shift: Enables, if set, rapid locking mode.
9-0	RESERVED	R	0h	Reads return zeros

Table 7-254. Register Call Summary for DDR_PHY_PLLCR

EMIF Registers

- DDR_PHY_DX2GCR Register (Offset = 240h) [reset = 7C000E81h]: [0][1][2]
- DDR_PHY_DX0GCR Register (Offset = 1C0h) [reset = 7C000E81h]: [0][1][2]
- DDR_PHY_DX3GCR Register (Offset = 280h) [reset = 7C000E81h]: [0][1][2]
- DDR_PHY_DX1GCR Register (Offset = 200h) [reset = 7C000E81h]: [0][1][2]
- DDR_PHY_PLLCR Register (Offset = 18h) [reset = 1C000h]: [0]
- EMIF Registers: [0]
- DDR_PHY_DX8GCR Register (Offset = 3C0h) [reset = 7C000E81h]: [0][1][2]

7.2.5.42 DDR_PHY_PTR0 Register (Offset = 1Ch) [reset = 42C21582h]

PHY Timing Registers are used to program different timing parameters used by the PHY logic. All the default values shown in these tables are in decimal format and correspond to the DDR3-2133N speed grade. This corresponds to DRAM bit rate of 2133Mbps, DRAM clock frequency of 1066MHz, controller clock frequency of 533MHz, and configuration clock frequency of 533MHz. Configure these parameters assuming controller and configuration clocks of 533MHz. The clocks in which the timing numbers are measured are specified in the description of each parameter.

Table 7-255. DDR_PHY_PTR0 Instances

Instance	Physical Address
DDR_PHY	0232 901Ch

Figure 7-88. DDR_PHY_PTR0 Register

31	30	29	28	27	26	25	24
TPLLPD							
RW-216h							
23	22	21	20	19	18	17	16
TPLLPD			TPLLGS				
RW-216h			RW-856h				
15	14	13	12	11	10	9	8
TPLLGS							
RW-856h							
7	6	5	4	3	2	1	0
TPLLGS		TPHYRST					
RW-856h		RW-2h					

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-256. DDR_PHY_PTR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-21	TPLLPD	RW	216h	PLL Power-Down Time: Number of VBUSP_CLK cycles that the PLL must remain in power-down mode, that is, number of clock cycles from when PLL power-down pin is asserted to when PLL power-down pin is de-asserted. This must correspond to a value that is equal to or more than 1us. Default value corresponds to 1us.
20-6	TPLLGS	RW	856h	PLL Gear Shift Time: Number of VBUSP_CLK cycles from when the PLL reset pin is de-asserted to when the PLL gear shift pin is de-asserted. This must correspond to a value that is equal to or more than 4us. Default value corresponds to 4us.
5-0	TPHYRST	RW	2h	PHY Reset Time: Number of VBUSP_CLK cycles that the PHY reset must remain asserted after PHY calibration is done before the reset to the PHY is de-asserted. This is used to extend the reset to the PHY so that the reset is asserted for some clock cycles after the clocks are stable. TI recommends setting this to 15.

Table 7-257. Register Call Summary for DDR_PHY_PTR0

EMIF Registers
- EMIF Registers: [0]

7.2.5.43 DDR_PHY_PTR1 Register (Offset = 20h) [reset = D05612C0h]

PHY Timing Registers are used to program different timing parameters used by the PHY logic. All the default values shown in these tables are in decimal format and correspond to the highest speed the PHY can be compiled for, that is, JEDEC DDR3-2133N speed grade. This corresponds to DRAM bit rate of 2133Mbps, DRAM clock frequency of 1066MHz, controller clock frequency of 533MHz, and configuration clock frequency of 533MHz. Configure these parameters assuming controller and configuration clocks of 533MHz. The clocks in which the timing numbers are measured are specified in the description of each parameter.

Table 7-258. DDR_PHY_PTR1 Instances

Instance	Physical Address
DDR_PHY	0232 9020h

Figure 7-89. DDR_PHY_PTR1 Register

31	30	29	28	27	26	25	24
TPLLLOCK							
RW-D056h							
23	22	21	20	19	18	17	16
TPLLLOCK							
RW-D056h							
15	14	13	12	11	10	9	8
RESERVED			TPLLRST				
R-0h			RW-12C0h				
7	6	5	4	3	2	1	0
TPLLRST							
RW-12C0h							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-259. DDR_PHY_PTR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	TPLLLOCK	RW	D056h	PLL Lock Time: Number of VBUSP_CLK cycles for the PLL to stabilize and lock, that is, number of clock cycles from when the PLL reset pin is de-asserted to when the PLL has lock and is ready for use. This must correspond to a value that is equal to or more than 100us. Default value corresponds to 100us
15-13	RESERVED	R	0h	Reads return zeros.
12-0	TPLLRST	RW	12C0h	PLL Reset Time: Number of VBUSP_CLK cycles that the PLL must remain in reset mode, that is, number of clock cycles from when PLL power-down pin is de-asserted and PLL reset pin is asserted to when PLL reset pin is de-asserted. This must correspond to a value that is equal to or more than 9 us. Default value corresponds to 9 us.

Table 7-260. Register Call Summary for DDR_PHY_PTR1

EMIF Registers
- EMIF Registers: [0]

7.2.5.44 DDR_PHY_PTR2 Register (Offset = 24h) [reset = 13DEFh]

PHY Timing Registers are used to program different timing parameters used by the PHY logic. All the default values shown in these tables are in decimal format and correspond to the highest speed the PHY can be compiled for, that is, JEDEC DDR3-2133N speed grade. This corresponds to DRAM bit rate of 2133Mbps, DRAM clock frequency of 1066MHz, controller clock frequency of 533MHz, and configuration clock frequency of 533MHz. Configure these parameters assuming controller and configuration clocks of 533MHz. The clocks in which the timing numbers are measured are specified in the description of each parameter.

Table 7-261. DDR_PHY_PTR2 Instances

Instance	Physical Address
DDR_PHY	0232 9024h

Figure 7-90. DDR_PHY_PTR2 Register

31	30	29	28	27	26	25	24
RESERVED							
RW-0h							
23	22	21	20	19	18	17	16
RESERVED				TWLDLYS			
RW-0h				RW-2h			
15	14	13	12	11	10	9	8
TWLDLYS	TCALH					TCALS	
RW-2h	RW-Fh					RW-Fh	
7	6	5	4	3	2	1	0
TCALS				TCALON			
RW-Fh				RW-Fh			

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-262. DDR_PHY_PTR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	RW	0h	Reads return zeros.
19-15	TWLDLYS	RW	2h	Write Leveling Delay Settling Time: Number of controller clock cycles from when a new value of the write leveling delay is applies to the LCDL to when to DQS high is driven high. This allows the delay to settle
14-10	TCALH	RW	Fh	Calibration Hold Time: Number of controller clock cycles from when the clock was disabled (cal_clk_en deasserted) to when calibration is enable (cal_en asserted).
9-5	TCALS	RW	Fh	Calibration Setup Time: Number of controller clock cycles from when calibration is enabled (cal_en asserted) to when the calibration clock is asserted again (cal_clk_en asserted).
4-0	TCALON	RW	Fh	Calibration On Time: Number of controller clock cycles that the calibration clock is enabled (cal_clk_en asserted).

Table 7-263. Register Call Summary for DDR_PHY_PTR2

EMIF Registers
- EMIF Registers: [0]

7.2.5.45 DDR_PHY_PTR3 Register (Offset = 28h) [reset = 1800D056h]

PHY Timing Registers are used to program different timing parameters used by the PHY logic. All the default values shown in these tables are in decimal format and correspond to the highest speed the PHY can be compiled for, that is, JEDEC DDR3-2133N speed grade. This corresponds to DRAM bit rate of 2133Mbps, DRAM clock frequency of 1066MHz, controller clock frequency of 533MHz, and configuration clock frequency of 533MHz. Configure these parameters assuming controller and configuration clocks of 533MHz. The clocks in which the timing numbers are measured are specified in the description of each parameter.

Table 7-264. DDR_PHY_PTR3 Instances

Instance	Physical Address
DDR_PHY	0232 9028h

Figure 7-91. DDR_PHY_PTR3 Register

31	30	29	28	27	26	25	24
RESERVED			TDINIT1				
R-0h			RW-180h				
23	22	21	20	19	18	17	16
TDINIT1				TDINIT0			
RW-180h				RW-D056h			
15	14	13	12	11	10	9	8
TDINIT0							
RW-D056h							
7	6	5	4	3	2	1	0
TDINIT0							
RW-D056h							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-265. DDR_PHY_PTR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reads return zeros.
28-20	TDINIT1	RW	180h	DRAM Initialization Time 1: DRAM initialization time in DRAM clock cycles corresponding to the following: DDR3 = CKE high time to first command (tRFC + 10 ns or 5 tCK, whichever is bigger) Default value corresponds to DDR3 tRFC of 360ns at 1066 MHz tDINIT1 = (tCKE/tCK)-1
19-0	TDINIT0	RW	D056h	DRAM Initialization Time 0: DRAM initialization time in DRAM clock cycles corresponding to the following: DDR3 = CKE low time with power and clock stable (500 us)

Table 7-266. Register Call Summary for DDR_PHY_PTR3

EMIF Registers
- EMIF Registers: [0]

7.2.5.46 DDR_PHY_PTR4 Register (Offset = 2Ch) [reset = AAF4156h]

PHY Timing Registers are used to program different timing parameters used by the PHY logic. All the default values shown in these tables are in decimal format and correspond to the highest speed the PHY can be compiled for, that is, JEDEC DDR3-2133N speed grade. This corresponds to DRAM bit rate of 2133Mbps, DRAM clock frequency of 1066MHz, controller clock frequency of 533MHz, and configuration clock frequency of 533MHz. Configure these parameters assuming controller and configuration clocks of 533MHz. The clocks in which the timing numbers are measured are specified in the description of each parameter.

Table 7-267. DDR_PHY_PTR4 Instances

Instance	Physical Address
DDR_PHY	0232 902Ch

Figure 7-92. DDR_PHY_PTR4 Register

31	30	29	28	27	26	25	24
RESERVED				TDINIT3			
R-0h				RW-2ABh			
23	22	21	20	19	18	17	16
TDINIT3						TDINIT2	
RW-2ABh						RW-34156h	
15	14	13	12	11	10	9	8
TDINIT2							
RW-34156h							
7	6	5	4	3	2	1	0
TDINIT2							
RW-34156h							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-268. DDR_PHY_PTR4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reads return zeros.
27-18	TDINIT3	RW	2ABh	DRAM Initialization Time 3: DRAM initialization time in DRAM clock cycles corresponding to the following: DDR3 = Time from ZQ initialization command to first command (1 us) Default value corresponds to the DDR3 640ns at 1066 MHz.
17-0	TDINIT2	RW	34156h	DRAM Initialization Time 2: DRAM initialization time in DRAM clock cycles corresponding to the following: DDR3 = Reset low time (200 us on power-up or 100 ns after power-up) Default value corresponds to DDR3 200 us at 1066 MHz.

Table 7-269. Register Call Summary for DDR_PHY_PTR4

EMIF Registers
- EMIF Registers: [0]

7.2.5.47 DDR_PHY_ACIOCR Register (Offset = 38h) [reset = 33C03812h]

The AC I/O Configuration Register is used to control the slew rate on the address/command lines.

Table 7-270. DDR_PHY_ACIOCR Instances

Instance	Physical Address
DDR_PHY	0232 9038h

Figure 7-93. DDR_PHY_ACIOCR Register

31	30	29	28	27	26	25	24
ACSR		RESERVED					
RW-0h		R-33C03812h					
23	22	21	20	19	18	17	16
RESERVED							
R-33C03812h							
15	14	13	12	11	10	9	8
RESERVED							
R-33C03812h							
7	6	5	4	3	2	1	0
RESERVED							
R-33C03812h							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-271. DDR_PHY_ACIOCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	ACSR	RW	0h	Address/Command Slew Rate: Selects slew rate of the I/O for all address and command pins. 00 = very fast (use as default for DDR3 and DDR3L) 01 = fast 10 = medium 11 = slow
29-0	RESERVED	R	33C03812h	Reserved.

Table 7-272. Register Call Summary for DDR_PHY_ACIOCR

EMIF Registers
- EMIF Registers: [0]

7.2.5.48 DDR_PHY_DXCCR Register (Offset = 3Ch) [reset = 44181884h]

The DATX8 Common Configuration Register is used to control features that affect all DATX8 macros. These include on-die termination enables and power-down enables for SSTL I/Os for all SDRAM data, data mask and data strobe signals.

Table 7-273. DDR_PHY_DXCCR Instances

Instance	Physical Address
DDR_PHY	0232 903Ch

Figure 7-94. DDR_PHY_DXCCR Register

31	30	29	28	27	26	25	24
DDPDRCDO				DDPDDCDO			
RW-4h				RW-4h			
23	22	21	20	19	18	17	16
DYNDXPDR	DYNDXPDD	UDQIOM	UDQPDR	UDQPDD	UDQODT	MSBUDQ	
RW-0h	RW-0h	RW-0h	RW-1h	RW-1h	RW-0h	RW-0h	
15	14	13	12	11	10	9	8
MSBUDQ	DXSR		DQSNRES				DQSRES
RW-0h	RW-0h		RW-Ch				RW-4h
7	6	5	4	3	2	1	0
DQSRES			DXPDR	DXPDD	MDLEN	DXIOM	DXODT
RW-4h			RW-0h	RW-0h	RW-1h	RW-0h	RW-0h

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-274. DDR_PHY_DXCCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	DDPDRCDO	RW	4h	Dynamic Data Power Down Receiver Count Down Offset: Offset applied in calculating window of time where receiver is powered up
27-24	DDPDDCDO	RW	4h	Dynamic Data Power Down Driver Count Down Offset: Offset applied in calculating window of time where driver is powered up
23	DYNDXPDR	RW	0h	Data Power Down Receiver: Dynamically powers down, when set, the input receiver on I/O for the DQ pins of the active DATX8 macros. Applies only when DXPDR and DXnGCR.DXPDR are not set to 1.
22	DYNDXPDD	RW	0h	Dynamic Data Power Down Driver: Dynamically powers down, when set, the output driver on I/O for the DQ pins of the active DATX8 macros. Applies only when DXPDD and DXnGCR.DXPDD are not set to 1.
21	UDQIOM	RW	0h	Unused DQ I/O Mode: Selects SSTL mode (when set to 0) or CMOS mode (when set to 1) of the I/O for unused DQ pins.
20	UDQPDR	RW	1h	Unused DQ Power Down Receiver: Powers down, when set, the input receiver on the I/O for unused DQ pins.
19	UDQPDD	RW	1h	Unused DQ Power Down Driver: Powers down, when set, the output driver on the I/O for unused DQ pins.
18	UDQODT	RW	0h	Unused DQ On-Die Termination: Enables, when set, the on-die termination on the I/O for unused DQ pins.
17-15	MSBUDQ	RW	0h	Most Significant Byte Unused DQs: Specifies the number of DQ bits that are not used in the most significant byte. The used (valid) bits for this byte are [8-MSBDQ- 1:0]. To disable the whole byte, use the DXnGCR.DXEN register.

Table 7-274. DDR_PHY_DXCCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14-13	DXSR	RW	0h	Data Slew Rate: Selects slew rate of the I/O for DQ, DM, and DQS/DQS# pins of all DATX8 macros. 00 = very fast (use as default for DDR3 and DDR3L) 01 = fast 10 = medium 11 = slow
12-9	DQSNRES	RW	Ch	DQS# Resistor: Selects the on-die pull-up/pull-down resistor for DQS# pins. Same encoding as DQSRES.
8-5	DQSRES	RW	4h	DQS Resistor: Selects the on-die pull-down/pull-up resistor for DQS pins. DQSRES[3] selects pull-down (when set to 0) or pull-up (when set to 1). DQSRES[2:0] selects the resistor value as follows: 000 = Open: On-die resistor disconnected (use external resistor) 001 = 688 ohms 010 = 611 ohms 011 = 550 ohms 100 = 500 ohms 101 = 458 ohms 110 = 393 ohms 111 = 344 ohms
4	DXPDR	RW	0h	Data Power Down Receiver: Powers down, when set, the input receiver on I/O for DQ, DM, and DQS/DQS# pins of all DATX8 macros. This bit is ORed with the PDR configuration bit of the individual DATX8.
3	DXPDD	RW	0h	Data Power Down Driver: Powers down, when set, the output driver on I/O for DQ,DM, and DQS/DQS# pins of all DATX8 macros. This bit is ORed with the PDD configuration bit of the individual DATX8.
2	MDLEN	RW	1h	Master Delay Line Enable: Enables, if set, all DATX8 master delay line calibration to perform subsequent period measurements following the initial period measurements that are performed after reset or on when calibration is manually triggered. These additional measurements are accumulated and filtered as long as this bit remains high. This bit is ANDed with the MDLEN bit in the individual DATX8
1	DXIOM	RW	0h	Data I/O Mode: Selects SSTL mode (when set to 0) or CMOS mode (when set to 1) of the I/O for DQ, DM, and DQS/DQS# pins of all DATX8 macros. This bit is ORed with the IOM configuration bit of the individual DATX8.
0	DXODT	RW	0h	Data On-Die Termination: Enables, when set, the on-die termination on the I/O for DQ, DM, and DQS/DQS# pins of all DATX8 macros. This bit is ORed with the ODT configuration bit of the individual DATX8

Table 7-275. Register Call Summary for DDR_PHY_DXCCR

EMIF Registers
- EMIF Registers: [0]

7.2.5.49 DDR_PHY_DCR Register (Offset = 44h) [reset = 40Bh]

The DDR_PHY_DCR register is used to configure the DRAM system, and is described in the figure and table below.

Table 7-276. DDR_PHY_DCR Instances

Instance	Physical Address
DDR_PHY	0232 9044h

Figure 7-95. DDR_PHY_DCR Register

31	30	29	28	27	26	25	24
RESERVED		UDIMM	RESERVED	NOSRA	RESERVED		
R-0h		RW-0h	R-0h	RW-0h	R-0h		
23	22	21	20	19	18	17	16
RESERVED						BYTEMASK	
R-0h						RW-1h	
15	14	13	12	11	10	9	8
BYTEMASK						RESERVED	
RW-1h						R-0h	
7	6	5	4	3	2	1	0
MPRDQ	PDQ			DDR8BNK	DDRMD		
RW-0h	RW-0h			RW-1h	RW-3h		

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-277. DDR_PHY_DCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reads return zeros.
29	UDIMM	RW	0h	Un-buffered DIMM Address Mirroring: Indicates if set that there is address mirroring on the second rank of an un-buffered DIMM (the rank connected to CS1). DDR_PHY_DCR[NOSRA] must be set if address mirroring is enabled.
28	RESERVED	R	0h	Reads return zeros.
27	NOSRA	RW	0h	No Simultaneous Rank Access: Specifies if set that simultaneous rank access on the same clock cycle is not allowed. This means that multiple chip select signals will not be asserted at the same time. This bit must be set if address mirroring is enabled in DDR_PHY_DCR[UDIMM]. NOSRA should be set to 0 if address mirroring is enabled (for example, DDR_PHY_DCR[UDIMM]=1).
26-18	RESERVED	R	0h	Reads return zeros.
17-10	BYTEMASK	RW	1h	Byte Mask: Mask applied to all beats of read data on all bytes lanes during read DQS gate training. This allows training to be conducted based on selected bit(s) from the byte lanes. Valid values for each bit are: • 0 = Disable compare for that bit • 1 = Enable compare for that bit Note that this mask in DDR3 MPR operation mode as well and must be in keeping with the PDQ field setting. TI recommends that this field be always programmed to 0000 0001b.
9-8	RESERVED	R	0h	Reads return zeros.

Table 7-277. DDR_PHY_DCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	MPRDQ	RW	0h	Multi-Purpose Register (MPR) DQ (DDR3 Only): Specifies the value that is driven on non-primary DQ pins during MPR reads. Valid values are: • 0 = Primary DQ drives out the data from MPR (0-1-0-1); non-primary DQs drive '0' • 1 = Primary DQ and non-primary DQs all drive the same data from MPR (0-1-0-1) TI recommends that this bit be always programmed to 0.
6-4	PDQ	RW	0h	Primary DQ (DDR3 Only): Specifies the DQ pin in a byte that is designated as a primary pin for Multi-Purpose Register (MPR) reads. Valid values are 0 to 7 for DQ[0] to DQ[7], respectively. TI recommends that this field be always programmed to 000b.
3	DDR8BNK	RW	1h	DDR 8-Bank: Indicates if set that the SDRAM used has 8 banks.
2-0	DDRMD	RW	3h	DDR Mode: Set to 3h for DDR3 mode.

Table 7-278. Register Call Summary for DDR_PHY_DCR

EMIF Registers
- EMIF Registers: [0]
- DDR_PHY_DCR Register (Offset = 44h) [reset = 40Bh]: [0][1][2][3]

7.2.5.50 DDR_PHY_DTPR0 Register (Offset = 48h) [reset = C9E4EE88h]

DDR_PHY_DTPR0-2 are used to program timing parameters for different DDR3 speed grades. These timing parameters are in DRAM clock cycles and are derived from corresponding parameters in the SDRAM data sheet divided by the DRAM clock cycle time. Non-integer values should be rounded up to the next integer. All the default values correspond to the DDR3-2133 speed grade.

Table 7-279. DDR_PHY_DTPR0 Instances

Instance	Physical Address
DDR_PHY	0232 9048h

Figure 7-96. DDR_PHY_DTPR0 Register

31	30	29	28	27	26	25	24
TRC						TRRD	
RW-32h						RW-7h	
23	22	21	20	19	18	17	16
TRRD			TRAS				
RW-7h			RW-24h				
15	14	13	12	11	10	9	8
TRCD				TRP			
RW-Eh				RW-Eh			
7	6	5	4	3	2	1	0
TWTR				TRTP			
RW-8h				RW-8h			

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-280. DDR_PHY_DTPR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	TRC	RW	32h	Activate to activate command delay (same bank). Valid values are 2 to 63
25-22	TRRD	RW	7h	Activate to activate command delay (different banks). Valid values are 1 to 15.
21-16	TRAS	RW	24h	Activate to precharge command delay. Valid values are 2 to 63
15-12	TRCD	RW	Eh	Activate to read or write delay. Minimum time from when an activate command is issued to when a read or write to the activated row can be issued. Valid values are 2 to 15.
11-8	TRP	RW	Eh	Precharge command period: The minimum time between a precharge command and any other command. Valid values are 2 to 15.
7-4	TWTR	RW	8h	Internal write to read command delay. Valid values are 1 to 15.
3-0	TRTP	RW	8h	Internal read to precharge command delay. Valid values are 2 to 15.

Table 7-281. Register Call Summary for DDR_PHY_DTPR0

EMIF Registers

- DDR_PHY_DTPR0 Register (Offset = 48h) [reset = C9E4EE88h]: [0]
- DDR_PHY_DTPR1 Register (Offset = 4Ch) [reset = 228BB4D2h]: [0]
- EMIF Registers: [0]
- DDR_PHY_DTPR2 Register (Offset = 50h) [reset = 10036A00h]: [0]

7.2.5.51 DDR_PHY_DTPR1 Register (Offset = 4Ch) [reset = 228BB4D2h]

DDR_PHY_DTPR0-2 are used to program timing parameters for different DDR3 speed grades. These timing parameters are in DRAM clock cycles and are derived from corresponding parameters in the SDRAM data sheet divided by the DRAM clock cycle time. Non-integer values should be rounded up to the next integer. All the default values correspond to the DDR3-2133 speed grade.

Table 7-282. DDR_PHY_DTPR1 Instances

Instance	Physical Address
DDR_PHY	0232 904Ch

Figure 7-97. DDR_PHY_DTPR1 Register

31	30	29	28	27	26	25	24
RESERVED		TWLO				TWLMRD	
R-0h		RW-8h				RW-28h	
23	22	21	20	19	18	17	16
TWLMRD				TRFC			
RW-28h				RW-176h			
15	14	13	12	11	10	9	8
TRFC					TFAW		
RW-176h					RW-26h		
7	6	5	4	3	2	1	0
TFAW			TMOD			TMRD	
RW-26h			RW-4h			RW-2h	

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-283. DDR_PHY_DTPR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	
29-26	TWLO	RW	8h	Write leveling output delay: Number of clock cycles from when write leveling DQS is driven high by the PHY to when the results from the SDRAM on DQ is sampled by the PHY. This must include the SDRAM tWLO timing parameter plus the round trip delay from the DUT to SDRAM back to the DUT. TI recommends that this field always be programmed to 12 decimal.
25-20	TWLMRD	RW	28h	Minimum delay from when write leveling mode is programmed to the first DQS/DQS# rising edge
19-11	TRFC	RW	176h	Refresh-to-Refresh: Indicates the minimum time, in clock cycles, between two refresh commands or between a refresh and an active command. This is derived from the minimum refresh interval from the data sheet, tRFC(min), divided by the clock cycle time.
10-5	TFAW	RW	26h	4-bank activate period. No more than 4-bank activate commands may be issued in a given tFAW period. Valid values are 2 to 63.
4-2	TMOD	RW	4h	Load mode update delay. The minimum time between a load mode register command and a non-load mode register command. Valid values are: • 000 = 12 • 001 = 13 • 010 = 14 • 011 = 15 • 100 = 16 • 101 = 17 • 110 – 111 = Reserved

Table 7-283. DDR_PHY_DTPR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	TMRD	RW	2h	Load mode cycle time: The minimum time between a load mode register command and any other command. For DDR3 this is the minimum time between two load mode register commands. the value used for tMRD is 4 plus the value programmed in these bits, that is, tMRD ranges from 4 to 7.

Table 7-284. Register Call Summary for DDR_PHY_DTPR1

EMIF Registers
- EMIF Registers: [0]

7.2.5.52 DDR_PHY_DTPR2 Register (Offset = 50h) [reset = 10036A00h]

DDR_PHY_DTPR0-2 are used to program timing parameters for different DDR3 speed grades. These timing parameters are in DRAM clock cycles and are derived from corresponding parameters in the SDRAM data sheet divided by the DRAM clock cycle time. Non-integer values should be rounded up to the next integer. All the default values correspond to the DDR3-2133 speed grade.

Table 7-285. DDR_PHY_DTPR2 Instances

Instance	Physical Address
DDR_PHY	0232 9050h

Figure 7-98. DDR_PHY_DTPR2 Register

31	30	29	28	27	26	25	24
TCCD	TRTW	TRTODT	TDLLK				
RW-0h	RW-0h	RW-0h	RW-200h				
23	22	21	20	19	18	17	16
TDLLK					TCKE		
RW-200h					RW-6h		
15	14	13	12	11	10	9	8
TCKE	TXP					TXS	
RW-6h	RW-1Ah					RW-200h	
7	6	5	4	3	2	1	0
TXS							
RW-200h							

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-286. DDR_PHY_DTPR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	TCCD	RW	0h	Read to read and write to write command delay. Valid values are: • 0 = 4 DRAM clock cycles • 1 = 5 DRAM clock cycles
30	TRTW	RW	0h	Read to Write command delay. This parameter is only used when the PHY issues commands during initialization and leveling. During normal operation, the controller issues commands to the SDRAM and uses the timing parameters programmed in the controller. TI recommends that this bit be always set to 1 to provide additional margin during PHY initiated initialization/leveling.
29	TRTODT	RW	0h	Read to ODT delay. This parameter is only used when the PHY issues commands during initialization and leveling. During normal operation, the controller issues commands to the SDRAM and uses the timing parameters programmed in the controller. TI recommends that this bit be always set to 0.
28-19	TDLLK	RW	200h	DLL locking time. Valid values are 2 to 1023.
18-15	TCKE	RW	6h	CKE minimum pulse width. Also specifies the minimum time that the SDRAM must remain in power down or self refresh mode. This parameter must be set to the value of tCKESR which is usually bigger than the value of tCKE. Valid values are 2 to 15.
14-10	TXP	RW	1Ah	Power down exit delay. The minimum time between a power down exit command and any other command. This parameter must be set to the maximum of the various minimum power down exit delay parameters specified in the SDRAM data sheet, that is, max(tXP, tXPDLL). Valid values are 2 to 31.

Table 7-286. DDR_PHY_DTPR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9-0	TXS	RW	200h	Self refresh exit delay. The minimum time between a self refresh exit command and any other command. This parameter must be set to the maximum of the various minimum self refresh exit delay parameters specified in the SDRAM data sheet, that is, (tXS, tXSDLL) for DDR3. Valid values are 2 to 1023.

Table 7-287. Register Call Summary for DDR_PHY_DTPR2

EMIF Registers
- EMIF Registers: [0]

7.2.5.53 DDR_PHY_MR0 Register (Offset = 54h) [reset = A53h]

SDRAM definitions controlled by the Mode Register 0 (DDR_PHY_MR0) include, among other things, burst length, burst type, CAS latency, operating mode, DLL reset, write recovery and power-down modes.

Table 7-288. DDR_PHY_MR0 Instances

Instance	Physical Address
DDR_PHY	0232 9054h

Figure 7-99. DDR_PHY_MR0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			PD	WR			DR
R-0h			RW-0h	RW-5h			RW-0h
7	6	5	4	3	2	1	0
TM	CL_UP			BT	CL_LOW	BL	
RW-0h	RW-5h			RW-0h	RW-0h	RW-3h	

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-289. DDR_PHY_MR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Returns zeros on reads. Bits [15-13] are JEDEC reserved bits and are recommended by JEDEC to be programmed to ‘0’.
12	PD	RW	0h	Power-Down Control: Controls the exit time for power-down modes. This must always be set to 1.
11-9	WR	RW	5h	Write Recovery: This is the value of the write recovery in clock cycles. It is calculated by dividing the data sheet write recovery time, tWR (ns) by the data sheet clock cycle time, tCK (ns) and rounding up a non-integer value to the next integer. Valid values are: • 001 = 5 • 010 = 6 • 011 = 7 • 100 = 8 • 101 = 10 • 110 = 12 • 111 = 14 • 000 = 16 All other settings are reserved and should not be used.
8	DR	RW	0h	DLL Reset: Writing a ‘1’ to this bit will reset the SDRAM DLL. This bit is self-clearing, that is, it returns back to ‘0’ after the DLL reset has been issued.
7	TM	RW	0h	Operating Mode: Always set to 0 for normal operating mode.

Table 7-289. DDR_PHY_MR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-4	CL_UP	RW	5h	**NOTE: bit [2] of this register is also part of the CL field.** CAS Latency: The delay, in clock cycles, between SDRAM read command to data available. Valid values are: • 0010 = 5 • 0100 = 6 • 0110 = 7 • 1000 = 8 • 1010 = 9 • 1100 = 10 • 1110 = 11 • 0001 = 12 • 0011 = 13 • 0101 = 14 All other settings are reserved and should not be used.
3	BT	RW	0h	Burst type: Set to 0 for sequential burst (interleaved burst is not supported)
2	CL_LOW	RW	0h	This is the least significant bit of the CL field. For description, see bits [6-4].
1-0	BL	RW	3h	Burst Length: Determines the maximum number of column locations that can be accessed during a given read or write command. Set to 0 for a fixed burst length of 8 (other burst lengths are not supported)

Table 7-290. Register Call Summary for DDR_PHY_MR0

EMIF Registers
- DDR_PHY_MR0 Register (Offset = 54h) [reset = A53h]: [0]
- EMIF Registers: [0]

7.2.5.54 DDR_PHY_MR1 Register (Offset = 58h) [reset = 0h]

SDRAM definitions controlled by the Mode register 1 (DDR_PHY_MR1) include, among other things, DLL enable/disable, drive strength, on-die termination (ODT) resistance, CAS additive latency, off-chip driver (OCD) impedance calibration and output enable.

Table 7-291. DDR_PHY_MR1 Instances

Instance	Physical Address
DDR_PHY	0232 9058h

Figure 7-100. DDR_PHY_MR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			QOFF	TDQS	RESERVED	RTT_UP	RESERVED
R-0h			RW-0h	RW-0h	R-0h	RW-0h	R-0h
7	6	5	4	3	2	1	0
LEVEL	RTT_MID	DIC_UP	AL		RTT_LOW	DIC_LOW	DE
RW-0h	RW-0h	RW-0h	RW-0h		RW-0h	RW-0h	RW-0h

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-292. DDR_PHY_MR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Returns zeros on reads. Bits [15-13] are JEDEC reserved bits and are recommended by JEDEC to be programmed to ‘0’.
12	QOFF	RW	0h	Output Enable/Disable: Program to ‘0’ for all outputs to function as normal.
11	TDQS	RW	0h	Termination Data Strobe: This must always be set to 0.
10	RESERVED	R	0h	Reserved. This is a JEDEC reserved bit for DDR3 and is recommended by JEDEC to be programmed to ‘0’.
9	RTT_UP	RW	0h	On Die Termination: Selects the effective resistance for SDRAM on die termination. Valid values are: • 000 = ODT disabled • 001 = RZQ/4 • 010 = RZQ/2 • 011 = RZQ/6 • 100 = RZQ/12 • 101 = RZQ/8 All other settings are reserved and should not be used **NOTE: This bit is used together with bit [6] and bit [2] of this register.**
8	RESERVED	R	0h	Reserved. This is a JEDEC reserved bit for DDR3 and is recommended by JEDEC to be programmed to ‘0’.
7	LEVEL	RW	0h	Write Leveling Enable: Enables write-leveling when set.

Table 7-292. DDR_PHY_MR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	RTT_MID	RW	0h	On Die Termination: Selects the effective resistance for SDRAM on die termination. Valid values are: • 000 = ODT disabled • 001 = RZQ/4 • 010 = RZQ/2 • 011 = RZQ/6 • 100 = RZQ/12 • 101 = RZQ/8 All other settings are reserved and should not be used **NOTE: This bit is used together with bit [9] and bit [2] of this register.**
5	DIC_UP	RW	0h	Output Driver Impedance Control: Controls the output drive strength. Valid values are: • 00 = Reserved for RZQ/6 • 01 = RZQ7 • 10 = Reserved • 11 = Reserved **NOTE: This bit is used together with bit [1] of this register.**
4-3	AL	RW	0h	Posted CAS Additive Latency: the controller does not support this feature. This must always be set to 0.
2	RTT_LOW	RW	0h	On Die Termination: Selects the effective resistance for SDRAM on die termination. Valid values are: • 000 = ODT disabled • 001 = RZQ/4 • 010 = RZQ/2 • 011 = RZQ/6 • 100 = RZQ/12 • 101 = RZQ/8 All other settings are reserved and should not be used **NOTE: This bit is used together with bit [9] and bit [6] of this register.**
1	DIC_LOW	RW	0h	Output Driver Impedance Control: Controls the output drive strength. Valid values are: • 00 = Reserved for RZQ/6 • 01 = RZQ7 • 10 = Reserved • 11 = Reserved **NOTE: This bit is used together with bit [5] of this register.**
0	DE	RW	0h	DLL Enable/Disable: Enable (0) or disable (1) the DLL. DLL must be enabled for normal operation.

Table 7-293. Register Call Summary for DDR_PHY_MR1

EMIF Registers
- DDR_PHY_MR1 Register (Offset = 58h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.55 DDR_PHY_MR2 Register (Offset = 5Ch) [reset = 0h]

Mode Register 2 controls (among other things) refresh related features of SDRAMs, and is described in the figure and table below.

Table 7-294. DDR_PHY_MR2 Instances

Instance	Physical Address
DDR_PHY	0232 905Ch

Figure 7-101. DDR_PHY_MR2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					RTTWR		RESERVED
R-0h					RW-0h		R-0h
7	6	5	4	3	2	1	0
SRT	ASR	CWL			PASR		
RW-0h	RW-0h	RW-0h			RW-0h		

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-295. DDR_PHY_MR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Returns zeros on reads. Bits [15-11] are JEDEC reserved bits and are recommended by JEDEC to be programmed to '0'.
10-9	RTTWR	RW	0h	Dynamic ODT: Selects RTT for dynamic ODT. Valid values are: • 00 = Dynamic ODT off • 01 = RZQ/4 • 10 = RZQ/2 • 11 = Reserved
8	RESERVED	R	0h	These are JEDEC reserved bits and are recommended by JEDEC to be programmed to '0'.
7	SRT	RW	0h	Self-Refresh Temperature Range: Selects either normal ('0') or extended ('1') operating temperature range during self-refresh.
6	ASR	RW	0h	Auto Self-Refresh: When enabled ('1'), SDRAM automatically provides self-refresh power management functions for all supported operating temperature values. Otherwise the SRT bit must be programmed to indicate the temperature range.
5-3	CWL	RW	0h	CAS Write Latency: The delay, in clock cycles, between when the SDRAM registers a write command to when write data is available. Valid values are: • 000 = 5 (tCK > 2.5ns) • 001 = 6 (2.5ns > tCK > 1.875ns) • 010 = 7 (1.875ns > tCK> 1.5ns) • 011 = 8 (1.5ns > tCK > 1.25ns) • 100 = 9 (1.25ns > tCK > 1.07ns) • 101 = 10 (1.07ns > tCK > 0.935ns) • 110 = 11 (0.935ns > tCK > 0.833ns) • 111 = 12 (0.833ns > tCK > 0.75ns) All other settings are reserved and should not be used

Table 7-295. DDR_PHY_MR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	PASR	RW	0h	Partial Array Self Refresh: Specifies that data located in areas of the array beyond the specified location will be lost if self refresh is entered. • 000 = Full Array • 001 = Half Array (BA[2:0] = 000, 001, 010 & 011) • 010 = Quarter Array (BA[2:0] = 000, 001) • 011 = 1/8 Array (BA[2:0] = 000) • 100 = 3/4 Array (BA[2:0] = 010, 011, 100, 101, 110 & 111) • 101 = Half Array (BA[2:0] = 100, 101, 110 & 111) • 110 = Quarter Array (BA[2:0] = 110 & 111) • 111 = 1/8 Array (BA[2:0] 111)

Table 7-296. Register Call Summary for DDR_PHY_MR2

EMIF Functional Description • SDRAM Extended Temperature Range: [0][1]
EMIF Registers • EMIF Registers: [0]

7.2.5.56 DDR_PHY_MR3 Register (Offset = 60h) [reset = 0h]

Mode Register 3 controls (among other things) the multi-purpose register, and is described in the figure and table below.

Table 7-297. DDR_PHY_MR3 Instances

Instance	Physical Address
DDR_PHY	0232 9060h

Figure 7-102. DDR_PHY_MR3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					MPR	MPRLOC	
R-0h					RW-0h	RW-0h	

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-298. DDR_PHY_MR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Returns zeros on reads. Bits [15-3] are JEDEC reserved bits and are recommended by JEDEC to be programmed to '0'.
2	MPR	RW	0h	Multi-Purpose Register Enable: Enables, if set, that read data should come from the Multi-Purpose Register. Otherwise read data come from the DRAM array.
1-0	MPRLOC	RW	0h	Multi-Purpose Register (MPR) Location: Selects MPR data location: Valid value are: • 00 = Predefined pattern for system calibration • All other settings are reserved and should not be used.

Table 7-299. Register Call Summary for DDR_PHY_MR3

EMIF Registers
• EMIF Registers: [0]

7.2.5.57 DDR_PHY_ODTCR Register (Offset = 64h) [reset = 84210000h]

This register configures how ODT should be controlled on the different ranks when writing or reading a particular rank. This provides full flexibility on how the overall termination of the SDRAM system is set depending on how many ranks are used and whether the DIMM slots are occupied or not.

For example, assume the system is a 2-rank configuration and during Read commands the user wishes to always enable the ODT on the SDRAM which is not providing the Read data. In this case, the user would write “0010” to RDODT0 and “0001” to RDODT1.

Table 7-300. DDR_PHY_ODTCR Instances

Instance	Physical Address
DDR_PHY	0232 9064h

Figure 7-103. DDR_PHY_ODTCR Register

31	30	29	28	27	26	25	24
WRODT3				WRODT2			
RW-8h				RW-4h			
23	22	21	20	19	18	17	16
WRODT1				WRODT0			
RW-2h				RW-1h			
15	14	13	12	11	10	9	8
RDODT3				RDODT2			
RW-0h				RW-0h			
7	6	5	4	3	2	1	0
RDODT1				RDODT0			
RW-0h				RW-0h			

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-301. DDR_PHY_ODTCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	WRODT3	RW	8h	Write ODT: Specifies whether ODT should be enabled (‘1’) or disabled (‘0’) on each of the up to four ranks when a write command is sent to rank n. WRODT0, WRODT1, WRODT2, and WRODT3 specify ODT settings when a write is to rank 0, rank 1, rank 2, and rank 3, respectively. The four bits of each field each represent a rank, the LSB being rank 0 and the MSB being rank 3. Default is to enable ODT only on rank being written to
27-24	WRODT2	RW	4h	See description for WRODT3.
23-20	WRODT1	RW	2h	See description for WRODT3.
19-16	WRODT0	RW	1h	See description for WRODT3.
15-12	RDODT3	RW	0h	Read ODT: Specifies whether ODT should be enabled (‘1’) or disabled (‘0’) on each of the up to four ranks when a read command is sent to rank n. RDODT0, RDODT1, RDODT2, and RDODT3 specify ODT settings when a read is to rank 0, rank 1, rank 2, and rank 3, respectively. The four bits of each field each represent a rank, the LSB being rank 0 and the MSB being rank 3. Default is to disable ODT during reads.
11-8	RDODT2	RW	0h	See description for RDODT3.
7-4	RDODT1	RW	0h	See description for RDODT3.
3-0	RDODT0	RW	0h	See description for RDODT3.

Table 7-302. Register Call Summary for DDR_PHY_ODTCR

EMIF Registers
- EMIF Registers: [0]

7.2.5.58 DDR_PHY_DTCR Register (Offset = 68h) [reset = 9F003587h]

The DDR_PHY_DTCR register is used to set various configurations for data training, and is described in the figure and register below.

Table 7-303. DDR_PHY_DTCR Instances

Instance	Physical Address
DDR_PHY	0232 9068h

Figure 7-104. DDR_PHY_DTCR Register

31	30	29	28	27	26	25	24
RFSHDT				RANKEN			
RW-9h				RW-Fh			
23	22	21	20	19	18	17	16
RESERVED	DTEXD	DTDSTP	DTDEN	DTDBS			
R-0h	RW-0h	RW-0h	RW-0h	RW-0h			
15	14	13	12	11	10	9	8
RESERVED		DTBDC	DTWBDDM	DTWDQM			
R-0h		RW-1h	RW-1h	RW-5h			
7	6	5	4	3	2	1	0
DTCMPD	DTMPR	DTRANK		DTRPTN			
RW-1h	RW-0h	RW-0h		RW-7h			

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-304. DDR_PHY_DTCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RFSHDT	RW	9h	Refresh During Training: A non-zero value specifies that a burst of refreshes equal to the number specified in this field should be sent to the SDRAM after training each rank except the last rank
27-24	RANKEN	RW	Fh	Rank Enable: Specifies the ranks that are enabled for data-training. Bit 0 controls rank 0, bit 1 controls rank 1, bit 2 controls rank 2, and bit 3 controls rank 3. Setting the bit to '1' enables the rank, and setting it to '0' disables the rank.
23	RESERVED	R	0h	
22	DTEXD	RW	0h	Data Training Extended Write DQS: Enables, if set, an extended write DQS whereby two additional pulses of DQS are added as post-amble to a burst of writes. Generally this should only be enabled when running read bit deskew with the intention of performing read eye deskew prior to running write leveling adjustment
21	DTDSTP	RW	0h	Data Training Debug Step: A write of 1 to this bit steps the data training algorithm through a single step. This bit is self-clearing
20	DTDEN	RW	0h	Data Training Debug Enable: Enables, if set, the data training debug mode
19-16	DTDBS	RW	0h	Data Training Debug Byte Select: Selects the byte during data training debug mod
15-14	RESERVED	R	0h	Returns zeros on reads.
13	DTBDC	RW	1h	Data Training Bit Deskew Centering: Enables, if set, eye centering capability during write and read bit deskew training.
12	DTWBDDM	RW	1h	Data Training Write Bit Deskew Data Mask. If set it enables write bit deskew of the data mask
11-8	DTWDQM	RW	5h	Training WDQ Margin: Defines how close to 0 or how close to 2*(wdq calibration_value) the WDQ lcdl can be moved during training. Basically defines how much timing margin.

Table 7-304. DDR_PHY_DTCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	DTCMPD	RW	1h	Data Training Compare Data: Specifies, if set, that DQS gate training should also check if the returning read data is correct. Otherwise data-training only checks if the correct number of DQS edges were returned.
6	DTMPR	RW	0h	Data Training Using MPR (DDR3 Only): Specifies, if set, that DQS gate training should use the SDRAM Multi-Purpose Register (MPR) register. Otherwise data training is performed by first writing to some locations in the SDRAM and then reading them back.
5-4	DTRANK	RW	0h	Data Training Rank: Selects the SDRAM rank to be used during data bit deskew and eye centering
3-0	DTRPTN	RW	7h	Data Training Repeat Number: Repeat number used to confirm stability of DDR write or read

Table 7-305. Register Call Summary for DDR_PHY_DTCR

EMIF Registers
- DDR_PHY_DTCR Register (Offset = 68h) [reset = 9F003587h]: [0]
- EMIF Registers: [0]

7.2.5.59 DDR_PHY_ZQ0CR0 Register (Offset = 180h) [reset = 4000014Ah]

This register is used for impedance control and calibration to enable the programmable and PVT-compensated on-die termination (ODT) and output impedance of the functional SSTL cells. The following figure and table describe the bits of the ZQnCR0 register.

Table 7-306. DDR_PHY_ZQ0CR0 Instances

Instance	Physical Address
DDR_PHY	0232 9180h

Figure 7-105. DDR_PHY_ZQ0CR0 Register

31	30	29	28	27	26	25	24
ZQPD	ZCALEN	ZCALBYP	ZDEN	ZDATA			
RW-0h	RW-1h	RW-0h	RW-0h	RW-014Ah			
23	22	21	20	19	18	17	16
ZDATA							
RW-014Ah							
15	14	13	12	11	10	9	8
ZDATA							
RW-014Ah							
7	6	5	4	3	2	1	0
ZDATA							
RW-014Ah							

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-307. DDR_PHY_ZQ0CR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ZQPD	RW	0h	ZQ Power Down: Powers down, if set, the impedance control block.
30	ZCALEN	RW	1h	Impedance Calibration Enable: Enables if set the impedance calibration of the ZQn control block when impedance calibration is triggered using either the ZCAL bit of DDR_PHY_PIR register or the DFI update interface.
29	ZCALBYP	RW	0h	Impedance Calibration Bypass: Bypasses, if set, impedance calibration of the ZQn control block when impedance calibration is already in progress. Impedance calibration can be disabled prior to trigger by using the ZCALEN bit.
28	ZDEN	RW	0h	Impedance Over-ride Enable: When this bit is set, it allows users to directly drive the impedance control using the data programmed in the ZDATA field. Otherwise, the control is generated automatically by the impedance control logic
27-0	ZDATA	RW	014Ah	Impedance Over-Ride Data: Data used to directly drive the impedance control. ZDATA field mapping is as follows: • ZDATA[27:21] is used to select the pull-up on-die termination impedance • ZDATA[20:14] is used to select the pull-down on-die termination impedance • ZDATA[13:7] is used to select the pull-up output impedance • ZDATA[6:0] is used to select the pull-down output impedance

Table 7-308. Register Call Summary for DDR_PHY_ZQ0CR0

EMIF Registers
• EMIF Registers: [0]

7.2.5.60 DDR_PHY_ZQ0CR1 Register (Offset = 184h) [reset = 4027Bh]

This register is used for impedance control and calibration to enable the programmable and PVT-compensated on-die termination (ODT) and output impedance of the functional SSTL cells. The following figure and table describe the bits of the ZQnCR1 register.

Table 7-309. DDR_PHY_ZQ0CR1 Instances

Instance	Physical Address
DDR_PHY	0232 9184h

Figure 7-106. DDR_PHY_ZQ0CR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-2h							
23	22	21	20	19	18	17	16
RESERVED							DFIPU0
R-2h							R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-2h							
7	6	5	4	3	2	1	0
ZPROG							
RW-7Bh							

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 7-310. DDR_PHY_ZQ0CR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	2h	Returns zeros on reads.
16	DFIPU0	R	0h	Impedance Calibration Enable: Setting to 1 enables the PHY to request periodic updates to compensate for VT drifts that might result in decreased timing margin. TI recommends this be set to 1.
15-8	RESERVED	R	2h	Returns zeros on reads.
7-0	ZPROG	RW	7Bh	Impedance Divide Ratio: Selects the external resistor divide ratio to be used to set the output impedance and the on-die termination as follows: • ZPROG[7:4] = On-die termination divide select. Available values are: – 2 decimal – 120 ohms – 5 decimal – 60 ohms – 8 decimal – 40 ohms • ZPROG[3:0] = Output impedance divide select. Available values are: – 11 decimal – 40 ohms – 13 decimal – 34 ohms

Table 7-311. Register Call Summary for DDR_PHY_ZQ0CR1

EMIF Registers
- EMIF Registers: [0]

7.2.5.61 DDR_PHY_ZQ0SR0 Register (Offset = 188h) [reset = 14Ah]

This register is used to provide the status for impedance control and calibration to enable the programmable and PVT-compensated on-die termination (ODT) and output impedance of the functional SSTL cells. The following figure and table describe the bits of the ZQnSR0 register.

Table 7-312. DDR_PHY_ZQ0SR0 Instances

Instance	Physical Address
DDR_PHY	0232 9188h

Figure 7-107. DDR_PHY_ZQ0SR0 Register

31	30	29	28	27	26	25	24
ZDONE	ZERR	RESERVED		ZCTRL			
R-0h	R-0h	R-0h		R-000014Ah			
23	22	21	20	19	18	17	16
ZCTRL							
R-000014Ah							
15	14	13	12	11	10	9	8
ZCTRL							
R-000014Ah							
7	6	5	4	3	2	1	0
ZCTRL							
R-000014Ah							

LEGEND: R = Read Only; *-n* = value after reset

Table 7-313. DDR_PHY_ZQ0SR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ZDONE	R	0h	Impedance Calibration Done: Indicates that impedance calibration has completed.
30	ZERR	R	0h	Impedance Calibration Error: If set, indicates that there was an error during impedance calibration.
29-28	RESERVED	R	0h	Returns zeros on reads.
27-0	ZCTRL	R	000014Ah	Impedance Control: Current value of impedance control. ZCTRL field mapping is as follows: • ZCTRL[27:21] is used to select the pull-up on-die termination impedance • ZCTRL[20:14] is used to select the pull-down on-die termination impedance • ZCTRL[13:7] is used to select the pull-up output impedance • ZCTRL[6:0] is used to select the pull-down output impedance

Table 7-314. Register Call Summary for DDR_PHY_ZQ0SR0

EMIF Registers
- EMIF Registers: [0]

7.2.5.62 DDR_PHY_ZQ0SR1 Register (Offset = 18Ch) [reset = 0h]

This register is used to provide the status for impedance control and calibration to enable the programmable and PVT-compensated on-die termination (ODT) and output impedance of the functional SSTL cells. The following figure and table describe the bits of the ZQnSR1 register.

Table 7-315. DDR_PHY_ZQ0SR1 Instances

Instance	Physical Address
DDR_PHY	0232 918Ch

Figure 7-108. DDR_PHY_ZQ0SR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
OPU		OPD		ZPU		ZPD	
R-0h		R-0h		R-0h		R-0h	

LEGEND: R = Read Only; *-n* = value after reset

Table 7-316. DDR_PHY_ZQ0SR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Returns zeros on reads.
7-6	OPU	R	0h	On-die termination (ODT) pull-up calibration status. Similar status encodings as ZPD.
5-4	OPD	R	0h	On-die termination (ODT) pull-down calibration status. Similar status encodings as ZPD.
3-2	ZPU	R	0h	Output impedance pull-up calibration status. Similar status encodings as ZPD.
1-0	ZPD	R	0h	Output impedance pull-down calibration status. Valid status encodings are: • 00 = Completed with no errors • 01 = Overflow error • 10 = Underflow error • 11 = Calibration in progress

Table 7-317. Register Call Summary for DDR_PHY_ZQ0SR1

EMIF Registers
• EMIF Registers: [0]

7.2.5.63 DDR_PHY_ZQ1CR0 Register (Offset = 190h) [reset = 4000014Ah]

This register is used for impedance control and calibration to enable the programmable and PVT-compensated on-die termination (ODT) and output impedance of the functional SSTL cells. The following figure and table describe the bits of the ZQnCR0 register.

Table 7-318. DDR_PHY_ZQ1CR0 Instances

Instance	Physical Address
DDR_PHY	0232 9190h

Figure 7-109. DDR_PHY_ZQ1CR0 Register

31	30	29	28	27	26	25	24
ZQPD	ZCALEN	ZCALBYP	ZDEN	ZDATA			
RW-0h	RW-1h	RW-0h	RW-0h	RW-014Ah			
23	22	21	20	19	18	17	16
ZDATA							
RW-014Ah							
15	14	13	12	11	10	9	8
ZDATA							
RW-014Ah							
7	6	5	4	3	2	1	0
ZDATA							
RW-014Ah							

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-319. DDR_PHY_ZQ1CR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ZQPD	RW	0h	ZQ Power Down: Powers down, if set, the impedance control block.
30	ZCALEN	RW	1h	Impedance Calibration Enable: Enables if set the impedance calibration of the ZQn control block when impedance calibration is triggered using either the ZCAL bit of DDR_PHY_PIR register or the DFI update interface.
29	ZCALBYP	RW	0h	Impedance Calibration Bypass: Bypasses, if set, impedance calibration of the ZQn control block when impedance calibration is already in progress. Impedance calibration can be disabled prior to trigger by using the ZCALEN bit.
28	ZDEN	RW	0h	Impedance Over-ride Enable: When this bit is set, it allows users to directly drive the impedance control using the data programmed in the ZDATA field. Otherwise, the control is generated automatically by the impedance control logic
27-0	ZDATA	RW	014Ah	Impedance Over-Ride Data: Data used to directly drive the impedance control. ZDATA field mapping is as follows: • ZDATA[27:21] is used to select the pull-up on-die termination impedance • ZDATA[20:14] is used to select the pull-down on-die termination impedance • ZDATA[13:7] is used to select the pull-up output impedance • ZDATA[6:0] is used to select the pull-down output impedance

Table 7-320. Register Call Summary for DDR_PHY_ZQ1CR0

EMIF Registers
• EMIF Registers: [0]

7.2.5.64 DDR_PHY_ZQ1CR1 Register (Offset = 194h) [reset = 4027Bh]

This register is used for impedance control and calibration to enable the programmable and PVT-compensated on-die termination (ODT) and output impedance of the functional SSTL cells. The following figure and table describe the bits of the ZQnCR1 register.

Table 7-321. DDR_PHY_ZQ1CR1 Instances

Instance	Physical Address
DDR_PHY	0232 9194h

Figure 7-110. DDR_PHY_ZQ1CR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-2h							
23	22	21	20	19	18	17	16
RESERVED							DFIPU0
R-2h							R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-2h							
7	6	5	4	3	2	1	0
ZPROG							
RW-7Bh							

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 7-322. DDR_PHY_ZQ1CR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	2h	Returns zeros on reads.
16	DFIPU0	R	0h	Impedance Calibration Enable: Setting to 1 enables the PHY to request periodic updates to compensate for VT drifts that might result in decreased timing margin. TI recommends this be set to 1.
15-8	RESERVED	R	2h	Returns zeros on reads.
7-0	ZPROG	RW	7Bh	Impedance Divide Ratio: Selects the external resistor divide ratio to be used to set the output impedance and the on-die termination as follows: • ZPROG[7:4] = On-die termination divide select. Available values are: – 2 decimal – 120 ohms – 5 decimal – 60 ohms – 8 decimal – 40 ohms • ZPROG[3:0] = Output impedance divide select. Available values are: – 11 decimal – 40 ohms – 13 decimal – 34 ohms

Table 7-323. Register Call Summary for DDR_PHY_ZQ1CR1

EMIF Registers
• EMIF Registers: [0]

7.2.5.65 DDR_PHY_ZQ1SR0 Register (Offset = 198h) [reset = 14Ah]

This register is used to provide the status for impedance control and calibration to enable the programmable and PVT-compensated on-die termination (ODT) and output impedance of the functional SSTL cells. The following figure and table describe the bits of the ZQnSR0 register.

Table 7-324. DDR_PHY_ZQ1SR0 Instances

Instance	Physical Address
DDR_PHY	0232 9198h

Figure 7-111. DDR_PHY_ZQ1SR0 Register

31	30	29	28	27	26	25	24
ZDONE	ZERR	RESERVED		ZCTRL			
R-0h	R-0h	R-0h		R-000014Ah			
23	22	21	20	19	18	17	16
ZCTRL							
R-000014Ah							
15	14	13	12	11	10	9	8
ZCTRL							
R-000014Ah							
7	6	5	4	3	2	1	0
ZCTRL							
R-000014Ah							

LEGEND: R = Read Only; *-n* = value after reset

Table 7-325. DDR_PHY_ZQ1SR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ZDONE	R	0h	Impedance Calibration Done: Indicates that impedance calibration has completed.
30	ZERR	R	0h	Impedance Calibration Error: If set, indicates that there was an error during impedance calibration.
29-28	RESERVED	R	0h	Returns zeros on reads.
27-0	ZCTRL	R	000014Ah	Impedance Control: Current value of impedance control. ZCTRL field mapping is as follows: • ZCTRL[27:21] is used to select the pull-up on-die termination impedance • ZCTRL[20:14] is used to select the pull-down on-die termination impedance • ZCTRL[13:7] is used to select the pull-up output impedance • ZCTRL[6:0] is used to select the pull-down output impedance

Table 7-326. Register Call Summary for DDR_PHY_ZQ1SR0

EMIF Registers
- EMIF Registers: [0]

7.2.5.66 DDR_PHY_ZQ1SR1 Register (Offset = 19Ch) [reset = 0h]

This register is used to provide the status for impedance control and calibration to enable the programmable and PVT-compensated on-die termination (ODT) and output impedance of the functional SSTL cells. The following figure and table describe the bits of the ZQnSR1 register.

Table 7-327. DDR_PHY_ZQ1SR1 Instances

Instance	Physical Address
DDR_PHY	0232 919Ch

Figure 7-112. DDR_PHY_ZQ1SR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
OPU		OPD		ZPU		ZPD	
R-0h		R-0h		R-0h		R-0h	

LEGEND: R = Read Only; *-n* = value after reset

Table 7-328. DDR_PHY_ZQ1SR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Returns zeros on reads.
7-6	OPU	R	0h	On-die termination (ODT) pull-up calibration status. Similar status encodings as ZPD.
5-4	OPD	R	0h	On-die termination (ODT) pull-down calibration status. Similar status encodings as ZPD.
3-2	ZPU	R	0h	Output impedance pull-up calibration status. Similar status encodings as ZPD.
1-0	ZPD	R	0h	Output impedance pull-down calibration status. Valid status encodings are: • 00 = Completed with no errors • 01 = Overflow error • 10 = Underflow error • 11 = Calibration in progress

Table 7-329. Register Call Summary for DDR_PHY_ZQ1SR1

EMIF Registers
• EMIF Registers: [0]

7.2.5.67 DDR_PHY_ZQ2CR0 Register (Offset = 1A0h) [reset = 4000014Ah]

This register is used for impedance control and calibration to enable the programmable and PVT-compensated on-die termination (ODT) and output impedance of the functional SSTL cells. The following figure and table describe the bits of the ZQnCR0 register.

Table 7-330. DDR_PHY_ZQ2CR0 Instances

Instance	Physical Address
DDR_PHY	0232 91A0h

Figure 7-113. DDR_PHY_ZQ2CR0 Register

31	30	29	28	27	26	25	24
ZQPD	ZCALEN	ZCALBYP	ZDEN	ZDATA			
RW-0h	RW-1h	RW-0h	RW-0h	RW-014Ah			
23	22	21	20	19	18	17	16
ZDATA							
RW-014Ah							
15	14	13	12	11	10	9	8
ZDATA							
RW-014Ah							
7	6	5	4	3	2	1	0
ZDATA							
RW-014Ah							

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-331. DDR_PHY_ZQ2CR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ZQPD	RW	0h	ZQ Power Down: Powers down, if set, the impedance control block.
30	ZCALEN	RW	1h	Impedance Calibration Enable: Enables if set the impedance calibration of the ZQn control block when impedance calibration is triggered using either the ZCAL bit of DDR_PHY_PIR register or the DFI update interface.
29	ZCALBYP	RW	0h	Impedance Calibration Bypass: Bypasses, if set, impedance calibration of the ZQn control block when impedance calibration is already in progress. Impedance calibration can be disabled prior to trigger by using the ZCALEN bit.
28	ZDEN	RW	0h	Impedance Over-ride Enable: When this bit is set, it allows users to directly drive the impedance control using the data programmed in the ZDATA field. Otherwise, the control is generated automatically by the impedance control logic
27-0	ZDATA	RW	014Ah	Impedance Over-Ride Data: Data used to directly drive the impedance control. ZDATA field mapping is as follows: • ZDATA[27:21] is used to select the pull-up on-die termination impedance • ZDATA[20:14] is used to select the pull-down on-die termination impedance • ZDATA[13:7] is used to select the pull-up output impedance • ZDATA[6:0] is used to select the pull-down output impedance

Table 7-332. Register Call Summary for DDR_PHY_ZQ2CR0

EMIF Registers
• EMIF Registers: [0]

7.2.5.68 DDR_PHY_ZQ2CR1 Register (Offset = 1A4h) [reset = 4027Bh]

This register is used for impedance control and calibration to enable the programmable and PVT-compensated on-die termination (ODT) and output impedance of the functional SSTL cells. The following figure and table describe the bits of the ZQnCR1 register.

Table 7-333. DDR_PHY_ZQ2CR1 Instances

Instance	Physical Address
DDR_PHY	0232 91A4h

Figure 7-114. DDR_PHY_ZQ2CR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-2h							
23	22	21	20	19	18	17	16
RESERVED							DFIPU0
R-2h							R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-2h							
7	6	5	4	3	2	1	0
ZPROG							
RW-7Bh							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-334. DDR_PHY_ZQ2CR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	2h	Returns zeros on reads.
16	DFIPU0	R	0h	Impedance Calibration Enable: Setting to 1 enables the PHY to request periodic updates to compensate for VT drifts that might result in decreased timing margin. TI recommends this be set to 1.
15-8	RESERVED	R	2h	Returns zeros on reads.
7-0	ZPROG	RW	7Bh	Impedance Divide Ratio: Selects the external resistor divide ratio to be used to set the output impedance and the on-die termination as follows: • ZPROG[7:4] = On-die termination divide select. Available values are: – 2 decimal – 120 ohms – 5 decimal – 60 ohms – 8 decimal – 40 ohms • ZPROG[3:0] = Output impedance divide select. Available values are: – 11 decimal – 40 ohms – 13 decimal – 34 ohms

Table 7-335. Register Call Summary for DDR_PHY_ZQ2CR1

EMIF Registers
- EMIF Registers: [0]

7.2.5.69 DDR_PHY_ZQ2SR0 Register (Offset = 1A8h) [reset = 14Ah]

This register is used to provide the status for impedance control and calibration to enable the programmable and PVT-compensated on-die termination (ODT) and output impedance of the functional SSTL cells. The following figure and table describe the bits of the ZQnSR0 register.

Table 7-336. DDR_PHY_ZQ2SR0 Instances

Instance	Physical Address
DDR_PHY	0232 91A8h

Figure 7-115. DDR_PHY_ZQ2SR0 Register

31	30	29	28	27	26	25	24
ZDONE	ZERR	RESERVED		ZCTRL			
R-0h	R-0h	R-0h		R-000014Ah			
23	22	21	20	19	18	17	16
ZCTRL							
R-000014Ah							
15	14	13	12	11	10	9	8
ZCTRL							
R-000014Ah							
7	6	5	4	3	2	1	0
ZCTRL							
R-000014Ah							

LEGEND: R = Read Only; *-n* = value after reset

Table 7-337. DDR_PHY_ZQ2SR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ZDONE	R	0h	Impedance Calibration Done: Indicates that impedance calibration has completed.
30	ZERR	R	0h	Impedance Calibration Error: If set, indicates that there was an error during impedance calibration.
29-28	RESERVED	R	0h	Returns zeros on reads.
27-0	ZCTRL	R	000014Ah	Impedance Control: Current value of impedance control. ZCTRL field mapping is as follows: • ZCTRL[27:21] is used to select the pull-up on-die termination impedance • ZCTRL[20:14] is used to select the pull-down on-die termination impedance • ZCTRL[13:7] is used to select the pull-up output impedance • ZCTRL[6:0] is used to select the pull-down output impedance

Table 7-338. Register Call Summary for DDR_PHY_ZQ2SR0

EMIF Registers
- EMIF Registers: [0]

7.2.5.70 DDR_PHY_ZQ2SR1 Register (Offset = 1ACh) [reset = 0h]

This register is used to provide the status for impedance control and calibration to enable the programmable and PVT-compensated on-die termination (ODT) and output impedance of the functional SSTL cells. The following figure and table describe the bits of the ZQnSR1 register.

Table 7-339. DDR_PHY_ZQ2SR1 Instances

Instance	Physical Address
DDR_PHY	0232 91ACh

Figure 7-116. DDR_PHY_ZQ2SR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
OPU		OPD		ZPU		ZPD	
R-0h		R-0h		R-0h		R-0h	

LEGEND: R = Read Only; *-n* = value after reset

Table 7-340. DDR_PHY_ZQ2SR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Returns zeros on reads.
7-6	OPU	R	0h	On-die termination (ODT) pull-up calibration status. Similar status encodings as ZPD.
5-4	OPD	R	0h	On-die termination (ODT) pull-down calibration status. Similar status encodings as ZPD.
3-2	ZPU	R	0h	Output impedance pull-up calibration status. Similar status encodings as ZPD.
1-0	ZPD	R	0h	Output impedance pull-down calibration status. Valid status encodings are: • 00 = Completed with no errors • 01 = Overflow error • 10 = Underflow error • 11 = Calibration in progress

Table 7-341. Register Call Summary for DDR_PHY_ZQ2SR1

EMIF Registers
• EMIF Registers: [0]

7.2.5.71 DDR_PHY_DX0GCR Register (Offset = 1C0h) [reset = 7C000E81h]

This register is used for miscellaneous configurations specific to a particular instantiation of the DATX8 macro.

Table 7-342. DDR_PHY_DX0GCR Instances

Instance	Physical Address
DDR_PHY	0232 91C0h

Figure 7-117. DDR_PHY_DX0GCR Register

31	30	29	28	27	26	25	24
CALBYP	MDLEN	WLRNKEN				RESERVED	
RW-0h	RW-1h	RW-Fh				R-0h	
23	22	21	20	19	18	17	16
RESERVED				PLLBYP	GSHIFT	PLLPD	PLLRST
R-0h				RW-0h	RW-0h	RW-0h	RW-0h
15	14	13	12	11	10	9	8
DXOEO		RTTOAL	RTTOH		DQRTT	DQSRTT	DSEN
RW-0h		RW-0h	RW-1h		RW-1h	RW-1h	RW-1h
7	6	5	4	3	2	1	0
DSEN	DQSRPD	DXPDR	DXPDD	DXIOM	DQODT	DQSODT	DXEN
RW-1h	RW-0h	RW-0h	RW-0h	RW-0h	RW-0h	RW-0h	RW-1h

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-343. DDR_PHY_DX0GCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CALBYP	RW	0h	Calibration Bypass: Prevents, if set, period measurement calibration from automatically triggering after PHY initialization.
30	MDLEN	RW	1h	Master Delay Line Enable: Enables, if set, the DATX8 master delay line calibration to perform subsequent period measurements following the initial period measurements that are performed after reset or when calibration is manually triggered. These additional measurements are accumulated and filtered as long as this bit remains high. This bit is ANDed with the common DATX8 MDL enable bit.
29-26	WLRNKEN	RW	Fh	Write Level Rank Enable: Specifies the ranks that should be write leveled for this byte. Write leveling responses from ranks that are not enabled for write leveling for a particular byte are ignored and write leveling is flagged as done for these ranks. WLRKEN[0] enables rank 0, [1] enables rank 1, [2] enables rank 2, and [3] enables rank 3.
25-20	RESERVED	R	0h	Reads return zeros
19	PLLBYP	RW	0h	PLL Bypass: Puts the byte PLL in bypass mode by driving the PLL bypass pin. This bit is not self-clearing and a '0' must be written to de-assert the bypass. This bit is ORed with the global BYP configuration bit in DDR_PHY_PLLCR
18	GSHIFT	RW	0h	Gear Shift: Enables, if set, rapid locking mode on the byte PLL
17	PLLPD	RW	0h	PLL Power Down: Puts the byte PLL in power down mode by driving the PLL power down pin. This bit is not self-clearing and a '0' must be written to de-assert the power-down. This bit is ORed with the global PLLPD configuration bit in DDR_PHY_PLLCR
16	PLLRST	RW	0h	PLL Rest: Resets the byte PLL by driving the PLL reset pin. This bit is not selfclearing and a '0' must be written to de-assert the reset. This bit is ORed with the global PLLRST configuration bit in DDR_PHY_PLLCR

Table 7-343. DDR_PHY_DX0GCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-14	DXOEO	RW	0h	Data Byte Output Enable Override: Specifies whether the output I/O output enable for the byte lane should be set to a fixed value. Valid values are: • 00 = No override. Output enable is controlled by DFI transactions • 01 = Ouput enable is asserted (I/O is forced to output mode). • 10 = Output enable is de-asserted (I/O is forced to input mode) • 11 = Reserved
13	RTTOAL	RW	0h	RTT On Additive Latency: Indicates when the ODT control of DQ/DQS SSTL I/Os is set to the value in DQODT/DQSODT during read cycles. Valid values are: • 0 = ODT control is set to DQSODT/DQODT almost two cycles before read data preamble • 1 = ODT control is set to DQSODT/DQODT almost one cycle before read data preamble
12-11	RTTOH	RW	1h	RTT Output Hold: Indicates the number of clock cycles (from 0 to 3) after the read data postamble for which ODT control should remain set to DQSODT for DQS or DQODT for DQ/DM before disabling it (setting it to '0') when using dynamic ODT control. ODT is disabled almost RTTOH clock cycles after the read postamble
10	DQRTT	RW	1h	DQ Dynamic RTT Control: Indicates, if set, that the ODT control of DQ/DM SSTL I/Os be dynamically controlled by setting it to the value in DQODT during reads and disabling it (setting it to '0') during any other cycle. If this bit is not set, then the ODT control of DQ SSTL I/Os is always set to the value in DQODT.
9	DQSRTT	RW	1h	DQS Dynamic RTT Control: Indicates, if set, that the ODT control of DQS SSTL I/Os be dynamically controlled by setting it to the value in DQSODT during reads and disabling it (setting it to '0') during any other cycle. If this bit is not set, then the ODT control of DQS SSTL I/Os is always set to the value in DQSODT field
8-7	DSEN	RW	1h	Write DQS Enable: Controls whether the write DQS going to the SDRAM is enabled (toggling) or disabled (static value) and whether the DQS is inverted. DQS# is always the inversion of DQS. These values are valid only when DQS/DQS# output enable is on, otherwise the DQS/DQS# is tri-stated. Valid settings are: • 00 = DQS disabled (Driven to constant 0) • 01 = DQS toggling with normal polarity (This should be the default setting) • 10 = DQS toggling with inverted polarity • 11 = DQS disabled (Driven to constant 1)
6	DQSRPD	RW	0h	DQSR Power Down: Powers down, if set, the PDQSR cell.
5	DXPDR	RW	0h	Data Power Down Receiver: Powers down, when set, the input receiver on I/O for DQ, DM, and DQS/DQS# pins of the byte.
4	DXPDD	RW	0h	Data Power Down Driver: Powers down, when set, the output driver on I/O for DQ, DM, and DQS/DQS# pins of the byte.
3	DXIOM	RW	0h	Data I/O Mode: Selects SSTL mode (when set to 0) or CMOS mode (when set to 1) of the I/O for DQ, DM, and DQS/DQS# pins of the byte.
2	DQODT	RW	0h	Data On-Die Termination: Enables, when set, the on-die termination on the I/O for DQ and DM pins of the byte.
1	DQSODT	RW	0h	DQS On-Die Termination: Enables, when set, the on-die termination on the I/O for DQS/DQS# pin of the byte

Table 7-343. DDR_PHY_DX0GCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	DXEN	RW	1h	Data Byte Enable: Enables if set the data byte. Setting this bit to '0' disables the byte, that is, the byte is not used in PHY initialization or training and is ignored during SDRAM read/write operations. Note: Software-initiated PHY initialization and leveling is only required after ECC has been enabled in the PHY by writing to DDR_PHY_DX8GCR.DXEN. Once ECC is enabled in the PHY and the PHY is initialized and leveled, the ECC enable bit in the controller (ECCCTL.ECC_EN) can be enabled or disabled without re-triggering PHY initialization and leveling.

Table 7-344. Register Call Summary for DDR_PHY_DX0GCR

EMIF Registers
- EMIF Registers: [0]

7.2.5.72 DDR_PHY_DX0GSR0 Register (Offset = 1C4h) [reset = 0h]

DDR_PHY_DX0GSR0 and DDR_PHY_DX0GSR2 are general status registers for the DATX8. They indicate among other things whether write leveling or period measurement calibration is done. Note that WDQCAL, RDQSCAL, RDQSNCAL, GDQSCAL, and WLCAL are calibration measurement-done flags that should be used for debug purposes if the global calibration done flag (DDR_PHY_PGSR0[CAL]) is not asserted. These flags will typically assert for a minimum of two VBUSP_CLK cycles and then de-assert when all measurement done flags are asserted.

Table 7-345. DDR_PHY_DX0GSR0 Instances

Instance	Physical Address
DDR_PHY	0232 91C4h

Figure 7-118. DDR_PHY_DX0GSR0 Register

31	30	29	28	27	26	25	24
RESERVED			WLDQ	QSGERR			
R-0h			R-0h	R-0h			
23	22	21	20	19	18	17	16
GDQSPRD							
R-0h							
15	14	13	12	11	10	9	8
DPLOCK	WLPRD						
R-0h	R-0h						
7	6	5	4	3	2	1	0
WLPRD	WLERR	WLDONE	WLCAL	GDQSCAL	RDQSNCAL	RDQSCAL	WDQCAL
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 7-346. DDR_PHY_DX0GSR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reads return zeros.
28	WLDQ	R	0h	Write Leveling DQ Status: Captures the write leveling DQ status from the DRAM during software write leveling
27-24	QSGERR	R	0h	DQS Gate Training Error: Indicates if set that there is an error in DQS gate training. One bit for each of the up to 4 ranks
23-16	GDQSPRD	R	0h	Read DQS gating Period: Returns the DDR clock period measured by the read DQS gating LCDL during calibration. This value is PVT compensated
15	DPLOCK	R	0h	DATX8 PLL Lock: Indicates, if set, that the DATX8 PLL has locked.
14-7	WLPRD	R	0h	Write Leveling Period: Returns the DDR clock period measured by the write leveling LCDL during calibration. The measured period is used to generate the control of the write leveling pipeline which is a function of the write-leveling delay and the clock period. This value is PVT compensated
6	WLERR	R	0h	Write Leveling Error: Indicates, if set, that there is a write leveling error in the DATX8.
5	WLDONE	R	0h	Write Leveling Done: Indicates, if set, that the DATX8 has completed write leveling.
4	WLCAL	R	0h	Write Leveling Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the write leveling slave delay line
3	GDQSCAL	R	0h	Read DQS gating Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the read DQS gating LCDL.

Table 7-346. DDR_PHY_DX0GSR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	RDQSNCAL	R	0h	Read DQS# Calibration (Type B/B1 PHY Only): Indicates, if set, that the DATX8 has finished doing period measurement calibration for the read DQS# LCDL
1	RDQSCAL	R	0h	Read DQS Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the read DQS LCDL.
0	WDQCAL	R	0h	Write DQ Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the write DQ LCDL.

Table 7-347. Register Call Summary for DDR_PHY_DX0GSR0

EMIF Registers
- DDR_PHY_DX0GSR0 Register (Offset = 1C4h) [reset = 0h]: [0]
- DDR_PHY_DX0GSR2 Register (Offset = 1F4h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.73 DDR_PHY_DX0GSR2 Register (Offset = 1F4h) [reset = 0h]

DDR_PHY_DX0GSR0 and DDR_PHY_DX0GSR2 are general status registers for the DATX8. They indicate among other things whether write leveling or period measurement calibration is done. Note that WDQCAL, RDQSCAL, RDQSNCAL, GDQSCAL, and WLCAL are calibration measurement-done flags that should be used for debug purposes if the global calibration done flag (DDR_PHY_PGSR0[CAL]) is not asserted. These flags will typically assert for a minimum of two VBUSP_CLK cycles and then de-assert when all measurement done flags are asserted.

Table 7-348. DDR_PHY_DX0GSR2 Instances

Instance	Physical Address
DDR_PHY	0232 91F4h

Figure 7-119. DDR_PHY_DX0GSR2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				ESTAT			
R-0h				R-0h			
7	6	5	4	3	2	1	0
WEWN	WEERR	REWN	REERR	WDWN	WDERR	RDWN	RDERR
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 7-349. DDR_PHY_DX0GSR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reads return zeros.
11-8	ESTAT	R	0h	Error Status: If an error occurred for this lane as indicated by RDERR, WDERR, REERR or WEERR the error status code can provide additional information regard when the error occurred during the algorithm execution.
7	WEWN	R	0h	Write Eye Centering Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the write eye centering training.
6	WEERR	R	0h	Write Eye Centering Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the write eye centering training.
5	REWN	R	0h	Read Eye Centering Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the read eye centering training.
4	REERR	R	0h	Read Eye Centering Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the read eye centering training.
3	WDWN	R	0h	Write Bit Deskew Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the write bit deskew training.
2	WDERR	R	0h	Write Bit Deskew Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the write bit deskew training.

Table 7-349. DDR_PHY_DX0GSR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	RDWN	R	0h	Read Bit Deskew Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the read bit deskew training.
0	RDERR	R	0h	Read Bit Deskew Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the read bit deskew training.

Table 7-350. Register Call Summary for DDR_PHY_DX0GSR2

EMIF Registers

- DDR_PHY_DX0GSR0 Register (Offset = 1C4h) [reset = 0h]: [0]
- DDR_PHY_DX0GSR2 Register (Offset = 1F4h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.74 DDR_PHY_DX0LCDLR0 Register (Offset = 1E0h) [reset = 0h]

The DATX8 local calibrated delay line registers 0-2 are used to select the delay value on the LCDLs used in the DATX8 macros. A single LCDL field in the LCDLR register connects to a corresponding LCDL. The following tables describe the bits of the DATX8 LCDLR registers. The write data delay (WDQD) and the read DQS delay (RDQSD) are automatically derived from the measured period during calibration. WDQD and RDQSD correspond to a 90 degrees phase shift for DQ during writes and DQS during reads, respectively. The 90 degrees phase shift is used to centre DQS into the write and read data eyes. A 90 degrees phase shift is equivalent to half the DDR clock period. After calibration WDQD and RDQSD fields will contain a value that corresponds to half the DDR clock period (or a quarter of the SDRAM clock period). The write leveling delay (RnWLD) and read DQS gating delay DQSGD are derived from the write leveling and DQS training algorithms. These are normally run automatically by the PHY control block or they can be executed in software by the user. After calibration RnWLD field will contain a value that corresponds to the DDR clock period (or half of the SDRAM clock period), while RnDQSGD field will contain a value that corresponds to half the DDR clock period (or a quarter of the SDRAM clock period).

After the initial setting, all LCDL register fields are automatically updated by the VT compensation logic to track drifts due to voltage and temperature. The user can however override these values by writing to the respective fields of the register and/or optionally disabling the automatic drift compensation.

Table 7-351. DDR_PHY_DX0LCDLR0 Instances

Instance	Physical Address
DDR_PHY	0232 91E0h

Figure 7-120. DDR_PHY_DX0LCDLR0 Register

31	30	29	28	27	26	25	24
R3WLD							
RW-0h							
23	22	21	20	19	18	17	16
R2WLD							
RW-0h							
15	14	13	12	11	10	9	8
R1WLD							
RW-0h							
7	6	5	4	3	2	1	0
R0WLD							
RW-0h							

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-352. DDR_PHY_DX0LCDLR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	R3WLD	RW	0h	Rank 3 Write Leveling Delay: Rank 3 delay select for the write leveling (WL) LCDL
23-16	R2WLD	RW	0h	Rank 2 Write Leveling Delay: Rank 2 delay select for the write leveling (WL) LCDL
15-8	R1WLD	RW	0h	Rank 1 Write Leveling Delay: Rank 1 delay select for the write leveling (WL) LCDL
7-0	R0WLD	RW	0h	Rank 0 Write Leveling Delay: Rank 0 delay select for the write leveling (WL) LCDL

Table 7-353. Register Call Summary for DDR_PHY_DX0LCDLR0

EMIF Registers
- EMIF Registers: [0]

7.2.5.75 DDR_PHY_DX0LCDLR1 Register (Offset = 1E4h) [reset = 0h]

The DXnLCDLR1 register is described in the figure and table below.

Table 7-354. DDR_PHY_DX0LCDLR1 Instances

Instance	Physical Address
DDR_PHY	0232 91E4h

Figure 7-121. DDR_PHY_DX0LCDLR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RDQSND							
RW-0h							
15	14	13	12	11	10	9	8
RDQSD							
RW-0h							
7	6	5	4	3	2	1	0
WDQD							
RW-0h							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-355. DDR_PHY_DX0LCDLR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return zeros
23-16	RDQSND	RW	0h	Read DQSN Delay: Delay select for the read DQSN (RDQS) LCDL
15-8	RDQSD	RW	0h	Read DQS Delay: Delay select for the read DQS (RDQS) LCDL
7-0	WDQD	RW	0h	Write Data Delay: Delay select for the write data (WDQ) LCDL.

Table 7-356. Register Call Summary for DDR_PHY_DX0LCDLR1

EMIF Registers
• EMIF Registers: [0]

7.2.5.76 DDR_PHY_DX0LCDLR2 Register (Offset = 1E8h) [reset = 0h]

The DXnLCDLR2 register is described in the figure and table below.

Table 7-357. DDR_PHY_DX0LCDLR2 Instances

Instance	Physical Address
DDR_PHY	0232 91E8h

Figure 7-122. DDR_PHY_DX0LCDLR2 Register

31	30	29	28	27	26	25	24
R3DQSGD							
RW-0h							
23	22	21	20	19	18	17	16
R2DQSGD							
RW-0h							
15	14	13	12	11	10	9	8
R1DQSGD							
RW-0h							
7	6	5	4	3	2	1	0
R0DQSGD							
RW-0h							

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-358. DDR_PHY_DX0LCDLR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	R3DQSGD	RW	0h	Rank 3 Read DQS Gating Delay: Rank 3 delay select for the read DQS gating (DQSG) LCDL.
23-16	R2DQSGD	RW	0h	Rank 2 Read DQS Gating Delay: Rank 2 delay select for the read DQS gating (DQSG) LCDL.
15-8	R1DQSGD	RW	0h	Rank 1 Read DQS Gating Delay: Rank 1 delay select for the read DQS gating (DQSG) LCDL.
7-0	R0DQSGD	RW	0h	Rank 0 Read DQS Gating Delay: Rank 0 delay select for the read DQS gating (DQSG) LCDL.

Table 7-359. Register Call Summary for DDR_PHY_DX0LCDLR2

EMIF Registers
- EMIF Registers: [0]

7.2.5.77 DDR_PHY_DX0MDLR Register (Offset = 1ECh) [reset = 0h]

The DATX8 Master Delay Line Registers return different period values measured by the master delay lines in the DATX8 macros. In the default normal operating conditions, the value of the DXnMDLR registers change based on ongoing calibration and measurements.

Table 7-360. DDR_PHY_DX0MDLR Instances

Instance	Physical Address
DDR_PHY	0232 91ECh

Figure 7-123. DDR_PHY_DX0MDLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
MDLD							
RW-0h							
15	14	13	12	11	10	9	8
TPRD							
RW-0h							
7	6	5	4	3	2	1	0
IPRD							
RW-0h							

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 7-361. DDR_PHY_DX0MDLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return zeros
23-16	MDLD	RW	0h	MDL Delay: Delay select for the LCDL for the Master Delay Line.
15-8	TPRD	RW	0h	Target Period: Target period measured by the master delay line calibration for VT drift compensation. This is the current measured value of the period and is continuously updated if the MDL is enabled to do so.
7-0	IPRD	RW	0h	Initial Period: Initial period measured by the master delay line calibration for VT drift compensation. This value is used as the denominator when calculating the ratios of updates during VT compensation.

Table 7-362. Register Call Summary for DDR_PHY_DX0MDLR

EMIF Registers
- EMIF Registers: [0]

7.2.5.78 DDR_PHY_DX0GTR Register (Offset = 1F0h) [reset = 55000h]

This register is used to control various timing settings of the byte lane, including DQS gating system latency and write leveling system latency.

Table 7-363. DDR_PHY_DX0GTR Instances

Instance	Physical Address
DDR_PHY	0232 91F0h

Figure 7-124. DDR_PHY_DX0GTR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				R3WLSL		R2WLSL	
R-0h				RW-1h		RW-1h	
15	14	13	12	11	10	9	8
R1WLSL		R0WLSL		R3DGSL			R2DGSL
RW-1h		RW-1h		RW-0h			RW-0h
7	6	5	4	3	2	1	0
R2DGSL		R1DGSL		R0DGSL			
RW-0h		RW-0h		RW-0h			

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-364. DDR_PHY_DX0GTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reads return zeros
19-18	R3WLSL	RW	1h	Rank n Write Leveling System Latency: This is used to adjust the write latency after write leveling. Power-up default is 01 (that is, no extra clock cycles required). The SL fields are initially set by the PUB during automatic write leveling but these values can be overwritten by a direct write to this register. Every two bits of this register control the latency of each of the (up to) four ranks. R0WLSL controls the latency of rank 0, R1WLSL controls rank 1, and so on. Valid values: • 00 = Write latency = WL - 1 • 01 = Write latency = WL • 10 = Write latency = WL + 1 • 11 = Reserved
17-16	R2WLSL	RW	1h	See description for R3WLSL.
15-14	R1WLSL	RW	1h	See description for R3WLSL.
13-12	R0WLSL	RW	1h	See description for R3WLSL.
11-9	R3DGSL	RW	0h	Rank n DQS Gating System Latency: This is used to increase the number of clock cycles needed to expect valid DDR read data by up to seven extra clock cycles. This is used to compensate for board delays and other system delays. Power-up default is 000 (that is, no extra clock cycles required). The SL fields are initially set by the PHY during automatic DQS data training but these values can be overwritten by a direct write to this register. Every three bits of this register control the latency of each of the (up to) four ranks. R0DGSL controls the latency of rank 0, R1DGSL controls rank 1, and so on. Valid values are 0 to 7.
8-6	R2DGSL	RW	0h	See description for R3DGSL.
5-3	R1DGSL	RW	0h	See description for R3DGSL.
2-0	R0DGSL	RW	0h	See description for R3DGSL.

Table 7-365. Register Call Summary for DDR_PHY_DX0GTR

EMIF Registers
• EMIF Registers: [0]

7.2.5.79 DDR_PHY_DX1GCR Register (Offset = 200h) [reset = 7C000E81h]

This register is used for miscellaneous configurations specific to a particular instantiation of the DATX8 macro.

Table 7-366. DDR_PHY_DX1GCR Instances

Instance	Physical Address
DDR_PHY	0232 9200h

Figure 7-125. DDR_PHY_DX1GCR Register

31	30	29	28	27	26	25	24
CALBYP	MDLEN	WLRNKEN				RESERVED	
RW-0h	RW-1h	RW-Fh				R-0h	
23	22	21	20	19	18	17	16
RESERVED				PLLBYP	GSHIFT	PLLPD	PLLRST
R-0h				RW-0h	RW-0h	RW-0h	RW-0h
15	14	13	12	11	10	9	8
DXOEO		RTTOAL	RTTOH		DQRTT	DQSRTT	DSEN
RW-0h		RW-0h	RW-1h		RW-1h	RW-1h	RW-1h
7	6	5	4	3	2	1	0
DSEN	DQSRPD	DXPDR	DXPDD	DXIOM	DQODT	DQSODT	DXEN
RW-1h	RW-0h	RW-0h	RW-0h	RW-0h	RW-0h	RW-0h	RW-1h

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-367. DDR_PHY_DX1GCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CALBYP	RW	0h	Calibration Bypass: Prevents, if set, period measurement calibration from automatically triggering after PHY initialization.
30	MDLEN	RW	1h	Master Delay Line Enable: Enables, if set, the DATX8 master delay line calibration to perform subsequent period measurements following the initial period measurements that are performed after reset or when calibration is manually triggered. These additional measurements are accumulated and filtered as long as this bit remains high. This bit is ANDed with the common DATX8 MDL enable bit.
29-26	WLRNKEN	RW	Fh	Write Level Rank Enable: Specifies the ranks that should be write leveled for this byte. Write leveling responses from ranks that are not enabled for write leveling for a particular byte are ignored and write leveling is flagged as done for these ranks. WLRKEN[0] enables rank 0, [1] enables rank 1, [2] enables rank 2, and [3] enables rank 3.
25-20	RESERVED	R	0h	Reads return zeros
19	PLLBYP	RW	0h	PLL Bypass: Puts the byte PLL in bypass mode by driving the PLL bypass pin. This bit is not self-clearing and a '0' must be written to de-assert the bypass. This bit is ORed with the global BYP configuration bit in DDR_PHY_PLLCR
18	GSHIFT	RW	0h	Gear Shift: Enables, if set, rapid locking mode on the byte PLL
17	PLLPD	RW	0h	PLL Power Down: Puts the byte PLL in power down mode by driving the PLL power down pin. This bit is not self-clearing and a '0' must be written to de-assert the power-down. This bit is ORed with the global PLLPD configuration bit in DDR_PHY_PLLCR
16	PLLRST	RW	0h	PLL Rest: Resets the byte PLL by driving the PLL reset pin. This bit is not selfclearing and a '0' must be written to de-assert the reset. This bit is ORed with the global PLLRST configuration bit in DDR_PHY_PLLCR

Table 7-367. DDR_PHY_DX1GCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-14	DXOEO	RW	0h	Data Byte Output Enable Override: Specifies whether the output I/O output enable for the byte lane should be set to a fixed value. Valid values are: • 00 = No override. Output enable is controlled by DFI transactions • 01 = Ouput enable is asserted (I/O is forced to output mode). • 10 = Output enable is de-asserted (I/O is forced to input mode) • 11 = Reserved
13	RTTOAL	RW	0h	RTT On Additive Latency: Indicates when the ODT control of DQ/DQS SSTL I/Os is set to the value in DQODT/DQSODT during read cycles. Valid values are: • 0 = ODT control is set to DQSODT/DQODT almost two cycles before read data preamble • 1 = ODT control is set to DQSODT/DQODT almost one cycle before read data preamble
12-11	RTTOH	RW	1h	RTT Output Hold: Indicates the number of clock cycles (from 0 to 3) after the read data postamble for which ODT control should remain set to DQSODT for DQS or DQODT for DQ/DM before disabling it (setting it to '0') when using dynamic ODT control. ODT is disabled almost RTTOH clock cycles after the read postamble
10	DQRTT	RW	1h	DQ Dynamic RTT Control: Indicates, if set, that the ODT control of DQ/DM SSTL I/Os be dynamically controlled by setting it to the value in DQODT during reads and disabling it (setting it to '0') during any other cycle. If this bit is not set, then the ODT control of DQ SSTL I/Os is always set to the value in DQODT.
9	DQSRTT	RW	1h	DQS Dynamic RTT Control: Indicates, if set, that the ODT control of DQS SSTL I/Os be dynamically controlled by setting it to the value in DQSODT during reads and disabling it (setting it to '0') during any other cycle. If this bit is not set, then the ODT control of DQS SSTL I/Os is always set to the value in DQSODT field
8-7	DSEN	RW	1h	Write DQS Enable: Controls whether the write DQS going to the SDRAM is enabled (toggling) or disabled (static value) and whether the DQS is inverted. DQS# is always the inversion of DQS. These values are valid only when DQS/DQS# output enable is on, otherwise the DQS/DQS# is tri-stated. Valid settings are: • 00 = DQS disabled (Driven to constant 0) • 01 = DQS toggling with normal polarity (This should be the default setting) • 10 = DQS toggling with inverted polarity • 11 = DQS disabled (Driven to constant 1)
6	DQSRPD	RW	0h	DQSR Power Down: Powers down, if set, the PDQSR cell.
5	DXPDR	RW	0h	Data Power Down Receiver: Powers down, when set, the input receiver on I/O for DQ, DM, and DQS/DQS# pins of the byte.
4	DXPDD	RW	0h	Data Power Down Driver: Powers down, when set, the output driver on I/O for DQ, DM, and DQS/DQS# pins of the byte.
3	DXIOM	RW	0h	Data I/O Mode: Selects SSTL mode (when set to 0) or CMOS mode (when set to 1) of the I/O for DQ, DM, and DQS/DQS# pins of the byte.
2	DQODT	RW	0h	Data On-Die Termination: Enables, when set, the on-die termination on the I/O for DQ and DM pins of the byte.
1	DQSODT	RW	0h	DQS On-Die Termination: Enables, when set, the on-die termination on the I/O for DQS/DQS# pin of the byte

Table 7-367. DDR_PHY_DX1GCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	DXEN	RW	1h	Data Byte Enable: Enables if set the data byte. Setting this bit to '0' disables the byte, that is, the byte is not used in PHY initialization or training and is ignored during SDRAM read/write operations. Note: Software-initiated PHY initialization and leveling is only required after ECC has been enabled in the PHY by writing to DDR_PHY_DX8GCR.DXEN. Once ECC is enabled in the PHY and the PHY is initialized and leveled, the ECC enable bit in the controller (ECCCTL.ECC_EN) can be enabled or disabled without re-triggering PHY initialization and leveling.

Table 7-368. Register Call Summary for DDR_PHY_DX1GCR

EMIF Registers
- EMIF Registers: [0]

7.2.5.80 DDR_PHY_DX1GSR0 Register (Offset = 204h) [reset = 0h]

DDR_PHY_DX1GSR0 and DDR_PHY_DX1GSR2 are general status registers for the DATX8. They indicate among other things whether write leveling or period measurement calibration is done. Note that WDQCAL, RDQSCAL, RDQSNCAL, GDQSCAL, and WLCAL are calibration measurement-done flags that should be used for debug purposes if the global calibration done flag (DDR_PHY_PGSR0[CAL]) is not asserted. These flags will typically assert for a minimum of two VBUSP_CLK cycles and then de-assert when all measurement done flags are asserted.

Table 7-369. DDR_PHY_DX1GSR0 Instances

Instance	Physical Address
DDR_PHY	0232 9204h

Figure 7-126. DDR_PHY_DX1GSR0 Register

31	30	29	28	27	26	25	24
RESERVED			WLDQ	QSGERR			
R-0h			R-0h	R-0h			
23	22	21	20	19	18	17	16
GDQSPRD							
R-0h							
15	14	13	12	11	10	9	8
DPLOCK	WLPRD						
R-0h	R-0h						
7	6	5	4	3	2	1	0
WLPRD	WLERR	WLDONE	WLCAL	GDQSCAL	RDQSNCAL	RDQSCAL	WDQCAL
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 7-370. DDR_PHY_DX1GSR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reads return zeros.
28	WLDQ	R	0h	Write Leveling DQ Status: Captures the write leveling DQ status from the DRAM during software write leveling
27-24	QSGERR	R	0h	DQS Gate Training Error: Indicates if set that there is an error in DQS gate training. One bit for each of the up to 4 ranks
23-16	GDQSPRD	R	0h	Read DQS gating Period: Returns the DDR clock period measured by the read DQS gating LCDL during calibration. This value is PVT compensated
15	DPLOCK	R	0h	DATX8 PLL Lock: Indicates, if set, that the DATX8 PLL has locked.
14-7	WLPRD	R	0h	Write Leveling Period: Returns the DDR clock period measured by the write leveling LCDL during calibration. The measured period is used to generate the control of the write leveling pipeline which is a function of the write-leveling delay and the clock period. This value is PVT compensated
6	WLERR	R	0h	Write Leveling Error: Indicates, if set, that there is a write leveling error in the DATX8.
5	WLDONE	R	0h	Write Leveling Done: Indicates, if set, that the DATX8 has completed write leveling.
4	WLCAL	R	0h	Write Leveling Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the write leveling slave delay line
3	GDQSCAL	R	0h	Read DQS gating Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the read DQS gating LCDL.

Table 7-370. DDR_PHY_DX1GSR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	RDQSNCAL	R	0h	Read DQS# Calibration (Type B/B1 PHY Only): Indicates, if set, that the DATX8 has finished doing period measurement calibration for the read DQS# LCDL
1	RDQSCAL	R	0h	Read DQS Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the read DQS LCDL.
0	WDQCAL	R	0h	Write DQ Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the write DQ LCDL.

Table 7-371. Register Call Summary for DDR_PHY_DX1GSR0

EMIF Registers

- DDR_PHY_DX1GSR0 Register (Offset = 204h) [reset = 0h]: [0]
- EMIF Registers: [0]
- DDR_PHY_DX1GSR2 Register (Offset = 234h) [reset = 0h]: [0]

7.2.5.81 DDR_PHY_DX1GSR2 Register (Offset = 234h) [reset = 0h]

DDR_PHY_DX1GSR0 and DDR_PHY_DX1GSR2 are general status registers for the DATX8. They indicate among other things whether write leveling or period measurement calibration is done. Note that WDQCAL, RDQSCAL, RDQSNCAL, GDQSCAL, and WLCAL are calibration measurement-done flags that should be used for debug purposes if the global calibration done flag (DDR_PHY_PGSR0[CAL]) is not asserted. These flags will typically assert for a minimum of two VBUSP_CLK cycles and then de-assert when all measurement done flags are asserted.

Table 7-372. DDR_PHY_DX1GSR2 Instances

Instance	Physical Address
DDR_PHY	0232 9234h

Figure 7-127. DDR_PHY_DX1GSR2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				ESTAT			
R-0h				R-0h			
7	6	5	4	3	2	1	0
WEWN	WEERR	REWN	REERR	WDWN	WDERR	RDWN	RDERR
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 7-373. DDR_PHY_DX1GSR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reads return zeros.
11-8	ESTAT	R	0h	Error Status: If an error occurred for this lane as indicated by RDERR, WDERR, REERR or WEERR the error status code can provide additional information regard when the error occurred during the algorithm execution.
7	WEWN	R	0h	Write Eye Centering Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the write eye centering training.
6	WEERR	R	0h	Write Eye Centering Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the write eye centering training.
5	REWN	R	0h	Read Eye Centering Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the read eye centering training.
4	REERR	R	0h	Read Eye Centering Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the read eye centering training.
3	WDWN	R	0h	Write Bit Deskew Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the write bit deskew training.
2	WDERR	R	0h	Write Bit Deskew Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the write bit deskew training.

Table 7-373. DDR_PHY_DX1GSR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	RDWN	R	0h	Read Bit Deskew Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the read bit deskew training.
0	RDERR	R	0h	Read Bit Deskew Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the read bit deskew training.

Table 7-374. Register Call Summary for DDR_PHY_DX1GSR2

EMIF Registers

- DDR_PHY_DX1GSR0 Register (Offset = 204h) [reset = 0h]: [0]
- EMIF Registers: [0]
- DDR_PHY_DX1GSR2 Register (Offset = 234h) [reset = 0h]: [0]

7.2.5.82 DDR_PHY_DX1LCDLR0 Register (Offset = 220h) [reset = 0h]

The DATX8 local calibrated delay line registers 0-2 are used to select the delay value on the LCDLs used in the DATX8 macros. A single LCDL field in the LCDLR register connects to a corresponding LCDL. The following tables describe the bits of the DATX8 LCDLR registers. The write data delay (WDQD) and the read DQS delay (RDQSD) are automatically derived from the measured period during calibration. WDQD and RDQSD correspond to a 90 degrees phase shift for DQ during writes and DQS during reads, respectively. The 90 degrees phase shift is used to centre DQS into the write and read data eyes. A 90 degrees phase shift is equivalent to half the DDR clock period. After calibration WDQD and RDQSD fields will contain a value that corresponds to half the DDR clock period (or a quarter of the SDRAM clock period). The write leveling delay (RnWLD) and read DQS gating delay DQSGD are derived from the write leveling and DQS training algorithms. These are normally run automatically by the PHY control block or they can be executed in software by the user. After calibration RnWLD field will contain a value that corresponds to the DDR clock period (or half of the SDRAM clock period), while RnDQSGD field will contain a value that corresponds to half the DDR clock period (or a quarter of the SDRAM clock period).

After the initial setting, all LCDL register fields are automatically updated by the VT compensation logic to track drifts due to voltage and temperature. The user can however override these values by writing to the respective fields of the register and/or optionally disabling the automatic drift compensation.

Table 7-375. DDR_PHY_DX1LCDLR0 Instances

Instance	Physical Address
DDR_PHY	0232 9220h

Figure 7-128. DDR_PHY_DX1LCDLR0 Register

31	30	29	28	27	26	25	24
R3WLD							
RW-0h							
23	22	21	20	19	18	17	16
R2WLD							
RW-0h							
15	14	13	12	11	10	9	8
R1WLD							
RW-0h							
7	6	5	4	3	2	1	0
R0WLD							
RW-0h							

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-376. DDR_PHY_DX1LCDLR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	R3WLD	RW	0h	Rank 3 Write Leveling Delay: Rank 3 delay select for the write leveling (WL) LCDL
23-16	R2WLD	RW	0h	Rank 2 Write Leveling Delay: Rank 2 delay select for the write leveling (WL) LCDL
15-8	R1WLD	RW	0h	Rank 1 Write Leveling Delay: Rank 1 delay select for the write leveling (WL) LCDL
7-0	R0WLD	RW	0h	Rank 0 Write Leveling Delay: Rank 0 delay select for the write leveling (WL) LCDL

Table 7-377. Register Call Summary for DDR_PHY_DX1LCDLR0

EMIF Registers
- EMIF Registers: [0]

7.2.5.83 DDR_PHY_DX1LCDLR1 Register (Offset = 224h) [reset = 0h]

The DDR_PHY_DX1LCDLR1 register is described in the figure and table below.

Table 7-378. DDR_PHY_DX1LCDLR1 Instances

Instance	Physical Address
DDR_PHY	0232 9224h

Figure 7-129. DDR_PHY_DX1LCDLR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RDQSND							
RW-0h							
15	14	13	12	11	10	9	8
RDQSD							
RW-0h							
7	6	5	4	3	2	1	0
WDQD							
RW-0h							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-379. DDR_PHY_DX1LCDLR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return zeros
23-16	RDQSND	RW	0h	Read DQSN Delay: Delay select for the read DQSN (RDQS) LCDL
15-8	RDQSD	RW	0h	Read DQS Delay: Delay select for the read DQS (RDQS) LCDL
7-0	WDQD	RW	0h	Write Data Delay: Delay select for the write data (WDQ) LCDL.

Table 7-380. Register Call Summary for DDR_PHY_DX1LCDLR1

EMIF Registers
- DDR_PHY_DX1LCDLR1 Register (Offset = 224h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.84 DDR_PHY_DX1LCDLR2 Register (Offset = 228h) [reset = 0h]

The DDR_PHY_DX1LCDLR2 register is described in the figure and table below.

Table 7-381. DDR_PHY_DX1LCDLR2 Instances

Instance	Physical Address
DDR_PHY	0232 9228h

Figure 7-130. DDR_PHY_DX1LCDLR2 Register

31	30	29	28	27	26	25	24
R3DQSGD							
RW-0h							
23	22	21	20	19	18	17	16
R2DQSGD							
RW-0h							
15	14	13	12	11	10	9	8
R1DQSGD							
RW-0h							
7	6	5	4	3	2	1	0
R0DQSGD							
RW-0h							

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-382. DDR_PHY_DX1LCDLR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	R3DQSGD	RW	0h	Rank 3 Read DQS Gating Delay: Rank 3 delay select for the read DQS gating (DQSG) LCDL.
23-16	R2DQSGD	RW	0h	Rank 2 Read DQS Gating Delay: Rank 2 delay select for the read DQS gating (DQSG) LCDL.
15-8	R1DQSGD	RW	0h	Rank 1 Read DQS Gating Delay: Rank 1 delay select for the read DQS gating (DQSG) LCDL.
7-0	R0DQSGD	RW	0h	Rank 0 Read DQS Gating Delay: Rank 0 delay select for the read DQS gating (DQSG) LCDL.

Table 7-383. Register Call Summary for DDR_PHY_DX1LCDLR2

EMIF Registers
- DDR_PHY_DX1LCDLR2 Register (Offset = 228h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.85 DDR_PHY_DX1MDLR Register (Offset = 22Ch) [reset = 0h]

The DATX8 Master Delay Line Registers return different period values measured by the master delay lines in the DATX8 macros. In the default normal operating conditions, the value of the DDR_PHY_DX1MDLR registers change based on ongoing calibration and measurements.

Table 7-384. DDR_PHY_DX1MDLR Instances

Instance	Physical Address
DDR_PHY	0232 922Ch

Figure 7-131. DDR_PHY_DX1MDLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
MDLD							
RW-0h							
15	14	13	12	11	10	9	8
TPRD							
RW-0h							
7	6	5	4	3	2	1	0
IPRD							
RW-0h							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-385. DDR_PHY_DX1MDLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return zeros
23-16	MDLD	RW	0h	MDL Delay: Delay select for the LCDL for the Master Delay Line.
15-8	TPRD	RW	0h	Target Period: Target period measured by the master delay line calibration for VT drift compensation. This is the current measured value of the period and is continuously updated if the MDL is enabled to do so.
7-0	IPRD	RW	0h	Initial Period: Initial period measured by the master delay line calibration for VT drift compensation. This value is used as the denominator when calculating the ratios of updates during VT compensation.

Table 7-386. Register Call Summary for DDR_PHY_DX1MDLR

EMIF Registers
- DDR_PHY_DX1MDLR Register (Offset = 22Ch) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.86 DDR_PHY_DX1GTR Register (Offset = 230h) [reset = 55000h]

This register is used to control various timing settings of the byte lane, including DQS gating system latency and write leveling system latency.

Table 7-387. DDR_PHY_DX1GTR Instances

Instance	Physical Address
DDR_PHY	0232 9230h

Figure 7-132. DDR_PHY_DX1GTR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				R3WLSL		R2WLSL	
R-0h				RW-1h		RW-1h	
15	14	13	12	11	10	9	8
R1WLSL		R0WLSL		R3DGSL			R2DGSL
RW-1h		RW-1h		RW-0h			RW-0h
7	6	5	4	3	2	1	0
R2DGSL		R1DGSL		R0DGSL			
RW-0h		RW-0h		RW-0h			

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-388. DDR_PHY_DX1GTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reads return zeros
19-18	R3WLSL	RW	1h	Rank n Write Leveling System Latency: This is used to adjust the write latency after write leveling. Power-up default is 01 (that is, no extra clock cycles required). The SL fields are initially set by the PUB during automatic write leveling but these values can be overwritten by a direct write to this register. Every two bits of this register control the latency of each of the (up to) four ranks. R0WLSL controls the latency of rank 0, R1WLSL controls rank 1, and so on. Valid values: • 00 = Write latency = WL - 1 • 01 = Write latency = WL • 10 = Write latency = WL + 1 • 11 = Reserved
17-16	R2WLSL	RW	1h	See description for R3WLSL.
15-14	R1WLSL	RW	1h	See description for R3WLSL.
13-12	R0WLSL	RW	1h	See description for R3WLSL.
11-9	R3DGSL	RW	0h	Rank n DQS Gating System Latency: This is used to increase the number of clock cycles needed to expect valid DDR read data by up to seven extra clock cycles. This is used to compensate for board delays and other system delays. Power-up default is 000 (that is, no extra clock cycles required). The SL fields are initially set by the PHY during automatic DQS data training but these values can be overwritten by a direct write to this register. Every three bits of this register control the latency of each of the (up to) four ranks. R0DGSL controls the latency of rank 0, R1DGSL controls rank 1, and so on. Valid values are 0 to 7.
8-6	R2DGSL	RW	0h	See description for R3DGSL.
5-3	R1DGSL	RW	0h	See description for R3DGSL.
2-0	R0DGSL	RW	0h	See description for R3DGSL.

Table 7-389. Register Call Summary for DDR_PHY_DX1GTR

EMIF Registers
• EMIF Registers: [0]

7.2.5.87 DDR_PHY_DX2GCR Register (Offset = 240h) [reset = 7C000E81h]

This register is used for miscellaneous configurations specific to a particular instantiation of the DATX8 macro.

Table 7-390. DDR_PHY_DX2GCR Instances

Instance	Physical Address
DDR_PHY	0232 9240h

Figure 7-133. DDR_PHY_DX2GCR Register

31	30	29	28	27	26	25	24
CALBYP	MDLEN	WLRNKEN				RESERVED	
RW-0h	RW-1h	RW-Fh				R-0h	
23	22	21	20	19	18	17	16
RESERVED				PLLBYP	GSHIFT	PLLPD	PLLRST
R-0h				RW-0h	RW-0h	RW-0h	RW-0h
15	14	13	12	11	10	9	8
DXOEO		RTTOAL	RTTOH		DQRTT	DQSRTT	DSEN
RW-0h		RW-0h	RW-1h		RW-1h	RW-1h	RW-1h
7	6	5	4	3	2	1	0
DSEN	DQSRPD	DXPDR	DXPDD	DXIOM	DQODT	DQSODT	DXEN
RW-1h	RW-0h	RW-0h	RW-0h	RW-0h	RW-0h	RW-0h	RW-1h

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-391. DDR_PHY_DX2GCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CALBYP	RW	0h	Calibration Bypass: Prevents, if set, period measurement calibration from automatically triggering after PHY initialization.
30	MDLEN	RW	1h	Master Delay Line Enable: Enables, if set, the DATX8 master delay line calibration to perform subsequent period measurements following the initial period measurements that are performed after reset or when calibration is manually triggered. These additional measurements are accumulated and filtered as long as this bit remains high. This bit is ANDed with the common DATX8 MDL enable bit.
29-26	WLRNKEN	RW	Fh	Write Level Rank Enable: Specifies the ranks that should be write leveled for this byte. Write leveling responses from ranks that are not enabled for write leveling for a particular byte are ignored and write leveling is flagged as done for these ranks. WLRKEN[0] enables rank 0, [1] enables rank 1, [2] enables rank 2, and [3] enables rank 3.
25-20	RESERVED	R	0h	Reads return zeros
19	PLLBYP	RW	0h	PLL Bypass: Puts the byte PLL in bypass mode by driving the PLL bypass pin. This bit is not self-clearing and a '0' must be written to de-assert the bypass. This bit is ORed with the global BYP configuration bit in DDR_PHY_PLLCR
18	GSHIFT	RW	0h	Gear Shift: Enables, if set, rapid locking mode on the byte PLL
17	PLLPD	RW	0h	PLL Power Down: Puts the byte PLL in power down mode by driving the PLL power down pin. This bit is not self-clearing and a '0' must be written to de-assert the power-down. This bit is ORed with the global PLLPD configuration bit in DDR_PHY_PLLCR
16	PLLRST	RW	0h	PLL Rest: Resets the byte PLL by driving the PLL reset pin. This bit is not selfclearing and a '0' must be written to de-assert the reset. This bit is ORed with the global PLLRST configuration bit in DDR_PHY_PLLCR

Table 7-391. DDR_PHY_DX2GCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-14	DXOEO	RW	0h	Data Byte Output Enable Override: Specifies whether the output I/O output enable for the byte lane should be set to a fixed value. Valid values are: • 00 = No override. Output enable is controlled by DFI transactions • 01 = Ouput enable is asserted (I/O is forced to output mode). • 10 = Output enable is de-asserted (I/O is forced to input mode) • 11 = Reserved
13	RTTOAL	RW	0h	RTT On Additive Latency: Indicates when the ODT control of DQ/DQS SSTL I/Os is set to the value in DQODT/DQSODT during read cycles. Valid values are: • 0 = ODT control is set to DQSODT/DQODT almost two cycles before read data preamble • 1 = ODT control is set to DQSODT/DQODT almost one cycle before read data preamble
12-11	RTTOH	RW	1h	RTT Output Hold: Indicates the number of clock cycles (from 0 to 3) after the read data postamble for which ODT control should remain set to DQSODT for DQS or DQODT for DQ/DM before disabling it (setting it to '0') when using dynamic ODT control. ODT is disabled almost RTTOH clock cycles after the read postamble
10	DQRTT	RW	1h	DQ Dynamic RTT Control: Indicates, if set, that the ODT control of DQ/DM SSTL I/Os be dynamically controlled by setting it to the value in DQODT during reads and disabling it (setting it to '0') during any other cycle. If this bit is not set, then the ODT control of DQ SSTL I/Os is always set to the value in DQODT.
9	DQSRTT	RW	1h	DQS Dynamic RTT Control: Indicates, if set, that the ODT control of DQS SSTL I/Os be dynamically controlled by setting it to the value in DQSODT during reads and disabling it (setting it to '0') during any other cycle. If this bit is not set, then the ODT control of DQS SSTL I/Os is always set to the value in DQSODT field
8-7	DSEN	RW	1h	Write DQS Enable: Controls whether the write DQS going to the SDRAM is enabled (toggling) or disabled (static value) and whether the DQS is inverted. DQS# is always the inversion of DQS. These values are valid only when DQS/DQS# output enable is on, otherwise the DQS/DQS# is tri-stated. Valid settings are: • 00 = DQS disabled (Driven to constant 0) • 01 = DQS toggling with normal polarity (This should be the default setting) • 10 = DQS toggling with inverted polarity • 11 = DQS disabled (Driven to constant 1)
6	DQSRPD	RW	0h	DQSR Power Down: Powers down, if set, the PDQSR cell.
5	DXPDR	RW	0h	Data Power Down Receiver: Powers down, when set, the input receiver on I/O for DQ, DM, and DQS/DQS# pins of the byte.
4	DXPDD	RW	0h	Data Power Down Driver: Powers down, when set, the output driver on I/O for DQ, DM, and DQS/DQS# pins of the byte.
3	DXIOM	RW	0h	Data I/O Mode: Selects SSTL mode (when set to 0) or CMOS mode (when set to 1) of the I/O for DQ, DM, and DQS/DQS# pins of the byte.
2	DQODT	RW	0h	Data On-Die Termination: Enables, when set, the on-die termination on the I/O for DQ and DM pins of the byte.
1	DQSODT	RW	0h	DQS On-Die Termination: Enables, when set, the on-die termination on the I/O for DQS/DQS# pin of the byte

Table 7-391. DDR_PHY_DX2GCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	DXEN	RW	1h	Data Byte Enable: Enables if set the data byte. Setting this bit to '0' disables the byte, that is, the byte is not used in PHY initialization or training and is ignored during SDRAM read/write operations. Note: Software-initiated PHY initialization and leveling is only required after ECC has been enabled in the PHY by writing to DDR_PHY_DX8GCR.DXEN. Once ECC is enabled in the PHY and the PHY is initialized and leveled, the ECC enable bit in the controller (ECCCTL.ECC_EN) can be enabled or disabled without re-triggering PHY initialization and leveling.

Table 7-392. Register Call Summary for DDR_PHY_DX2GCR

EMIF Registers
- EMIF Registers: [0]

7.2.5.88 DDR_PHY_DX2GSR0 Register (Offset = 244h) [reset = 0h]

DDR_PHY_DX2GSR0 and DDR_PHY_DX2GSR2 are general status registers for the DATX8. They indicate among other things whether write leveling or period measurement calibration is done. Note that WDQCAL, RDQSCAL, RDQSNCAL, GDQSCAL, and WLCAL are calibration measurement-done flags that should be used for debug purposes if the global calibration done flag (DDR_PHY_PGSR0[CAL]) is not asserted. These flags will typically assert for a minimum of two VBUSP_CLK cycles and then de-assert when all measurement done flags are asserted.

Table 7-393. DDR_PHY_DX2GSR0 Instances

Instance	Physical Address
DDR_PHY	0232 9244h

Figure 7-134. DDR_PHY_DX2GSR0 Register

31	30	29	28	27	26	25	24
RESERVED			WLDQ	QSGERR			
R-0h			R-0h	R-0h			
23	22	21	20	19	18	17	16
GDQSPRD							
R-0h							
15	14	13	12	11	10	9	8
DPLOCK	WLPRD						
R-0h	R-0h						
7	6	5	4	3	2	1	0
WLPRD	WLERR	WLDONE	WLCAL	GDQSCAL	RDQSNCAL	RDQSCAL	WDQCAL
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 7-394. DDR_PHY_DX2GSR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reads return zeros.
28	WLDQ	R	0h	Write Leveling DQ Status: Captures the write leveling DQ status from the DRAM during software write leveling
27-24	QSGERR	R	0h	DQS Gate Training Error: Indicates if set that there is an error in DQS gate training. One bit for each of the up to 4 ranks
23-16	GDQSPRD	R	0h	Read DQS gating Period: Returns the DDR clock period measured by the read DQS gating LCDL during calibration. This value is PVT compensated
15	DPLOCK	R	0h	DATX8 PLL Lock: Indicates, if set, that the DATX8 PLL has locked.
14-7	WLPRD	R	0h	Write Leveling Period: Returns the DDR clock period measured by the write leveling LCDL during calibration. The measured period is used to generate the control of the write leveling pipeline which is a function of the write-leveling delay and the clock period. This value is PVT compensated
6	WLERR	R	0h	Write Leveling Error: Indicates, if set, that there is a write leveling error in the DATX8.
5	WLDONE	R	0h	Write Leveling Done: Indicates, if set, that the DATX8 has completed write leveling.
4	WLCAL	R	0h	Write Leveling Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the write leveling slave delay line
3	GDQSCAL	R	0h	Read DQS gating Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the read DQS gating LCDL.

Table 7-394. DDR_PHY_DX2GSR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	RDQSNCAL	R	0h	Read DQS# Calibration (Type B/B1 PHY Only): Indicates, if set, that the DATX8 has finished doing period measurement calibration for the read DQS# LCDL
1	RDQSCAL	R	0h	Read DQS Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the read DQS LCDL.
0	WDQCAL	R	0h	Write DQ Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the write DQ LCDL.

Table 7-395. Register Call Summary for DDR_PHY_DX2GSR0

EMIF Registers
- DDR_PHY_DX2GSR2 Register (Offset = 274h) [reset = 0h]: [0]
- DDR_PHY_DX2GSR0 Register (Offset = 244h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.89 DDR_PHY_DX2GSR2 Register (Offset = 274h) [reset = 0h]

DDR_PHY_DX2GSR0 and DDR_PHY_DX2GSR2 are general status registers for the DATX8. They indicate among other things whether write leveling or period measurement calibration is done. Note that WDQCAL, RDQSCAL, RDQSNCAL, GDQSCAL, and WLCAL are calibration measurement-done flags that should be used for debug purposes if the global calibration done flag (DDR_PHY_PGSR0[CAL]) is not asserted. These flags will typically assert for a minimum of two VBUSP_CLK cycles and then de-assert when all measurement done flags are asserted.

Table 7-396. DDR_PHY_DX2GSR2 Instances

Instance	Physical Address
DDR_PHY	0232 9274h

Figure 7-135. DDR_PHY_DX2GSR2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				ESTAT			
R-0h				R-0h			
7	6	5	4	3	2	1	0
WEWN	WEERR	REWN	REERR	WDWN	WDERR	RDWN	RDERR
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 7-397. DDR_PHY_DX2GSR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reads return zeros.
11-8	ESTAT	R	0h	Error Status: If an error occurred for this lane as indicated by RDERR, WDERR, REERR or WEERR the error status code can provide additional information regard when the error occurred during the algorithm execution.
7	WEWN	R	0h	Write Eye Centering Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the write eye centering training.
6	WEERR	R	0h	Write Eye Centering Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the write eye centering training.
5	REWN	R	0h	Read Eye Centering Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the read eye centering training.
4	REERR	R	0h	Read Eye Centering Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the read eye centering training.
3	WDWN	R	0h	Write Bit Deskew Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the write bit deskew training.
2	WDERR	R	0h	Write Bit Deskew Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the write bit deskew training.

Table 7-397. DDR_PHY_DX2GSR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	RDWN	R	0h	Read Bit Deskew Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the read bit deskew training.
0	RDERR	R	0h	Read Bit Deskew Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the read bit deskew training.

Table 7-398. Register Call Summary for DDR_PHY_DX2GSR2

EMIF Registers

- DDR_PHY_DX2GSR2 Register (Offset = 274h) [reset = 0h]: [0]
- DDR_PHY_DX2GSR0 Register (Offset = 244h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.90 DDR_PHY_DX2LCDLR0 Register (Offset = 260h) [reset = 0h]

The DATX8 local calibrated delay line registers 0-2 are used to select the delay value on the LCDLs used in the DATX8 macros. A single LCDL field in the LCDLR register connects to a corresponding LCDL. The following tables describe the bits of the DATX8 LCDLR registers. The write data delay (WDQD) and the read DQS delay (RDQSD) are automatically derived from the measured period during calibration. WDQD and RDQSD correspond to a 90 degrees phase shift for DQ during writes and DQS during reads, respectively. The 90 degrees phase shift is used to centre DQS into the write and read data eyes. A 90 degrees phase shift is equivalent to half the DDR clock period. After calibration WDQD and RDQSD fields will contain a value that corresponds to half the DDR clock period (or a quarter of the SDRAM clock period). The write leveling delay (RnWLD) and read DQS gating delay DQSGD are derived from the write leveling and DQS training algorithms. These are normally run automatically by the PHY control block or they can be executed in software by the user. After calibration RnWLD field will contain a value that corresponds to the DDR clock period (or half of the SDRAM clock period), while RnDQSGD field will contain a value that corresponds to half the DDR clock period (or a quarter of the SDRAM clock period).

After the initial setting, all LCDL register fields are automatically updated by the VT compensation logic to track drifts due to voltage and temperature. The user can however override these values by writing to the respective fields of the register and/or optionally disabling the automatic drift compensation.

Table 7-399. DDR_PHY_DX2LCDLR0 Instances

Instance	Physical Address
DDR_PHY	0232 9260h

Figure 7-136. DDR_PHY_DX2LCDLR0 Register

31	30	29	28	27	26	25	24
R3WLD							
RW-0h							
23	22	21	20	19	18	17	16
R2WLD							
RW-0h							
15	14	13	12	11	10	9	8
R1WLD							
RW-0h							
7	6	5	4	3	2	1	0
R0WLD							
RW-0h							

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-400. DDR_PHY_DX2LCDLR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	R3WLD	RW	0h	Rank 3 Write Leveling Delay: Rank 3 delay select for the write leveling (WL) LCDL
23-16	R2WLD	RW	0h	Rank 2 Write Leveling Delay: Rank 2 delay select for the write leveling (WL) LCDL
15-8	R1WLD	RW	0h	Rank 1 Write Leveling Delay: Rank 1 delay select for the write leveling (WL) LCDL
7-0	R0WLD	RW	0h	Rank 0 Write Leveling Delay: Rank 0 delay select for the write leveling (WL) LCDL

Table 7-401. Register Call Summary for DDR_PHY_DX2LCDLR0

EMIF Registers
- EMIF Registers: [0]

7.2.5.91 DDR_PHY_DX2LCDLR1 Register (Offset = 264h) [reset = 0h]

The DDR_PHY_DX2LCDLR1 register is described in the figure and table below.

Table 7-402. DDR_PHY_DX2LCDLR1 Instances

Instance	Physical Address
DDR_PHY	0232 9264h

Figure 7-137. DDR_PHY_DX2LCDLR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RDQSND							
RW-0h							
15	14	13	12	11	10	9	8
RDQSD							
RW-0h							
7	6	5	4	3	2	1	0
WDQD							
RW-0h							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-403. DDR_PHY_DX2LCDLR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return zeros
23-16	RDQSND	RW	0h	Read DQSN Delay: Delay select for the read DQSN (RDQS) LCDL
15-8	RDQSD	RW	0h	Read DQS Delay: Delay select for the read DQS (RDQS) LCDL
7-0	WDQD	RW	0h	Write Data Delay: Delay select for the write data (WDQ) LCDL.

Table 7-404. Register Call Summary for DDR_PHY_DX2LCDLR1

EMIF Registers
- DDR_PHY_DX2LCDLR1 Register (Offset = 264h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.92 DDR_PHY_DX2LCDLR2 Register (Offset = 268h) [reset = 0h]

The DDR_PHY_DX2LCDLR2 register is described in the figure and table below.

Table 7-405. DDR_PHY_DX2LCDLR2 Instances

Instance	Physical Address
DDR_PHY	0232 9268h

Figure 7-138. DDR_PHY_DX2LCDLR2 Register

31	30	29	28	27	26	25	24
R3DQSGD							
RW-0h							
23	22	21	20	19	18	17	16
R2DQSGD							
RW-0h							
15	14	13	12	11	10	9	8
R1DQSGD							
RW-0h							
7	6	5	4	3	2	1	0
R0DQSGD							
RW-0h							

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-406. DDR_PHY_DX2LCDLR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	R3DQSGD	RW	0h	Rank 3 Read DQS Gating Delay: Rank 3 delay select for the read DQS gating (DQSG) LCDL.
23-16	R2DQSGD	RW	0h	Rank 2 Read DQS Gating Delay: Rank 2 delay select for the read DQS gating (DQSG) LCDL.
15-8	R1DQSGD	RW	0h	Rank 1 Read DQS Gating Delay: Rank 1 delay select for the read DQS gating (DQSG) LCDL.
7-0	R0DQSGD	RW	0h	Rank 0 Read DQS Gating Delay: Rank 0 delay select for the read DQS gating (DQSG) LCDL.

Table 7-407. Register Call Summary for DDR_PHY_DX2LCDLR2

EMIF Registers
- EMIF Registers: [0]
- DDR_PHY_DX2LCDLR2 Register (Offset = 268h) [reset = 0h]: [0]

7.2.5.93 DDR_PHY_DX2MDLR Register (Offset = 26Ch) [reset = 0h]

The DATX8 Master Delay Line Registers return different period values measured by the master delay lines in the DATX8 macros. In the default normal operating conditions, the value of the DDR_PHY_DX2MDLR registers change based on ongoing calibration and measurements.

Table 7-408. DDR_PHY_DX2MDLR Instances

Instance	Physical Address
DDR_PHY	0232 926Ch

Figure 7-139. DDR_PHY_DX2MDLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
MDLD							
RW-0h							
15	14	13	12	11	10	9	8
TPRD							
RW-0h							
7	6	5	4	3	2	1	0
IPRD							
RW-0h							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-409. DDR_PHY_DX2MDLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return zeros
23-16	MDLD	RW	0h	MDL Delay: Delay select for the LCDL for the Master Delay Line.
15-8	TPRD	RW	0h	Target Period: Target period measured by the master delay line calibration for VT drift compensation. This is the current measured value of the period and is continuously updated if the MDL is enabled to do so.
7-0	IPRD	RW	0h	Initial Period: Initial period measured by the master delay line calibration for VT drift compensation. This value is used as the denominator when calculating the ratios of updates during VT compensation.

Table 7-410. Register Call Summary for DDR_PHY_DX2MDLR

EMIF Registers
- DDR_PHY_DX2MDLR Register (Offset = 26Ch) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.94 DDR_PHY_DX2GTR Register (Offset = 270h) [reset = 55000h]

This register is used to control various timing settings of the byte lane, including DQS gating system latency and write leveling system latency.

Table 7-411. DDR_PHY_DX2GTR Instances

Instance	Physical Address
DDR_PHY	0232 9270h

Figure 7-140. DDR_PHY_DX2GTR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				R3WLSL		R2WLSL	
R-0h				RW-1h		RW-1h	
15	14	13	12	11	10	9	8
R1WLSL		R0WLSL		R3DGSL			R2DGSL
RW-1h		RW-1h		RW-0h			RW-0h
7	6	5	4	3	2	1	0
R2DGSL		R1DGSL		R0DGSL			
RW-0h		RW-0h		RW-0h			

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-412. DDR_PHY_DX2GTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reads return zeros
19-18	R3WLSL	RW	1h	Rank n Write Leveling System Latency: This is used to adjust the write latency after write leveling. Power-up default is 01 (that is, no extra clock cycles required). The SL fields are initially set by the PUB during automatic write leveling but these values can be overwritten by a direct write to this register. Every two bits of this register control the latency of each of the (up to) four ranks. R0WLSL controls the latency of rank 0, R1WLSL controls rank 1, and so on. Valid values: • 00 = Write latency = WL - 1 • 01 = Write latency = WL • 10 = Write latency = WL + 1 • 11 = Reserved
17-16	R2WLSL	RW	1h	See description for R3WLSL.
15-14	R1WLSL	RW	1h	See description for R3WLSL.
13-12	R0WLSL	RW	1h	See description for R3WLSL.
11-9	R3DGSL	RW	0h	Rank n DQS Gating System Latency: This is used to increase the number of clock cycles needed to expect valid DDR read data by up to seven extra clock cycles. This is used to compensate for board delays and other system delays. Power-up default is 000 (that is, no extra clock cycles required). The SL fields are initially set by the PHY during automatic DQS data training but these values can be overwritten by a direct write to this register. Every three bits of this register control the latency of each of the (up to) four ranks. R0DGSL controls the latency of rank 0, R1DGSL controls rank 1, and so on. Valid values are 0 to 7.
8-6	R2DGSL	RW	0h	See description for R3DGSL.
5-3	R1DGSL	RW	0h	See description for R3DGSL.
2-0	R0DGSL	RW	0h	See description for R3DGSL.

Table 7-413. Register Call Summary for DDR_PHY_DX2GTR

EMIF Registers
• EMIF Registers: [0]

7.2.5.95 DDR_PHY_DX3GCR Register (Offset = 280h) [reset = 7C000E81h]

This register is used for miscellaneous configurations specific to a particular instantiation of the DATX8 macro.

Table 7-414. DDR_PHY_DX3GCR Instances

Instance	Physical Address
DDR_PHY	0232 9280h

Figure 7-141. DDR_PHY_DX3GCR Register

31	30	29	28	27	26	25	24
CALBYP	MDLEN	WLRNKEN				RESERVED	
RW-0h	RW-1h	RW-Fh				R-0h	
23	22	21	20	19	18	17	16
RESERVED				PLLBYP	GSHIFT	PLLPD	PLLRST
R-0h				RW-0h	RW-0h	RW-0h	RW-0h
15	14	13	12	11	10	9	8
DXOEO		RTTOAL	RTTOH		DQRTT	DQSRTT	DSEN
RW-0h		RW-0h	RW-1h		RW-1h	RW-1h	RW-1h
7	6	5	4	3	2	1	0
DSEN	DQSRPD	DXPDR	DXPDD	DXIOM	DQODT	DQSODT	DXEN
RW-1h	RW-0h	RW-0h	RW-0h	RW-0h	RW-0h	RW-0h	RW-1h

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-415. DDR_PHY_DX3GCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CALBYP	RW	0h	Calibration Bypass: Prevents, if set, period measurement calibration from automatically triggering after PHY initialization.
30	MDLEN	RW	1h	Master Delay Line Enable: Enables, if set, the DATX8 master delay line calibration to perform subsequent period measurements following the initial period measurements that are performed after reset or when calibration is manually triggered. These additional measurements are accumulated and filtered as long as this bit remains high. This bit is ANDed with the common DATX8 MDL enable bit.
29-26	WLRNKEN	RW	Fh	Write Level Rank Enable: Specifies the ranks that should be write leveled for this byte. Write leveling responses from ranks that are not enabled for write leveling for a particular byte are ignored and write leveling is flagged as done for these ranks. WLRKEN[0] enables rank 0, [1] enables rank 1, [2] enables rank 2, and [3] enables rank 3.
25-20	RESERVED	R	0h	Reads return zeros
19	PLLBYP	RW	0h	PLL Bypass: Puts the byte PLL in bypass mode by driving the PLL bypass pin. This bit is not self-clearing and a '0' must be written to de-assert the bypass. This bit is ORed with the global BYP configuration bit in DDR_PHY_PLLCR
18	GSHIFT	RW	0h	Gear Shift: Enables, if set, rapid locking mode on the byte PLL
17	PLLPD	RW	0h	PLL Power Down: Puts the byte PLL in power down mode by driving the PLL power down pin. This bit is not self-clearing and a '0' must be written to de-assert the power-down. This bit is ORed with the global PLLPD configuration bit in DDR_PHY_PLLCR
16	PLLRST	RW	0h	PLL Rest: Resets the byte PLL by driving the PLL reset pin. This bit is not selfclearing and a '0' must be written to de-assert the reset. This bit is ORed with the global PLLRST configuration bit in DDR_PHY_PLLCR

Table 7-415. DDR_PHY_DX3GCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-14	DXOEO	RW	0h	Data Byte Output Enable Override: Specifies whether the output I/O output enable for the byte lane should be set to a fixed value. Valid values are: • 00 = No override. Output enable is controlled by DFI transactions • 01 = Ouput enable is asserted (I/O is forced to output mode). • 10 = Output enable is de-asserted (I/O is forced to input mode) • 11 = Reserved
13	RTTOAL	RW	0h	RTT On Additive Latency: Indicates when the ODT control of DQ/DQS SSTL I/Os is set to the value in DQODT/DQSODT during read cycles. Valid values are: • 0 = ODT control is set to DQSODT/DQODT almost two cycles before read data preamble • 1 = ODT control is set to DQSODT/DQODT almost one cycle before read data preamble
12-11	RTTOH	RW	1h	RTT Output Hold: Indicates the number of clock cycles (from 0 to 3) after the read data postamble for which ODT control should remain set to DQSODT for DQS or DQODT for DQ/DM before disabling it (setting it to '0') when using dynamic ODT control. ODT is disabled almost RTTOH clock cycles after the read postamble
10	DQRTT	RW	1h	DQ Dynamic RTT Control: Indicates, if set, that the ODT control of DQ/DM SSTL I/Os be dynamically controlled by setting it to the value in DQODT during reads and disabling it (setting it to '0') during any other cycle. If this bit is not set, then the ODT control of DQ SSTL I/Os is always set to the value in DQODT.
9	DQSRTT	RW	1h	DQS Dynamic RTT Control: Indicates, if set, that the ODT control of DQS SSTL I/Os be dynamically controlled by setting it to the value in DQSODT during reads and disabling it (setting it to '0') during any other cycle. If this bit is not set, then the ODT control of DQS SSTL I/Os is always set to the value in DQSODT field
8-7	DSEN	RW	1h	Write DQS Enable: Controls whether the write DQS going to the SDRAM is enabled (toggling) or disabled (static value) and whether the DQS is inverted. DQS# is always the inversion of DQS. These values are valid only when DQS/DQS# output enable is on, otherwise the DQS/DQS# is tri-stated. Valid settings are: • 00 = DQS disabled (Driven to constant 0) • 01 = DQS toggling with normal polarity (This should be the default setting) • 10 = DQS toggling with inverted polarity • 11 = DQS disabled (Driven to constant 1)
6	DQSRPD	RW	0h	DQSR Power Down: Powers down, if set, the PDQSR cell.
5	DXPDR	RW	0h	Data Power Down Receiver: Powers down, when set, the input receiver on I/O for DQ, DM, and DQS/DQS# pins of the byte.
4	DXPDD	RW	0h	Data Power Down Driver: Powers down, when set, the output driver on I/O for DQ, DM, and DQS/DQS# pins of the byte.
3	DXIOM	RW	0h	Data I/O Mode: Selects SSTL mode (when set to 0) or CMOS mode (when set to 1) of the I/O for DQ, DM, and DQS/DQS# pins of the byte.
2	DQODT	RW	0h	Data On-Die Termination: Enables, when set, the on-die termination on the I/O for DQ and DM pins of the byte.
1	DQSODT	RW	0h	DQS On-Die Termination: Enables, when set, the on-die termination on the I/O for DQS/DQS# pin of the byte

Table 7-415. DDR_PHY_DX3GCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	DXEN	RW	1h	Data Byte Enable: Enables if set the data byte. Setting this bit to '0' disables the byte, that is, the byte is not used in PHY initialization or training and is ignored during SDRAM read/write operations. Note: Software-initiated PHY initialization and leveling is only required after ECC has been enabled in the PHY by writing to DDR_PHY_DX8GCR.DXEN. Once ECC is enabled in the PHY and the PHY is initialized and leveled, the ECC enable bit in the controller (ECCCTL.ECC_EN) can be enabled or disabled without re-triggering PHY initialization and leveling.

Table 7-416. Register Call Summary for DDR_PHY_DX3GCR

EMIF Registers
- EMIF Registers: [0]

7.2.5.96 DDR_PHY_DX3GSR0 Register (Offset = 284h) [reset = 0h]

DDR_PHY_DX3GSR0 and DDR_PHY_DX3GSR2 are general status registers for the DATX8. They indicate among other things whether write leveling or period measurement calibration is done. Note that WDQCAL, RDQSCAL, RDQSNCAL, GDQSCAL, and WLCAL are calibration measurement-done flags that should be used for debug purposes if the global calibration done flag (DDR_PHY_PGSR0[CAL]) is not asserted. These flags will typically assert for a minimum of two VBUSP_CLK cycles and then de-assert when all measurement done flags are asserted.

Table 7-417. DDR_PHY_DX3GSR0 Instances

Instance	Physical Address
DDR_PHY	0232 9284h

Figure 7-142. DDR_PHY_DX3GSR0 Register

31	30	29	28	27	26	25	24
RESERVED			WLDQ	QSGERR			
R-0h			R-0h	R-0h			
23	22	21	20	19	18	17	16
GDQSPRD							
R-0h							
15	14	13	12	11	10	9	8
DPLOCK	WLPRD						
R-0h	R-0h						
7	6	5	4	3	2	1	0
WLPRD	WLERR	WLDONE	WLCAL	GDQSCAL	RDQSNCAL	RDQSCAL	WDQCAL
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 7-418. DDR_PHY_DX3GSR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reads return zeros.
28	WLDQ	R	0h	Write Leveling DQ Status: Captures the write leveling DQ status from the DRAM during software write leveling
27-24	QSGERR	R	0h	DQS Gate Training Error: Indicates if set that there is an error in DQS gate training. One bit for each of the up to 4 ranks
23-16	GDQSPRD	R	0h	Read DQS gating Period: Returns the DDR clock period measured by the read DQS gating LCDL during calibration. This value is PVT compensated
15	DPLOCK	R	0h	DATX8 PLL Lock: Indicates, if set, that the DATX8 PLL has locked.
14-7	WLPRD	R	0h	Write Leveling Period: Returns the DDR clock period measured by the write leveling LCDL during calibration. The measured period is used to generate the control of the write leveling pipeline which is a function of the write-leveling delay and the clock period. This value is PVT compensated
6	WLERR	R	0h	Write Leveling Error: Indicates, if set, that there is a write leveling error in the DATX8.
5	WLDONE	R	0h	Write Leveling Done: Indicates, if set, that the DATX8 has completed write leveling.
4	WLCAL	R	0h	Write Leveling Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the write leveling slave delay line
3	GDQSCAL	R	0h	Read DQS gating Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the read DQS gating LCDL.

Table 7-418. DDR_PHY_DX3GSR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	RDQSNCAL	R	0h	Read DQS# Calibration (Type B/B1 PHY Only): Indicates, if set, that the DATX8 has finished doing period measurement calibration for the read DQS# LCDL
1	RDQSCAL	R	0h	Read DQS Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the read DQS LCDL.
0	WDQCAL	R	0h	Write DQ Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the write DQ LCDL.

Table 7-419. Register Call Summary for DDR_PHY_DX3GSR0

EMIF Registers
- DDR_PHY_DX3GSR2 Register (Offset = 2B4h) [reset = 0h]: [0]
- DDR_PHY_DX3GSR0 Register (Offset = 284h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.97 DDR_PHY_DX3GSR2 Register (Offset = 2B4h) [reset = 0h]

DDR_PHY_DX3GSR0 and DDR_PHY_DX3GSR2 are general status registers for the DATX8. They indicate among other things whether write leveling or period measurement calibration is done. Note that WDQCAL, RDQSCAL, RDQSNCAL, GDQSCAL, and WLCAL are calibration measurement-done flags that should be used for debug purposes if the global calibration done flag (DDR_PHY_PGSR0[CAL]) is not asserted. These flags will typically assert for a minimum of two VBUSP_CLK cycles and then de-assert when all measurement done flags are asserted.

Table 7-420. DDR_PHY_DX3GSR2 Instances

Instance	Physical Address
DDR_PHY	0232 92B4h

Figure 7-143. DDR_PHY_DX3GSR2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				ESTAT			
R-0h				R-0h			
7	6	5	4	3	2	1	0
WEWN	WEERR	REWN	REERR	WDWN	WDERR	RDWN	RDERR
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 7-421. DDR_PHY_DX3GSR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reads return zeros.
11-8	ESTAT	R	0h	Error Status: If an error occurred for this lane as indicated by RDERR, WDERR, REERR or WEERR the error status code can provide additional information regard when the error occurred during the algorithm execution.
7	WEWN	R	0h	Write Eye Centering Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the write eye centering training.
6	WEERR	R	0h	Write Eye Centering Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the write eye centering training.
5	REWN	R	0h	Read Eye Centering Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the read eye centering training.
4	REERR	R	0h	Read Eye Centering Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the read eye centering training.
3	WDWN	R	0h	Write Bit Deskew Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the write bit deskew training.
2	WDERR	R	0h	Write Bit Deskew Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the write bit deskew training.

Table 7-421. DDR_PHY_DX3GSR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	RDWN	R	0h	Read Bit Deskew Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the read bit deskew training.
0	RDERR	R	0h	Read Bit Deskew Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the read bit deskew training.

Table 7-422. Register Call Summary for DDR_PHY_DX3GSR2

EMIF Registers

- DDR_PHY_DX3GSR2 Register (Offset = 2B4h) [reset = 0h]: [0]
- DDR_PHY_DX3GSR0 Register (Offset = 284h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.98 DDR_PHY_DX3LCDLR0 Register (Offset = 2A0h) [reset = 0h]

The DATX8 local calibrated delay line registers 0-2 are used to select the delay value on the LCDLs used in the DATX8 macros. A single LCDL field in the LCDLR register connects to a corresponding LCDL. The following tables describe the bits of the DATX8 LCDLR registers. The write data delay (WDQD) and the read DQS delay (RDQSD) are automatically derived from the measured period during calibration. WDQD and RDQSD correspond to a 90 degrees phase shift for DQ during writes and DQS during reads, respectively. The 90 degrees phase shift is used to centre DQS into the write and read data eyes. A 90 degrees phase shift is equivalent to half the DDR clock period. After calibration WDQD and RDQSD fields will contain a value that corresponds to half the DDR clock period (or a quarter of the SDRAM clock period). The write leveling delay (RnWLD) and read DQS gating delay DQSGD are derived from the write leveling and DQS training algorithms. These are normally run automatically by the PHY control block or they can be executed in software by the user. After calibration RnWLD field will contain a value that corresponds to the DDR clock period (or half of the SDRAM clock period), while RnDQSGD field will contain a value that corresponds to half the DDR clock period (or a quarter of the SDRAM clock period).

After the initial setting, all LCDL register fields are automatically updated by the VT compensation logic to track drifts due to voltage and temperature. The user can however override these values by writing to the respective fields of the register and/or optionally disabling the automatic drift compensation.

Table 7-423. DDR_PHY_DX3LCDLR0 Instances

Instance	Physical Address
DDR_PHY	0232 92A0h

Figure 7-144. DDR_PHY_DX3LCDLR0 Register

31	30	29	28	27	26	25	24
R3WLD							
RW-0h							
23	22	21	20	19	18	17	16
R2WLD							
RW-0h							
15	14	13	12	11	10	9	8
R1WLD							
RW-0h							
7	6	5	4	3	2	1	0
R0WLD							
RW-0h							

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-424. DDR_PHY_DX3LCDLR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	R3WLD	RW	0h	Rank 3 Write Leveling Delay: Rank 3 delay select for the write leveling (WL) LCDL
23-16	R2WLD	RW	0h	Rank 2 Write Leveling Delay: Rank 2 delay select for the write leveling (WL) LCDL
15-8	R1WLD	RW	0h	Rank 1 Write Leveling Delay: Rank 1 delay select for the write leveling (WL) LCDL
7-0	R0WLD	RW	0h	Rank 0 Write Leveling Delay: Rank 0 delay select for the write leveling (WL) LCDL

Table 7-425. Register Call Summary for DDR_PHY_DX3LCDLR0

EMIF Registers
- EMIF Registers: [0]

7.2.5.99 DDR_PHY_DX3LCDLR1 Register (Offset = 2A4h) [reset = 0h]

The DDR_PHY_DX3LCDLR1 register is described in the figure and table below.

Table 7-426. DDR_PHY_DX3LCDLR1 Instances

Instance	Physical Address
DDR_PHY	0232 92A4h

Figure 7-145. DDR_PHY_DX3LCDLR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RDQSND							
RW-0h							
15	14	13	12	11	10	9	8
RDQSD							
RW-0h							
7	6	5	4	3	2	1	0
WDQD							
RW-0h							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-427. DDR_PHY_DX3LCDLR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return zeros
23-16	RDQSND	RW	0h	Read DQSN Delay: Delay select for the read DQSN (RDQS) LCDL
15-8	RDQSD	RW	0h	Read DQS Delay: Delay select for the read DQS (RDQS) LCDL
7-0	WDQD	RW	0h	Write Data Delay: Delay select for the write data (WDQ) LCDL.

Table 7-428. Register Call Summary for DDR_PHY_DX3LCDLR1

EMIF Registers
- DDR_PHY_DX3LCDLR1 Register (Offset = 2A4h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.100 DDR_PHY_DX3LCDLR2 Register (Offset = 2A8h) [reset = 0h]

The DDR_PHY_DX3LCDLR2 register is described in the figure and table below.

Table 7-429. DDR_PHY_DX3LCDLR2 Instances

Instance	Physical Address
DDR_PHY	0232 92A8h

Figure 7-146. DDR_PHY_DX3LCDLR2 Register

31	30	29	28	27	26	25	24
R3DQSGD							
RW-0h							
23	22	21	20	19	18	17	16
R2DQSGD							
RW-0h							
15	14	13	12	11	10	9	8
R1DQSGD							
RW-0h							
7	6	5	4	3	2	1	0
R0DQSGD							
RW-0h							

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-430. DDR_PHY_DX3LCDLR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	R3DQSGD	RW	0h	Rank 3 Read DQS Gating Delay: Rank 3 delay select for the read DQS gating (DQSG) LCDL.
23-16	R2DQSGD	RW	0h	Rank 2 Read DQS Gating Delay: Rank 2 delay select for the read DQS gating (DQSG) LCDL.
15-8	R1DQSGD	RW	0h	Rank 1 Read DQS Gating Delay: Rank 1 delay select for the read DQS gating (DQSG) LCDL.
7-0	R0DQSGD	RW	0h	Rank 0 Read DQS Gating Delay: Rank 0 delay select for the read DQS gating (DQSG) LCDL.

Table 7-431. Register Call Summary for DDR_PHY_DX3LCDLR2

EMIF Registers
- DDR_PHY_DX3LCDLR2 Register (Offset = 2A8h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.101 DDR_PHY_DX3MDLR Register (Offset = 2ACh) [reset = 0h]

The DATX8 Master Delay Line Registers return different period values measured by the master delay lines in the DATX8 macros. In the default normal operating conditions, the value of the DDR_PHY_DX3MDLR registers change based on ongoing calibration and measurements.

Table 7-432. DDR_PHY_DX3MDLR Instances

Instance	Physical Address
DDR_PHY	0232 92ACh

Figure 7-147. DDR_PHY_DX3MDLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
MDLD							
RW-0h							
15	14	13	12	11	10	9	8
TPRD							
RW-0h							
7	6	5	4	3	2	1	0
IPRD							
RW-0h							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-433. DDR_PHY_DX3MDLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return zeros
23-16	MDLD	RW	0h	MDL Delay: Delay select for the LCDL for the Master Delay Line.
15-8	TPRD	RW	0h	Target Period: Target period measured by the master delay line calibration for VT drift compensation. This is the current measured value of the period and is continuously updated if the MDL is enabled to do so.
7-0	IPRD	RW	0h	Initial Period: Initial period measured by the master delay line calibration for VT drift compensation. This value is used as the denominator when calculating the ratios of updates during VT compensation.

Table 7-434. Register Call Summary for DDR_PHY_DX3MDLR

EMIF Registers
- DDR_PHY_DX3MDLR Register (Offset = 2ACh) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.102 DDR_PHY_DX3GTR Register (Offset = 2B0h) [reset = 55000h]

This register is used to control various timing settings of the byte lane, including DQS gating system latency and write leveling system latency.

Table 7-435. DDR_PHY_DX3GTR Instances

Instance	Physical Address
DDR_PHY	0232 92B0h

Figure 7-148. DDR_PHY_DX3GTR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				R3WLSL		R2WLSL	
R-0h				RW-1h		RW-1h	
15	14	13	12	11	10	9	8
R1WLSL		R0WLSL		R3DGSL			R2DGSL
RW-1h		RW-1h		RW-0h			RW-0h
7	6	5	4	3	2	1	0
R2DGSL		R1DGSL		R0DGSL			
RW-0h		RW-0h		RW-0h			

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-436. DDR_PHY_DX3GTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reads return zeros
19-18	R3WLSL	RW	1h	Rank n Write Leveling System Latency: This is used to adjust the write latency after write leveling. Power-up default is 01 (that is, no extra clock cycles required). The SL fields are initially set by the PUB during automatic write leveling but these values can be overwritten by a direct write to this register. Every two bits of this register control the latency of each of the (up to) four ranks. R0WLSL controls the latency of rank 0, R1WLSL controls rank 1, and so on. Valid values: • 00 = Write latency = WL - 1 • 01 = Write latency = WL • 10 = Write latency = WL + 1 • 11 = Reserved
17-16	R2WLSL	RW	1h	See description for R3WLSL.
15-14	R1WLSL	RW	1h	See description for R3WLSL.
13-12	R0WLSL	RW	1h	See description for R3WLSL.
11-9	R3DGSL	RW	0h	Rank n DQS Gating System Latency: This is used to increase the number of clock cycles needed to expect valid DDR read data by up to seven extra clock cycles. This is used to compensate for board delays and other system delays. Power-up default is 000 (that is, no extra clock cycles required). The SL fields are initially set by the PHY during automatic DQS data training but these values can be overwritten by a direct write to this register. Every three bits of this register control the latency of each of the (up to) four ranks. R0DGSL controls the latency of rank 0, R1DGSL controls rank 1, and so on. Valid values are 0 to 7.
8-6	R2DGSL	RW	0h	See description for R3DGSL.
5-3	R1DGSL	RW	0h	See description for R3DGSL.
2-0	R0DGSL	RW	0h	See description for R3DGSL.

Table 7-437. Register Call Summary for DDR_PHY_DX3GTR

EMIF Registers
• EMIF Registers: [0]

7.2.5.103 DDR_PHY_DX8GCR Register (Offset = 3C0h) [reset = 7C000E81h]

This register is used for miscellaneous configurations specific to a particular instantiation of the DATX8 macro.

Table 7-438. DDR_PHY_DX8GCR Instances

Instance	Physical Address
DDR_PHY	0232 93C0h

Figure 7-149. DDR_PHY_DX8GCR Register

31	30	29	28	27	26	25	24
CALBYP	MDLEN	WLRNKEN				RESERVED	
RW-0h	RW-1h	RW-Fh				R-0h	
23	22	21	20	19	18	17	16
RESERVED				PLLBYP	GSHIFT	PLLPD	PLLRST
R-0h				RW-0h	RW-0h	RW-0h	RW-0h
15	14	13	12	11	10	9	8
DXOEO		RTTOAL	RTTOH		DQRTT	DQSRTT	DSEN
RW-0h		RW-0h	RW-1h		RW-1h	RW-1h	RW-1h
7	6	5	4	3	2	1	0
DSEN	DQSRPD	DXPDR	DXPDD	DXIOM	DQODT	DQSODT	DXEN
RW-1h	RW-0h	RW-0h	RW-0h	RW-0h	RW-0h	RW-0h	RW-1h

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-439. DDR_PHY_DX8GCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CALBYP	RW	0h	Calibration Bypass: Prevents, if set, period measurement calibration from automatically triggering after PHY initialization.
30	MDLEN	RW	1h	Master Delay Line Enable: Enables, if set, the DATX8 master delay line calibration to perform subsequent period measurements following the initial period measurements that are performed after reset or when calibration is manually triggered. These additional measurements are accumulated and filtered as long as this bit remains high. This bit is ANDed with the common DATX8 MDL enable bit.
29-26	WLRNKEN	RW	Fh	Write Level Rank Enable: Specifies the ranks that should be write leveled for this byte. Write leveling responses from ranks that are not enabled for write leveling for a particular byte are ignored and write leveling is flagged as done for these ranks. WLRKEN[0] enables rank 0, [1] enables rank 1, [2] enables rank 2, and [3] enables rank 3.
25-20	RESERVED	R	0h	Reads return zeros
19	PLLBYP	RW	0h	PLL Bypass: Puts the byte PLL in bypass mode by driving the PLL bypass pin. This bit is not self-clearing and a '0' must be written to de-assert the bypass. This bit is ORed with the global BYP configuration bit in DDR_PHY_PLLCR
18	GSHIFT	RW	0h	Gear Shift: Enables, if set, rapid locking mode on the byte PLL
17	PLLPD	RW	0h	PLL Power Down: Puts the byte PLL in power down mode by driving the PLL power down pin. This bit is not self-clearing and a '0' must be written to de-assert the power-down. This bit is ORed with the global PLLPD configuration bit in DDR_PHY_PLLCR
16	PLLRST	RW	0h	PLL Rest: Resets the byte PLL by driving the PLL reset pin. This bit is not selfclearing and a '0' must be written to de-assert the reset. This bit is ORed with the global PLLRST configuration bit in DDR_PHY_PLLCR

Table 7-439. DDR_PHY_DX8GCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-14	DXOEO	RW	0h	Data Byte Output Enable Override: Specifies whether the output I/O output enable for the byte lane should be set to a fixed value. Valid values are: • 00 = No override. Output enable is controlled by DFI transactions • 01 = Ouput enable is asserted (I/O is forced to output mode). • 10 = Output enable is de-asserted (I/O is forced to input mode) • 11 = Reserved
13	RTTOAL	RW	0h	RTT On Additive Latency: Indicates when the ODT control of DQ/DQS SSTL I/Os is set to the value in DQODT/DQSODT during read cycles. Valid values are: • 0 = ODT control is set to DQSODT/DQODT almost two cycles before read data preamble • 1 = ODT control is set to DQSODT/DQODT almost one cycle before read data preamble
12-11	RTTOH	RW	1h	RTT Output Hold: Indicates the number of clock cycles (from 0 to 3) after the read data postamble for which ODT control should remain set to DQSODT for DQS or DQODT for DQ/DM before disabling it (setting it to '0') when using dynamic ODT control. ODT is disabled almost RTTOH clock cycles after the read postamble
10	DQRTT	RW	1h	DQ Dynamic RTT Control: Indicates, if set, that the ODT control of DQ/DM SSTL I/Os be dynamically controlled by setting it to the value in DQODT during reads and disabling it (setting it to '0') during any other cycle. If this bit is not set, then the ODT control of DQ SSTL I/Os is always set to the value in DQODT.
9	DQSRTT	RW	1h	DQS Dynamic RTT Control: Indicates, if set, that the ODT control of DQS SSTL I/Os be dynamically controlled by setting it to the value in DQSODT during reads and disabling it (setting it to '0') during any other cycle. If this bit is not set, then the ODT control of DQS SSTL I/Os is always set to the value in DQSODT field
8-7	DSEN	RW	1h	Write DQS Enable: Controls whether the write DQS going to the SDRAM is enabled (toggling) or disabled (static value) and whether the DQS is inverted. DQS# is always the inversion of DQS. These values are valid only when DQS/DQS# output enable is on, otherwise the DQS/DQS# is tri-stated. Valid settings are: • 00 = DQS disabled (Driven to constant 0) • 01 = DQS toggling with normal polarity (This should be the default setting) • 10 = DQS toggling with inverted polarity • 11 = DQS disabled (Driven to constant 1)
6	DQSRPD	RW	0h	DQSR Power Down: Powers down, if set, the PDQSR cell.
5	DXPDR	RW	0h	Data Power Down Receiver: Powers down, when set, the input receiver on I/O for DQ, DM, and DQS/DQS# pins of the byte.
4	DXPDD	RW	0h	Data Power Down Driver: Powers down, when set, the output driver on I/O for DQ, DM, and DQS/DQS# pins of the byte.
3	DXIOM	RW	0h	Data I/O Mode: Selects SSTL mode (when set to 0) or CMOS mode (when set to 1) of the I/O for DQ, DM, and DQS/DQS# pins of the byte.
2	DQODT	RW	0h	Data On-Die Termination: Enables, when set, the on-die termination on the I/O for DQ and DM pins of the byte.
1	DQSODT	RW	0h	DQS On-Die Termination: Enables, when set, the on-die termination on the I/O for DQS/DQS# pin of the byte

Table 7-439. DDR_PHY_DX8GCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	DXEN	RW	1h	Data Byte Enable: Enables if set the data byte. Setting this bit to '0' disables the byte, that is, the byte is not used in PHY initialization or training and is ignored during SDRAM read/write operations. Note: Software-initiated PHY initialization and leveling is only required after ECC has been enabled in the PHY by writing to DDR_PHY_DX8GCR.DXEN. Once ECC is enabled in the PHY and the PHY is initialized and leveled, the ECC enable bit in the controller (ECCCTL.ECC_EN) can be enabled or disabled without re-triggering PHY initialization and leveling.

Table 7-440. Register Call Summary for DDR_PHY_DX8GCR

EMIF Functional Description • Error Correction And Detection: [0][1]
EMIF Registers • DDR_PHY_DX2GCR Register (Offset = 240h) [reset = 7C000E81h]: [0] • DDR_PHY_DX0GCR Register (Offset = 1C0h) [reset = 7C000E81h]: [0] • DDR_PHY_DX3GCR Register (Offset = 280h) [reset = 7C000E81h]: [0] • DDR_PHY_DX1GCR Register (Offset = 200h) [reset = 7C000E81h]: [0] • EMIF Registers: [0] • DDR_PHY_DX8GCR Register (Offset = 3C0h) [reset = 7C000E81h]: [0]

7.2.5.104 DDR_PHY_DX8GSR0 Register (Offset = 3C4h) [reset = 0h]

DDR_PHY_DX8GSR0 and DDR_PHY_DX8GSR2 are general status registers for the DATX8. They indicate among other things whether write leveling or period measurement calibration is done. Note that WDQCAL, RDQSCAL, RDQSNCAL, GDQSCAL, and WLCAL are calibration measurement-done flags that should be used for debug purposes if the global calibration done flag (DDR_PHY_PGSR0[CAL]) is not asserted. These flags will typically assert for a minimum of two VBUSP_CLK cycles and then de-assert when all measurement done flags are asserted.

Table 7-441. DDR_PHY_DX8GSR0 Instances

Instance	Physical Address
DDR_PHY	0232 93C4h

Figure 7-150. DDR_PHY_DX8GSR0 Register

31	30	29	28	27	26	25	24
RESERVED			WLDQ	QSGERR			
R-0h			R-0h	R-0h			
23	22	21	20	19	18	17	16
GDQSPRD							
R-0h							
15	14	13	12	11	10	9	8
DPLOCK	WLPRD						
R-0h	R-0h						
7	6	5	4	3	2	1	0
WLPRD	WLERR	WLDONE	WLCAL	GDQSCAL	RDQSNCAL	RDQSCAL	WDQCAL
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 7-442. DDR_PHY_DX8GSR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reads return zeros.
28	WLDQ	R	0h	Write Leveling DQ Status: Captures the write leveling DQ status from the DRAM during software write leveling
27-24	QSGERR	R	0h	DQS Gate Training Error: Indicates if set that there is an error in DQS gate training. One bit for each of the up to 4 ranks
23-16	GDQSPRD	R	0h	Read DQS gating Period: Returns the DDR clock period measured by the read DQS gating LCDL during calibration. This value is PVT compensated
15	DPLOCK	R	0h	DATX8 PLL Lock: Indicates, if set, that the DATX8 PLL has locked.
14-7	WLPRD	R	0h	Write Leveling Period: Returns the DDR clock period measured by the write leveling LCDL during calibration. The measured period is used to generate the control of the write leveling pipeline which is a function of the write-leveling delay and the clock period. This value is PVT compensated
6	WLERR	R	0h	Write Leveling Error: Indicates, if set, that there is a write leveling error in the DATX8.
5	WLDONE	R	0h	Write Leveling Done: Indicates, if set, that the DATX8 has completed write leveling.
4	WLCAL	R	0h	Write Leveling Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the write leveling slave delay line
3	GDQSCAL	R	0h	Read DQS gating Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the read DQS gating LCDL.

Table 7-442. DDR_PHY_DX8GSR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	RDQSNCAL	R	0h	Read DQS# Calibration (Type B/B1 PHY Only): Indicates, if set, that the DATX8 has finished doing period measurement calibration for the read DQS# LCDL
1	RDQSCAL	R	0h	Read DQS Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the read DQS LCDL.
0	WDQCAL	R	0h	Write DQ Calibration: Indicates, if set, that the DATX8 has finished doing period measurement calibration for the write DQ LCDL.

Table 7-443. Register Call Summary for DDR_PHY_DX8GSR0

EMIF Registers
- DDR_PHY_DX8GSR0 Register (Offset = 3C4h) [reset = 0h]: [0]
- DDR_PHY_DX8GSR2 Register (Offset = 3F4h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.105 DDR_PHY_DX8GSR2 Register (Offset = 3F4h) [reset = 0h]

DDR_PHY_DX8GSR0 and DDR_PHY_DX8GSR2 are general status registers for the DATX8. They indicate among other things whether write leveling or period measurement calibration is done. Note that WDQCAL, RDQSCAL, RDQSNCAL, GDQSCAL, and WLCAL are calibration measurement-done flags that should be used for debug purposes if the global calibration done flag (DDR_PHY_PGSR0[CAL]) is not asserted. These flags will typically assert for a minimum of two VBUSP_CLK cycles and then de-assert when all measurement done flags are asserted.

Table 7-444. DDR_PHY_DX8GSR2 Instances

Instance	Physical Address
DDR_PHY	0232 93F4h

Figure 7-151. DDR_PHY_DX8GSR2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				ESTAT			
R-0h				R-0h			
7	6	5	4	3	2	1	0
WEWN	WEERR	REWN	REERR	WDWN	WDERR	RDWN	RDERR
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 7-445. DDR_PHY_DX8GSR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reads return zeros.
11-8	ESTAT	R	0h	Error Status: If an error occurred for this lane as indicated by RDERR, WDERR, REERR or WEERR the error status code can provide additional information regard when the error occurred during the algorithm execution.
7	WEWN	R	0h	Write Eye Centering Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the write eye centering training.
6	WEERR	R	0h	Write Eye Centering Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the write eye centering training.
5	REWN	R	0h	Read Eye Centering Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the read eye centering training.
4	REERR	R	0h	Read Eye Centering Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the read eye centering training.
3	WDWN	R	0h	Write Bit Deskew Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the write bit deskew training.
2	WDERR	R	0h	Write Bit Deskew Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the write bit deskew training.

Table 7-445. DDR_PHY_DX8GSR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	RDWN	R	0h	Read Bit Deskew Warning: Indicates, if set, that the byte lane *n* has encountered a warning during execution of the read bit deskew training.
0	RDERR	R	0h	Read Bit Deskew Error: Indicates, if set, that the byte lane *n* has encountered an error during execution of the read bit deskew training.

Table 7-446. Register Call Summary for DDR_PHY_DX8GSR2

EMIF Registers

- DDR_PHY_DX8GSR0 Register (Offset = 3C4h) [reset = 0h]: [0]
- DDR_PHY_DX8GSR2 Register (Offset = 3F4h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.106 DDR_PHY_DX8LCDLR0 Register (Offset = 3E0h) [reset = 0h]

The DATX8 local calibrated delay line registers 0-2 are used to select the delay value on the LCDLs used in the DATX8 macros. A single LCDL field in the LCDLR register connects to a corresponding LCDL. The following tables describe the bits of the DATX8 LCDLR registers. The write data delay (WDQD) and the read DQS delay (RDQSD) are automatically derived from the measured period during calibration. WDQD and RDQSD correspond to a 90 degrees phase shift for DQ during writes and DQS during reads, respectively. The 90 degrees phase shift is used to centre DQS into the write and read data eyes. A 90 degrees phase shift is equivalent to half the DDR clock period. After calibration WDQD and RDQSD fields will contain a value that corresponds to half the DDR clock period (or a quarter of the SDRAM clock period). The write leveling delay (RnWLD) and read DQS gating delay DQSGD are derived from the write leveling and DQS training algorithms. These are normally run automatically by the PHY control block or they can be executed in software by the user. After calibration RnWLD field will contain a value that corresponds to the DDR clock period (or half of the SDRAM clock period), while RnDQSGD field will contain a value that corresponds to half the DDR clock period (or a quarter of the SDRAM clock period).

After the initial setting, all LCDL register fields are automatically updated by the VT compensation logic to track drifts due to voltage and temperature. The user can however override these values by writing to the respective fields of the register and/or optionally disabling the automatic drift compensation.

Table 7-447. DDR_PHY_DX8LCDLR0 Instances

Instance	Physical Address
DDR_PHY	0232 93E0h

Figure 7-152. DDR_PHY_DX8LCDLR0 Register

31	30	29	28	27	26	25	24
R3WLD							
RW-0h							
23	22	21	20	19	18	17	16
R2WLD							
RW-0h							
15	14	13	12	11	10	9	8
R1WLD							
RW-0h							
7	6	5	4	3	2	1	0
R0WLD							
RW-0h							

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-448. DDR_PHY_DX8LCDLR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	R3WLD	RW	0h	Rank 3 Write Leveling Delay: Rank 3 delay select for the write leveling (WL) LCDL
23-16	R2WLD	RW	0h	Rank 2 Write Leveling Delay: Rank 2 delay select for the write leveling (WL) LCDL
15-8	R1WLD	RW	0h	Rank 1 Write Leveling Delay: Rank 1 delay select for the write leveling (WL) LCDL
7-0	R0WLD	RW	0h	Rank 0 Write Leveling Delay: Rank 0 delay select for the write leveling (WL) LCDL

Table 7-449. Register Call Summary for DDR_PHY_DX8LCDLR0

EMIF Registers
• EMIF Registers: [0]

7.2.5.107 DDR_PHY_DX8LCDLR1 Register (Offset = 3E4h) [reset = 0h]

The DDR_PHY_DX8LCDLR1 register is described in the figure and table below.

Table 7-450. DDR_PHY_DX8LCDLR1 Instances

Instance	Physical Address
DDR_PHY	0232 93E4h

Figure 7-153. DDR_PHY_DX8LCDLR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RDQSND							
RW-0h							
15	14	13	12	11	10	9	8
RDQSD							
RW-0h							
7	6	5	4	3	2	1	0
WDQD							
RW-0h							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-451. DDR_PHY_DX8LCDLR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return zeros
23-16	RDQSND	RW	0h	Read DQSN Delay: Delay select for the read DQSN (RDQS) LCDL
15-8	RDQSD	RW	0h	Read DQS Delay: Delay select for the read DQS (RDQS) LCDL
7-0	WDQD	RW	0h	Write Data Delay: Delay select for the write data (WDQ) LCDL.

Table 7-452. Register Call Summary for DDR_PHY_DX8LCDLR1

EMIF Registers
- DDR_PHY_DX8LCDLR1 Register (Offset = 3E4h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.108 DDR_PHY_DX8LCDLR2 Register (Offset = 3E8h) [reset = 0h]

The DDR_PHY_DX8LCDLR2 register is described in the figure and table below.

Table 7-453. DDR_PHY_DX8LCDLR2 Instances

Instance	Physical Address
DDR_PHY	0232 93E8h

Figure 7-154. DDR_PHY_DX8LCDLR2 Register

31	30	29	28	27	26	25	24
R3DQSGD							
RW-0h							
23	22	21	20	19	18	17	16
R2DQSGD							
RW-0h							
15	14	13	12	11	10	9	8
R1DQSGD							
RW-0h							
7	6	5	4	3	2	1	0
R0DQSGD							
RW-0h							

LEGEND: RW = Read/Write; *-n* = value after reset

Table 7-454. DDR_PHY_DX8LCDLR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	R3DQSGD	RW	0h	Rank 3 Read DQS Gating Delay: Rank 3 delay select for the read DQS gating (DQSG) LCDL.
23-16	R2DQSGD	RW	0h	Rank 2 Read DQS Gating Delay: Rank 2 delay select for the read DQS gating (DQSG) LCDL.
15-8	R1DQSGD	RW	0h	Rank 1 Read DQS Gating Delay: Rank 1 delay select for the read DQS gating (DQSG) LCDL.
7-0	R0DQSGD	RW	0h	Rank 0 Read DQS Gating Delay: Rank 0 delay select for the read DQS gating (DQSG) LCDL.

Table 7-455. Register Call Summary for DDR_PHY_DX8LCDLR2

EMIF Registers
- DDR_PHY_DX8LCDLR2 Register (Offset = 3E8h) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.109 DDR_PHY_DX8MDLR Register (Offset = 3ECh) [reset = 0h]

The DATX8 Master Delay Line Registers return different period values measured by the master delay lines in the DATX8 macros. In the default normal operating conditions, the value of the DDR_PHY_DX8MDLR registers change based on ongoing calibration and measurements.

Table 7-456. DDR_PHY_DX8MDLR Instances

Instance	Physical Address
DDR_PHY	0232 93ECh

Figure 7-155. DDR_PHY_DX8MDLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
MDLD							
RW-0h							
15	14	13	12	11	10	9	8
TPRD							
RW-0h							
7	6	5	4	3	2	1	0
IPRD							
RW-0h							

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-457. DDR_PHY_DX8MDLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return zeros
23-16	MDLD	RW	0h	MDL Delay: Delay select for the LCDL for the Master Delay Line.
15-8	TPRD	RW	0h	Target Period: Target period measured by the master delay line calibration for VT drift compensation. This is the current measured value of the period and is continuously updated if the MDL is enabled to do so.
7-0	IPRD	RW	0h	Initial Period: Initial period measured by the master delay line calibration for VT drift compensation. This value is used as the denominator when calculating the ratios of updates during VT compensation.

Table 7-458. Register Call Summary for DDR_PHY_DX8MDLR

EMIF Registers
- DDR_PHY_DX8MDLR Register (Offset = 3ECh) [reset = 0h]: [0]
- EMIF Registers: [0]

7.2.5.110 DDR_PHY_DX8GTR Register (Offset = 3F0h) [reset = 55000h]

This register is used to control various timing settings of the byte lane, including DQS gating system latency and write leveling system latency.

Table 7-459. DDR_PHY_DX8GTR Instances

Instance	Physical Address
DDR_PHY	0232 93F0h

Figure 7-156. DDR_PHY_DX8GTR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				R3WLSL		R2WLSL	
R-0h				RW-1h		RW-1h	
15	14	13	12	11	10	9	8
R1WLSL		R0WLSL		R3DGSL			R2DGSL
RW-1h		RW-1h		RW-0h			RW-0h
7	6	5	4	3	2	1	0
R2DGSL		R1DGSL		R0DGSL			
RW-0h		RW-0h		RW-0h			

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 7-460. DDR_PHY_DX8GTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reads return zeros
19-18	R3WLSL	RW	1h	Rank n Write Leveling System Latency: This is used to adjust the write latency after write leveling. Power-up default is 01 (that is, no extra clock cycles required). The SL fields are initially set by the PUB during automatic write leveling but these values can be overwritten by a direct write to this register. Every two bits of this register control the latency of each of the (up to) four ranks. R0WLSL controls the latency of rank 0, R1WLSL controls rank 1, and so on. Valid values: • 00 = Write latency = WL - 1 • 01 = Write latency = WL • 10 = Write latency = WL + 1 • 11 = Reserved
17-16	R2WLSL	RW	1h	See description for R3WLSL.
15-14	R1WLSL	RW	1h	See description for R3WLSL.
13-12	R0WLSL	RW	1h	See description for R3WLSL.
11-9	R3DGSL	RW	0h	Rank n DQS Gating System Latency: This is used to increase the number of clock cycles needed to expect valid DDR read data by up to seven extra clock cycles. This is used to compensate for board delays and other system delays. Power-up default is 000 (that is, no extra clock cycles required). The SL fields are initially set by the PHY during automatic DQS data training but these values can be overwritten by a direct write to this register. Every three bits of this register control the latency of each of the (up to) four ranks. R0DGSL controls the latency of rank 0, R1DGSL controls rank 1, and so on. Valid values are 0 to 7.
8-6	R2DGSL	RW	0h	See description for R3DGSL.
5-3	R1DGSL	RW	0h	See description for R3DGSL.
2-0	R0DGSL	RW	0h	See description for R3DGSL.

Table 7-461. Register Call Summary for DDR_PHY_DX8GTR

EMIF Registers
• EMIF Registers: [0]

7.3 General-Purpose Memory Controller (GPMC)

This section describes the General-Purpose Memory Controller (GPMC) for the device.

NOTE: Due to device limitation, NAND functionality of the GPMC is not supported and the NAND functionality described in Section 7.3, *General-Purpose Memory Controller (GPMC)* is not applicable.

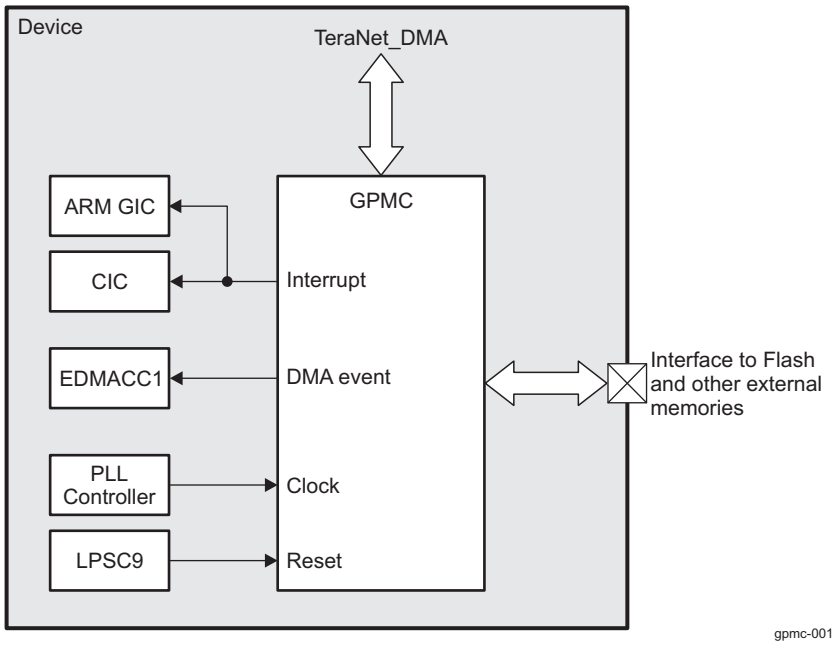
7.3.1 GPMC Overview

The general-purpose memory controller (GPMC) is a unified memory controller dedicated for interfacing with external memory devices like:

- Asynchronous SRAM-like memories and application-specific integrated circuit (ASIC) devices
- Asynchronous, synchronous, and page mode (available only in non-multiplexed mode) burst NOR flash devices
- NAND flash
- Pseudo-SRAM devices

Figure 7-157 shows the GPMC module overview.

Figure 7-157. GPMC Overview



The main features of the GPMC are:

- 8- or 16-bit-wide data path to external memory device
- Supports up to 4 chip select regions of programmable size and programmable base addresses in a total address space of 1 GB
- Supports on-the-fly error code detection using the Bose-ChaudhurI-Hocquenghem (BCH) (t = 4, 8, or 16) or Hamming code to improve the reliability of NAND with a minimum effect on software (NAND flash with 512-byte page size or greater)
- Fully pipelined operation for optimal memory bandwidth usage
- The clock to the external memory is provided from GPMC_FCLK divided by 1, 2, 3, or 4
- Supports programmable autoclock gating when no access is detected
- Independent and programmable control signal timing parameters for setup and hold time on a per-chip basis. Parameters are set according to the memory device timing parameters with a timing granularity

of one GPMC_FCLK clock cycle.

- Flexible internal access time control (wait state) and flexible handshake mode using external WAIT pin monitoring
- Support bus keeping
- Support bus turnaround
- Prefetch and write posting engine associated with DMA controller at system level to achieve full performance from the NAND device with minimum effect on NOR/SRAM concurrent access
- 32-bit TeraNet slave interface which supports non-wrapping and wrapping burst of up to 16x32 bits.

The GPMC supports the following various access types:

- Asynchronous read/write access
- Asynchronous read page access (4, 8, and 16 Word16)
- Synchronous read/write access
- Synchronous read/write burst access without wrap capability (4, 8 and 16 Word16)
- Synchronous read/write burst access with wrap capability (4, 8 and 16 Word16)
- Address-data-multiplexed (AD) access
- Address-address-data (AAD) multiplexed access
- Little-endian access only

The GPMC can communicate with a wide range of external devices:

- External asynchronous or synchronous 8-bit wide memory or device (non burst device)
- External asynchronous or synchronous 16-bit wide memory or device
- External 16-bit non-multiplexed NOR flash device
- External 16-bit address and data multiplexed NOR Flash device
- External 8-bit and 16-bit NAND flash device
- External 16-bit pseudo-SRAM (pSRAM) device

NOTE: Page mode is available only in non-multiplexed mode.

7.3.2 GPMC Environment

This section describes the GPMC application fields from an environment point of view (external connections). It describes GPMC connectivity options and gives four possible interfaces.

7.3.2.1 GPMC Modes

This section shows four GPMC external connection options:

- Figure 7-158 shows a connection between the GPMC and a 16-bit synchronous address/data-multiplexed (or AAD-multiplexed but this protocol uses fewer address pins) external memory device.
- Figure 7-159 shows a connection between the GPMC and a 16-bit synchronous non-multiplexed external memory device.
- Figure 7-160 shows a connection between the GPMC and an 8-bit synchronous non-multiplexed external memory device.
- Figure 7-161 shows a connection between the GPMC and an 8-bit NAND device.

Figure 7-158. GPMC to 16-Bit Address/Data-Multiplexed Memory

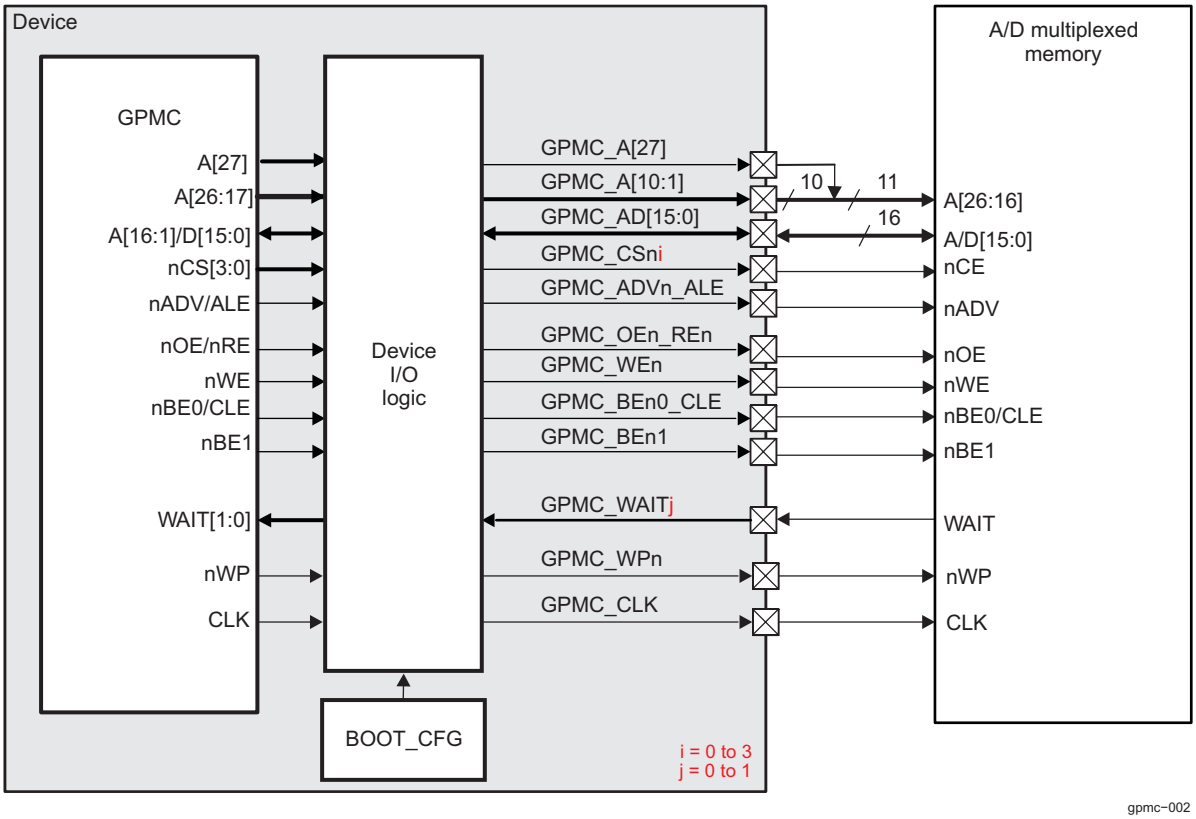


Figure 7-159. GPMC to 16-Bit Non-Multiplexed Memory

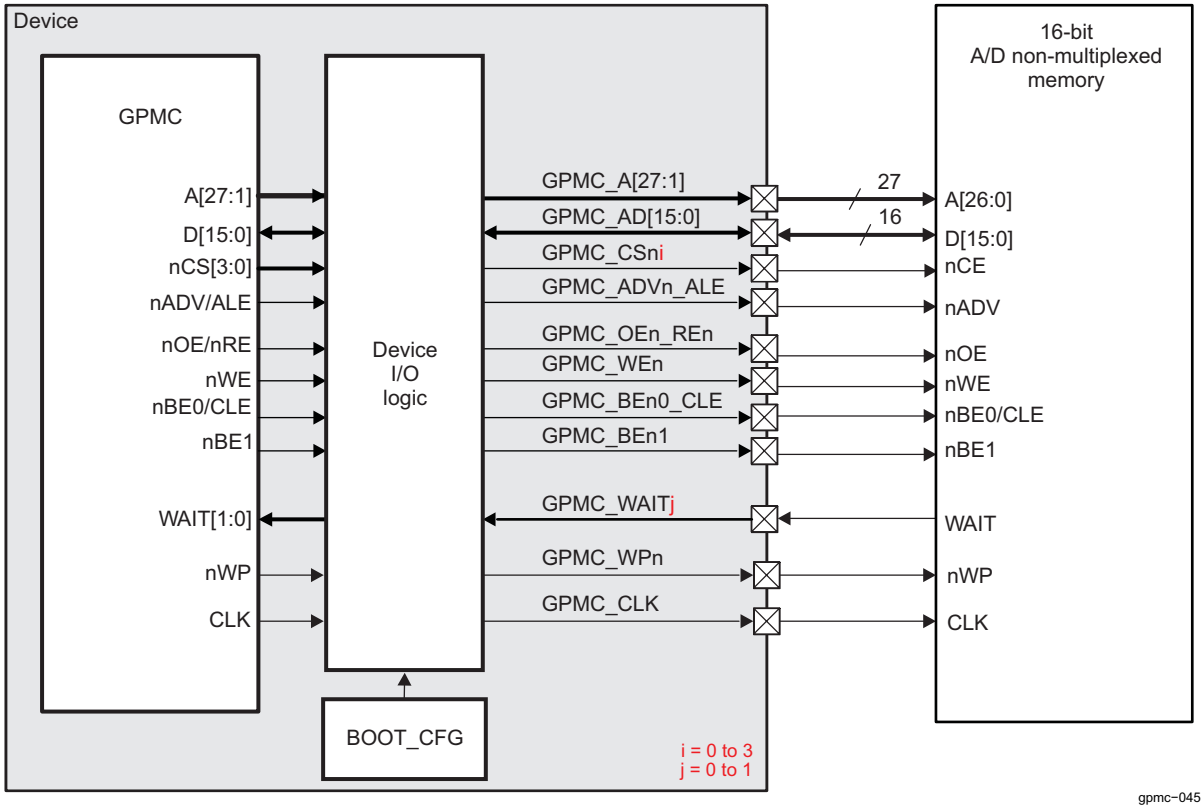


Figure 7-160. GPMC to 8-Bit Non-Multiplexed Memory

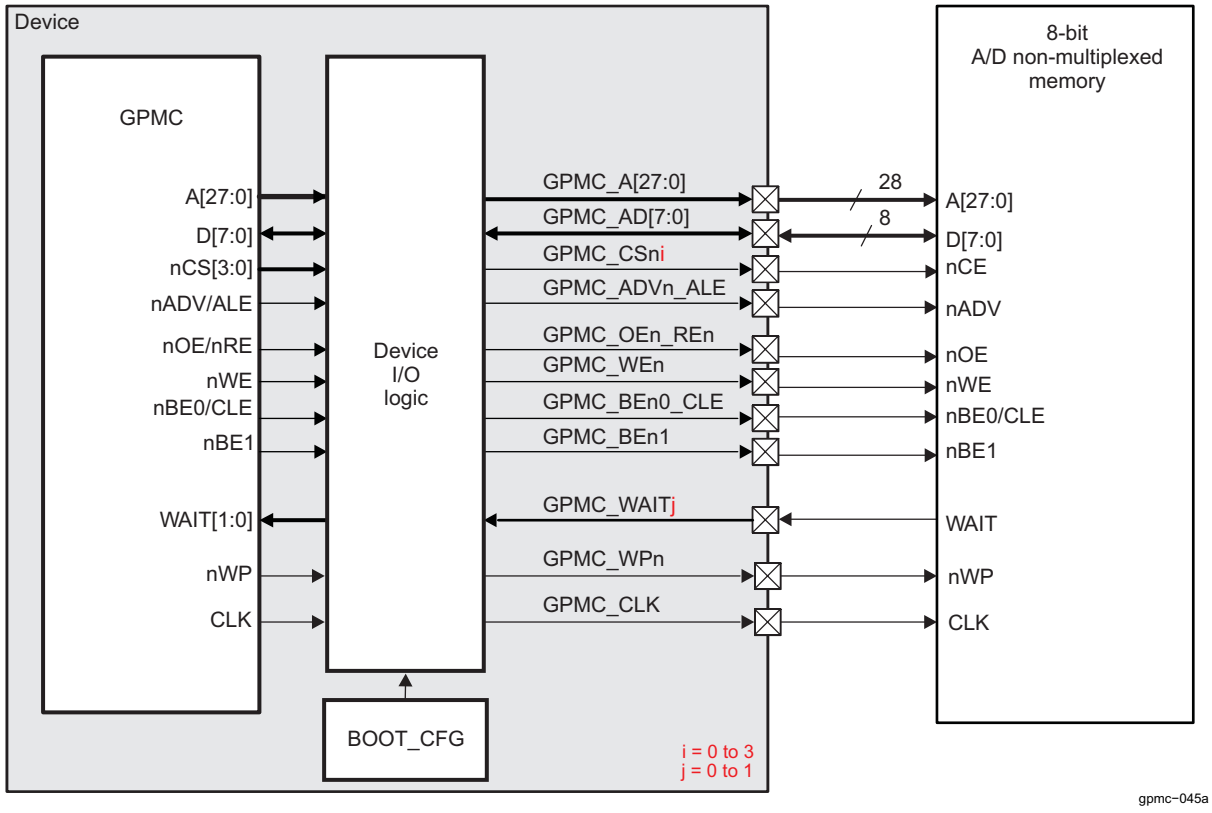
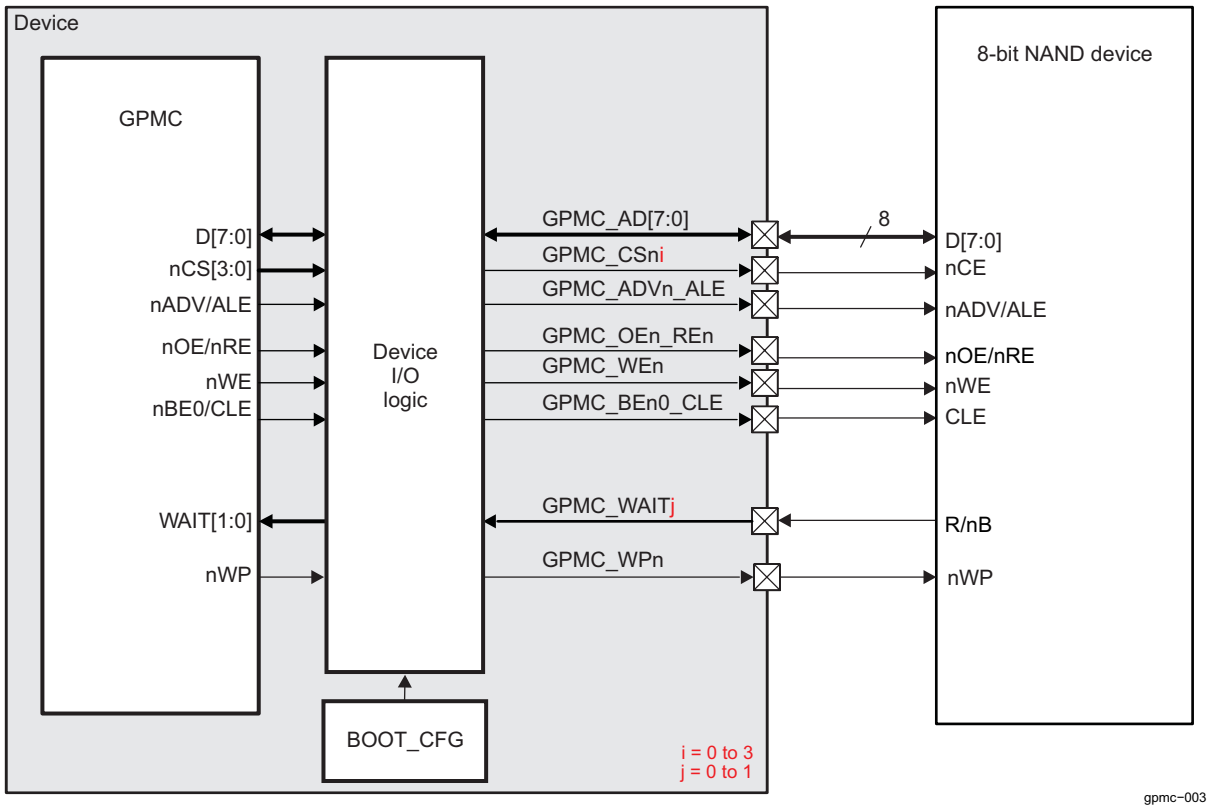


Figure 7-161. GPMC to 8-Bit NAND Device



7.3.2.2 GPMC Signals

Table 7-462 lists the GPMC subsystem input/output (I/O) pins.

Table 7-462. GPMC I/O Description

Pin Name	Device Signal	IOZ[1]	Description
A[27:0]	GPMC_A[27:0]	OZ	28-bit output address bus.
A[16:1]/D[15:0]	GPMC_AD[15:0]	IOZ	Multiplexed address/data.
nCS[3:0]	GPMC_CSn[3:0]	OZ	Chip-selects (active low).
CLK	GPMC_CLK	IOZ	Clock generated for the external memory or device
nADV/ALE	GPMC_ADVn_ALE	OZ	Address valid (active low). Also used as address latch enable (active high) for NAND protocol memories.
nOE/nRE	GPMC_OEn_REn	OZ	Output enable (active low). Also used as read enable (active low) for NAND protocol memories.
nWE	GPMC_WEn	OZ	Write enable (active low)
nBE0/CLE	GPMC_BEn0_CLE	OZ	Lower-byte enable (active low). Also used as command latch enable for NAND protocol memories.
nBE1	GPMC_BEn1	OZ	Upper-byte enable (active low).
WAIT[1:0]	GPMC_WAIT[1:0]	I	External wait signal for NOR and NAND protocol memories. The wait signals can be mapped on any of the chip-selects.
nWP	GPMC_WPn	OZ	Write protect (active low).
DIR	GPMC_DIR	OZ	This signal can be used to control an external buffer direction. It also controls the signal direction of D[15:0]. Low during transmit (for write access: data OUT from GPMC to memory). High during receive (for read access: data IN from memory to GPMC).

(1) I = Input; O = Output; Z = High Impedance

Table 7-463 shows the use of address and data GPMC pins based on the type of external device.

NOTE: Due to device limitation, NAND functionality of the GPMC is not supported and the NAND functionality described in Section 7.3, *General-Purpose Memory Controller (GPMC)* is not applicable.

Table 7-463. GPMC Pin Multiplexing Options

GPMC Pin	Multiplexed Address Data 16-Bit Device	non-multiplexed Address Data 16-Bit Device (complete 28-bit address range)	non-multiplexed Address Data 8-Bit Device (complete 28-bit address range)	16-Bit NAND Device	8-Bit NAND Device
GPMC_A[27]	A27	A27	A27	Not used	Not used
GPMC_A[26]	Not used	A26	A26	Not used	Not used
GPMC_A[25]	Not used	A25	A25	Not used	Not used
GPMC_A[24]	Not used	A24	A24	Not used	Not used
GPMC_A[23]	Not used	A23	A23	Not used	Not used
GPMC_A[22]	Not used	A22	A22	Not used	Not used
GPMC_A[21]	Not used	A21	A21	Not used	Not used
GPMC_A[20]	Not used	A20	A20	Not used	Not used
GPMC_A[19]	Not used	A19	A19	Not used	Not used
GPMC_A[18]	Not used	A18	A18	Not used	Not used
GPMC_A[17]	Not used	A17	A17	Not used	Not used
GPMC_A[16]	Not used	A16	A16	Not used	Not used
GPMC_A[15]	Not used	A15	A15	Not used	Not used

Table 7-463. GPMC Pin Multiplexing Options (continued)

GPMC Pin	Multiplexed Address Data 16-Bit Device	non-multiplexed Address Data 16-Bit Device (complete 28-bit address range)	non-multiplexed Address Data 8-Bit Device (complete 28-bit address range)	16-Bit NAND Device	8-Bit NAND Device
GPMC_A[14]	Not used	A14	A14	Not used	Not used
GPMC_A[13]	Not used	A13	A13	Not used	Not used
GPMC_A[12]	Not used	A12	A12	Not used	Not used
GPMC_A[11]	Not used	A11	A11	Not used	Not used
GPMC_A[10]	A26	A10	A10	Not used	Not used
GPMC_A[9]	A25	A9	A9	Not used	Not used
GPMC_A[8]	A24	A8	A8	Not used	Not used
GPMC_A[7]	A23	A7	A7	Not used	Not used
GPMC_A[6]	A22	A6	A6	Not used	Not used
GPMC_A[5]	A21	A5	A5	Not used	Not used
GPMC_A[4]	A20	A4	A4	Not used	Not used
GPMC_A[3]	A19	A3	A3	Not used	Not used
GPMC_A[2]	A18	A2	A2	Not used	Not used
GPMC_A[1]	A17	A1	A1	Not used	Not used
GPMC_A[0](1)	A0 - Not used	Not used	A0	Not used	Not used
GPMC_AD[15]	A16/D15	D15	Not used	D15	Not used
GPMC_AD[14]	A15/D14	D14	Not used	D14	Not used
GPMC_AD[13]	A14/D13	D13	Not used	D13	Not used
GPMC_AD[12]	A13/D12	D12	Not used	D12	Not used
GPMC_AD[11]	A12/D11	D11	Not used	D11	Not used
GPMC_AD[10]	A11/D10	D10	Not used	D10	Not used
GPMC_AD[9]	A10/D9	D9	Not used	D9	Not used
GPMC_AD[8]	A9/D8	D8	Not used	D8	Not used
GPMC_AD[7]	A8/D7	D7	D7	D7	D7
GPMC_AD[6]	A7/D6	D6	D6	D6	D6
GPMC_AD[5]	A6/D5	D5	D5	D5	D5
GPMC_AD[4]	A5/D4	D4	D4	D4	D4
GPMC_AD[3]	A4/D3	D3	D3	D3	D3
GPMC_AD[2]	A3/D2	D2	D2	D2	D2
GPMC_AD[1]	A2/D1	D1	D1	D1	D1
GPMC_AD[0]	A1/D0	D0	D0	D0	D0

(1) Used to effectively address 8-bit (only) non-multiplexed memories

With all device types, the GPMC does not drive unnecessary address lines. They stay at their reset value of 0h.

Address mapping supports address/data-multiplexed 16-bit-wide devices:

- The NOR flash memory controller still supports non-multiplexed address and data memory devices.
- Multiplexing mode can be selected through the GPMC_CONFIG1_i[9-8] MUXADDDATA bit field (where i = 0 to 3).
- Asynchronous page mode is not supported for multiplexed address and data devices.

7.3.3 GPMC Integration

This section describes module integration in the device, including information about clocks, resets, and hardware requests.

Table 7-466 shows the GPMC integration.

Figure 7-162. GPMC Integration

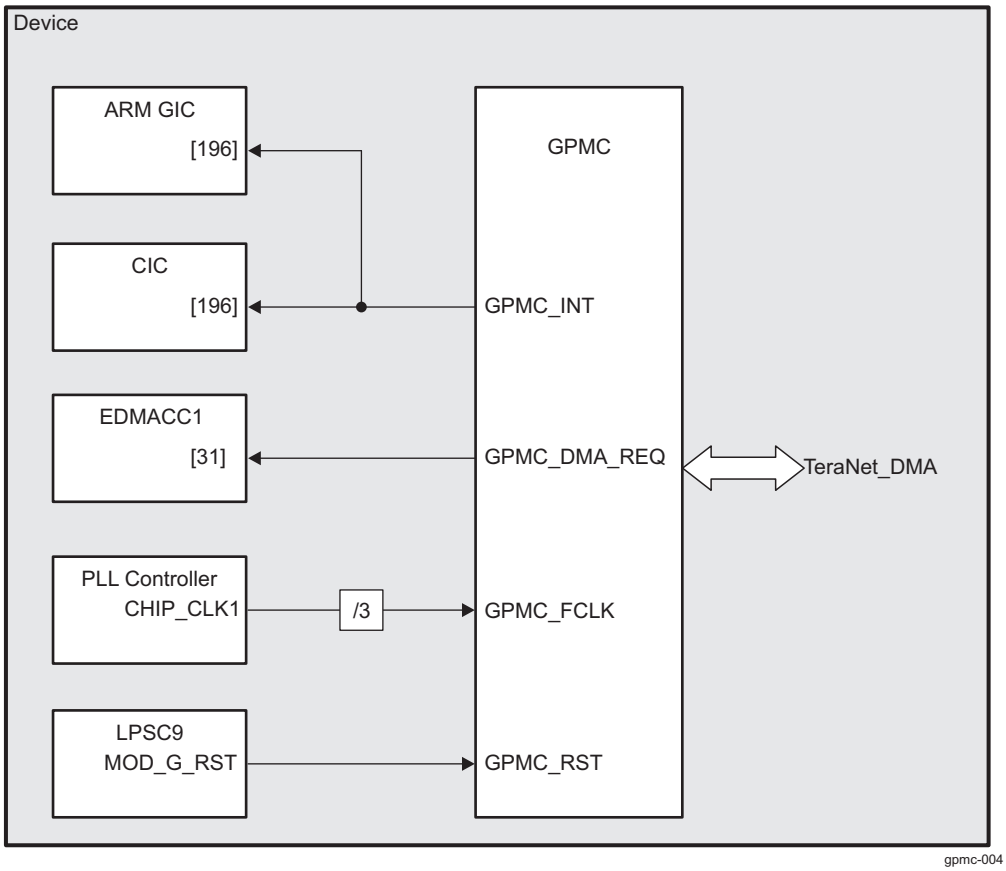


Table 7-464 through Table 7-466 summarize the integration of the GPMC module in the device.

Table 7-464. GPMC Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
GPMC	PD5	LPSC9	TeraNet_DMA

Table 7-465. GPMC Clocks and Resets

Clocks				
Module Instance	**Destination Signal Name**	**Source Signal Name**	**Source**	**Description**
GPMC	GPMC_FCLK	CHIP_CLK1/3	PLL Controller	Functional and interface clock
Resets				
Module Instance	**Destination Signal Name**	**Source Signal Name**	**Source**	**Description**
GPMC	GPMC_RST	MOD_G_RST	LPSC9	Module asynchronous reset

NOTE: For clock description, see Section 7.3.4.2, *GPMC Clock Configuration*.

Table 7-466. GPMC Hardware Requests

Interrupt Requests				
Module Instance	**Event Name**	**Mapped To Input Event [Number]**		**Description**
		ARM GIC	**CIC**	
GPMC	GPMC_INT	[196]	[196]	GPMC interrupt
DMA Requests				
Module Instance	**Event Name**	**Mapped To Input Event [Number]**		**Description**
		EDMACC1		
GPMC	GPMC_DMA_REQ	[31]		DMA event

NOTE: For interrupt description, see Section 7.3.4.5, *GPMC Interrupt Requests*.

7.3.4 GPMC Functional Description

The GPMC basic programming model offers maximum flexibility to support various access protocols for each of the four configurable chip-selects. Use optimal chip-select settings, based on the characteristics of the external device:

- Different protocols can be selected to support generic asynchronous or synchronous random-access devices (NOR flash, SRAM) or to support specific NAND devices.
- The address and data bus can be multiplexed on the same external bus.
- Read and write access can be independently defined as asynchronous or synchronous.
- System requests (byte, 16-bit word, burst) are performed through single or multiple accesses. External access profiles (single, multiple with optimized burst length, native- or emulated-wrap) are based on external device characteristics (supported protocol, bus width, data buffer size, native-wrap support).
- System burst read or write requests are synchronous-burst (multiple-read or multiple-write). When neither burst nor page mode is supported by external memory or ASIC devices, system burst read or write requests are translated to successive single synchronous or asynchronous accesses (single reads or single writes). 8-bit wide devices are supported only in single synchronous or single asynchronous read or write mode.
- To simulate a programmable internal-wait-state, an external WAIT pin can be monitored to dynamically control external access at the beginning (initial access time) of and during a burst access.

Each control signal is controlled independently for each chip-select. The internal functional clock of the GPMC (GPMC_FCLK) is used as a time reference to specify the following:

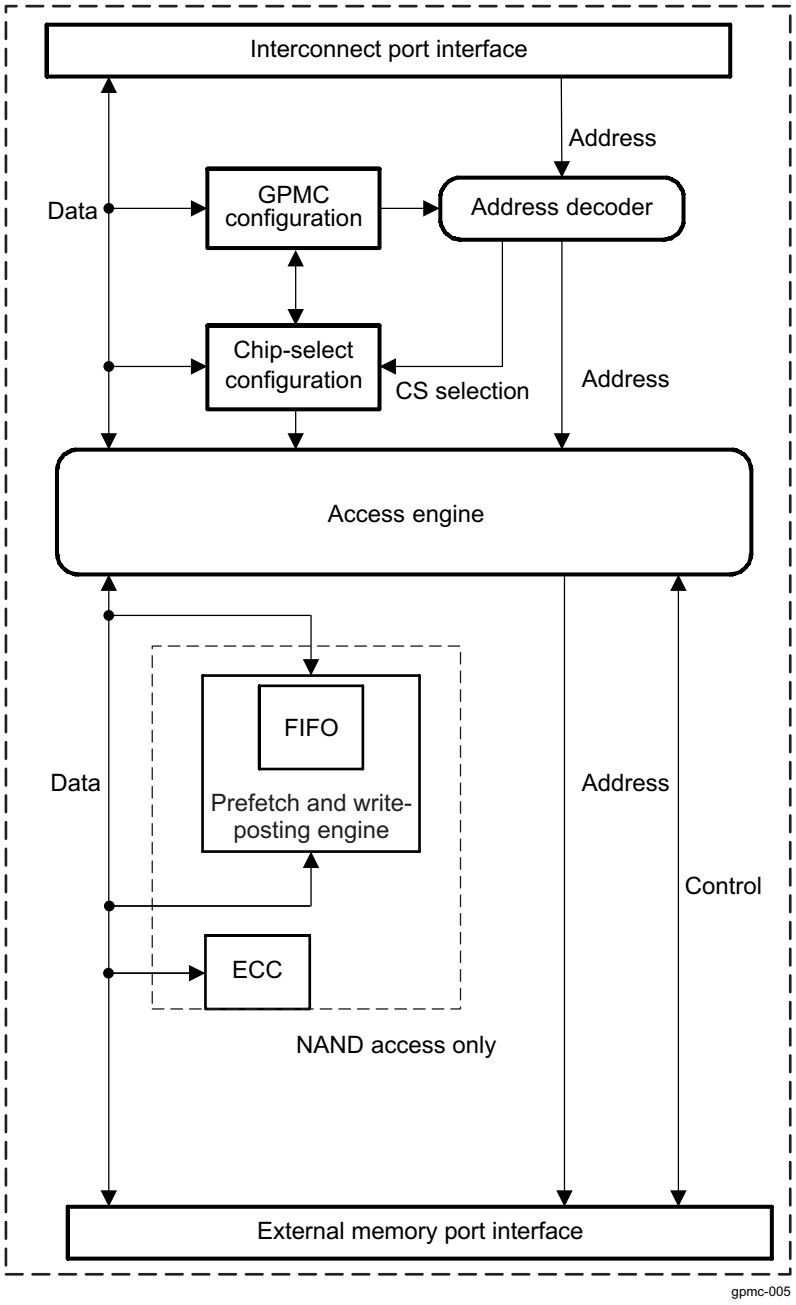
- Read- and write-access duration
- Most GPMC external interface control-signal assertion and deassertion times
- Data-capture time during read access
- External wait-pin monitoring time
- Duration of idle time between accesses, when required

7.3.4.1 GPMC Block Diagram

Figure 7-163 shows the GPMC functional block diagram. The GPMC consists of six blocks:

- Interconnect port interface
- Address decoder, GPMC configuration, and chip-select configuration register file
- Access engine
- Prefetch and write-posting engine
- Error correction code engine (ECC)
- External device/memory port interface

Figure 7-163. GPMC Block Diagram



The GPMC can access various external devices. The flexible programming model allows a wide range of attached device types and access schemes.

Based on the programmed configuration bit fields stored in the GPMC registers, the GPMC can generate the timing of all control signals depending on the attached device and access type.

Given the chip-select decoding and its associated configuration registers, the GPMC selects the appropriate control signal timing for the device type.

7.3.4.2 GPMC Clock Configuration

Table 7-467 describes the GPMC clocks.

Table 7-467. GPMC Clocks

Signal	I/O[1]	Description
GPMC_FCLK	I	Functional and interface clock
GPMC_CLK	O	External clock provided to synchronous external memory devices

(1) I = Input; O = Output

The GPMC_CLK is generated by the GPMC from the internal GPMC_FCLK clock. The source of the GPMC_FCLK is described in Table 7-465. The GPMC_CLK is configured using the GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER bit field (where i = 0 to 3), as shown in Table 7-468.

Table 7-468. GPMC_CLK Configuration

Source Clock	GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER	GPMC_CLK Generated Clock Provided to External Memory Device
GPMC_FCLK	00	GPMC_FCLK
	01	GPMC_FCLK/2
	10	GPMC_FCLK/3
	11	GPMC_FCLK/4

7.3.4.3 GPMC Software Reset

The GPMC can be reset by software through the GPMC_SYSCONFIG[1] SOFTRESET bit. Setting the bit to 1 enables an active software reset that is functionally equivalent to a hardware reset. Hardware and software resets initialize all GPMC registers and the finite state-machine (FSM) immediately and unconditionally. The GPMC_SYSSTATUS[0] RESETDONE bit indicates that the software reset is complete when its value is 1. Software must ensure that the software reset completes before performing GPMC operations.

7.3.4.4 GPMC Power Management

Table 7-469 describes the local power-management features available for the GPMC module.

Table 7-469. GPMC Local Power-Management Features

Feature	Registers	Description
Clock autogating	GPMC_SYSCONFIG[0] AUTOIDLE	This bit allows a local power optimization inside the module, by gating the GPMC_FCLK clock upon the internal activity.
Slave idle modes	GPMC_SYSCONFIG[4-3] SIDLEMODE	Force-idle, no-idle and smart-idle modes are available.

7.3.4.5 GPMC Interrupt Requests

The GPMC generates one interrupt request (see Table 7-466).

Table 7-470 lists the event flags, and their mask, that can cause module interrupts.

Table 7-470. GPMC Interrupt Events

Event Flag	Event Mask	Sensitivity	Description
GPMC_IRQSTATUS[9] WAIT1EDGE DETECTIONSTATUS	GPMC_IRQENABLE[9] WAIT1EDGE DETECTIONENABLE	Edge	Wait1 edge detection interrupt: Triggered if a rising or falling edge is detected on the GPMC_WAIT1 signal. The rising or falling edge detection of Wait1 is selected through the GPMC_CONFIG[9] WAIT1PINPOLARITY bit.
GPMC_IRQSTATUS[8] WAIT0EDGE DETECTIONSTATUS	GPMC_IRQENABLE[8] WAIT0EDGE DETECTIONENABLE	Edge	Wait0 edge detection interrupt: Triggered if a rising or falling edge is detected on the GPMC_WAIT0 signal. The rising or falling edge detection of Wait0 is selected through the GPMC_CONFIG[8] WAIT0PINPOLARITY bit.

Table 7-470. GPMC Interrupt Events (continued)

Event Flag	Event Mask	Sensitivity	Description
GPMC_IRQSTATUS[1] TERMINAL COUNTSTATUS	GPMC_IRQENABLE[1] TERMINAL COUNTENABLE	Level	Terminal count event: Triggered on prefetch process completion; that is, when the number of currently remaining data to be requested reaches 0
GPMC_IRQSTATUS[0] FIFOEVENTSTATUS	GPMC_IRQENABLE[0] FIFOEVENTENABLE	Level	FIFO event interrupt: Indicates available FIFO levels for write-posting mode and prefetch mode. GPMC_PREFETCH_CONFIG1[2] DMAMODE must be set to 0.

7.3.4.6 Interconnect Port Interface

The GPMC interconnect interface is a pipelined interface including a 16 × 32-bit word write buffer.

Any system host can issue external access requests through the GPMC.

The device system can issue the following requests through this interface:

- One 8-, 16-, or 32-bit interconnect access (read/write)
- Two incrementing 32-bit interconnect accesses (read/write)
- Two wrapped 32-bit interconnect accesses (read/write)
- Four incrementing 32-bit interconnect accesses (read/write)
- Four wrapped 32-bit interconnect accesses (read/write)
- Eight incrementing 32-bit interconnect accesses (read/write)
- Eight wrapped 32-bit interconnect accesses (read/write)

Only linear burst transactions are supported; interleaved burst transactions are not supported. Only power-of-two-length precise bursts 2 × 32, 4 × 32, 8 × 32, and 16 × 32, with the burst base address aligned on the total burst size, are supported (this limitation applies to incrementing bursts only).

This interface also provides one interrupt and one DMA request line for specific event control.

It is recommended to program the GPMC_CONFIG1_i[24-23] ATTACHEDDEVICEPAGELENGTH bit field according to the page length of the effective attached device and to enable the GPMC_CONFIG1_i[31] WRAPBURST bit if the attached device supports wrapping burst.

It is possible, however, to emulate wrapping burst on a nonwrapping memory by providing relevant addresses within the page or by splitting transactions. Bursts larger than the memory page length are chopped into multiple burst transactions. Because of the alignment requirements, a page boundary is never crossed.

7.3.4.7 GPMC Address and Data Bus

The current application supports GPMC connection to NAND devices and to address/data-multiplexed memories or devices. Connection to address/data-non-multiplexed memories or devices is supported with a limited address range of 2KB.

Depending on the GPMC configuration of each chip-select, address and data bus lines that are not required for a particular access protocol are not updated (changed from current value) and are not sampled when input (input data bus).

- For address/data-multiplexed and AAD-multiplexed NOR devices, the address is multiplexed on the data bus.
- 8-bit-wide NOR devices do not use GPMC I/O: GPMC_AD[15:8] for data (they are used for address if needed).
- 16-bit-wide NAND devices do not use GPMC I/O: GPMC_A[27:16].
- 8-bit-wide NAND devices do not use GPMC I/O: GPMC_A[27:16] and GPMC I/O: GPMC_AD[15:8].

7.3.4.7.1 GPMC I/O Configuration Setting

NOTE: In this section and the following sections, the i in GPMC_CONFIGx_i stands for the GPMC chip-select i, where i = 0 to 3.

To select a NAND device, program the following register fields:

- GPMC_CONFIG1_i[11-10] DEVICETYPE = 0b10
- GPMC_CONFIG1_i[9-8] MUXADDDATA = 0b00

To select an address/data-multiplexed device, program the following register fields:

- GPMC_CONFIG1_i[11-10] DEVICETYPE = 0b00
- GPMC_CONFIG1_i[9-8] MUXADDDATA = 0b10

To select an address/address/data-multiplexed device, program the following register fields:

- GPMC_CONFIG1_i[11-10] DEVICETYPE = 0b00
- GPMC_CONFIG1_i[9-8] MUXADDDATA = 0b01

To select an address/data-non-multiplexed device, program the following register fields:

- GPMC_CONFIG1_i[11-10] DEVICETYPE = 0b00
- GPMC_CONFIG1_i[9-8] MUXADDDATA = 0b00

7.3.4.8 Address Decoder and Chip-Select Configuration

Addresses are decoded accordingly with the address request of the chip-select and the content of the chip-select base address register file, which includes a set of global GPMC configuration registers and four sets of chip-select configuration registers.

After the chip-select is configured, the access engine accesses the external device, drives the external interface control signals, and applies the interface protocol based on user-defined timing parameters and settings.

7.3.4.8.1 Chip-Select Base Address and Region Size

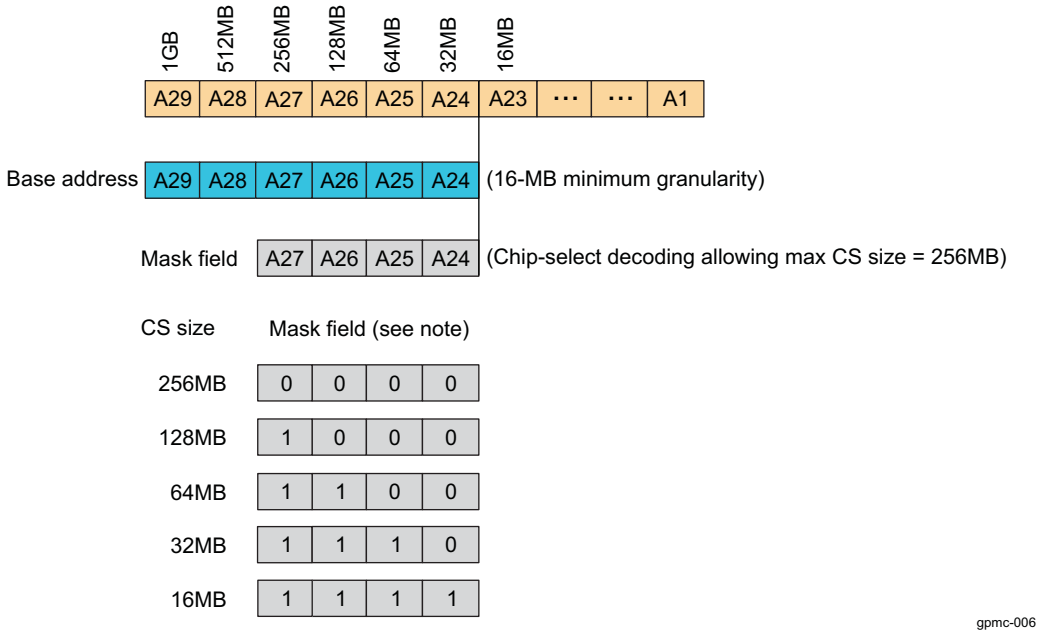
Any external memory or ASIC device attached to the GPMC external interface can be accessed by any device system host within the GPMC 512-MB contiguous address space. For more information, see Chapter 2, *Memory Mapping*.

The GPMC 512-MB address space can be divided into a maximum of four chip-select regions with programmable base address and programmable chip-select size. The chip-select size is programmable from 16 MB to 256 MB (must be a power-of-two) and is defined by the mask field. Attached memory smaller than the programmed chip-select region size is accessed through the entire chip-select region (aliasing).

Each chip-select has a 6-bit base address encoding and 4-bit decoding mask, which must be programmed according to the following rules:

- The programmed chip-select region base address must be aligned on the chip-select region size address boundary and is limited to a power-of-two address value. During access decoding, the value of the register base address is used to compare the address with the address bit line mapping, as shown in Figure 7-164 (with A0 as the device system byte-address line). The base address is programmed through the GPMC_CONFIG7_i[5-0] BASEADDRESS bit field.
- The register mask is used to exclude some address lines from the decoding. A register mask bit field set to 0 suppresses the associated address line from the address comparison (incoming address bit line is don't care). The value of the register mask must be limited to the subsequent value, based on the desired chip-select region size. Any other value has an undefined result. When multiple chip-select regions with overlapping addresses are enabled concurrently, access to these chip-select regions is cancelled and a GPMC access error is posted. The mask field is programmed through the GPMC_CONFIG7_i[11-8] MASKADDRESS bit field.

Figure 7-164. Chip-Select Address Mapping and Decoding Mask



Chip-select configuration (base and mask address or any protocol and timing settings) must be performed while the associated chip-select is disabled through the GPMC_CONFIG7_i[6] CSVALID bit (where i stands for the GPMC chip-select i, where i = 0 to 3). In addition, a chip-select configuration can be disabled only if there is no ongoing access to that chip-select. This requires monitoring the activity of the prefetch or write-posting engine if the engine is active on the chip-select. Also, the write buffer state must be monitored to wait for any posted write completion to the chip-select.

Any access attempted to a nonvalid GPMC address region (CSVALID disabled or address decoding outside a valid chip-select region) is not propagated to the external interface and a GPMC access error is posted. In case of overlapping chip-selects, an error is generated and no access occurs on either chip-select.

CS0 is the only chip-select region enabled after a power up or GPMC reset.

CAUTION

Although the GPMC interface can drive up to four chip-selects, the frequency specified for this interface is for a specific load. If this load is exceeded, the maximum frequency cannot be reached. One solution is to implement a board with buffers to allow the slowest device to maintain the total load on the lines at the value specified in the device Data Manual.

7.3.4.8.2 Access Protocol

7.3.4.8.2.1 Supported Devices

The access protocol of each chip-select can be independently specified through the GPMC_CONFIG1_i[11-10] DEVICETYPE parameter (where i = 0 to 3) for:

- Random-access synchronous or asynchronous memory, such as NOR flash and SRAM
- NAND flash asynchronous devices

NOTE: For more information about the NAND flash GPMC basic programming model and NAND support, see Section 7.3.4.12, *NAND Device Basic Programming Model*, and Section 7.3.4.12.1, *NAND Memory Device in Byte or Word16 Stream Mode*.

7.3.4.8.2.2 Access Size Adaptation and Device Width

Each chip-select can be independently configured through the GPMC_CONFIG1_i[13-12] DEVICESIZE bit field (where i = 0 to 3) to interface with a 16- or 8-bit-wide device. System requests with data width greater than the external device data bus width are split into successive accesses according to the external device data-bus width and little-endian data organization.

7.3.4.8.2.3 Address/Data-Multiplexing Interface

For random synchronous or asynchronous memory interfacing (DEVICETYPE = 0b00), an address- and data-multiplexing protocol can be selected through the GPMC_CONFIG1_i[9-8] MUXADDDATA bit field (where i = 0 to 3). The nADV signal must be used as the external device address latch control signal. For the associated chip-select configuration, nADV assertion and deassertion time and nOE assertion time must be set to the appropriate value to meet the address latch setup/hold time requirements of the external device. See Section 7.3.3, *GPMC Integration*.

NOTE: This address/data-multiplexing interface is not applicable to NAND device interfacing. NAND devices require a specific address, command, and data-multiplexing protocol. See Section 7.3.4.12, *NAND Device Basic Programming Model*.

7.3.4.8.3 External Signals

7.3.4.8.3.1 WAIT Pin Monitoring Control

GPMC access time can be dynamically controlled using an external GPMC_WAIT pin when the external device access time is not deterministic and cannot be defined and controlled using only the GPMC internal RDACCESSTIME, WRACCESSTIME, and PAGEBURSTACCESSTIME wait-state generator.

The GPMC features two input wait pins: GPMC_WAIT1, and GPMC_WAIT0. These pins allow control of external devices with different WAIT pin polarity. They also allow the overlap of WAIT pin assertion from different devices without affecting access to devices for which the WAIT pin is not asserted.

- The GPMC_CONFIG1_i[17-16] WAITPINSELECT bit field (where i = 0 to 3) selects which input GPMC_WAIT pin is used for the device attached to the corresponding chip-select.
- The polarity of the WAIT pin is defined through the WAITxPINPOLARITY bit of the GPMC_CONFIG register. A WAIT pin configured to be active low means that low level on the WAIT signal indicates that the data is not ready and that the data bus is invalid. When a WAIT pin is inactive, data is valid.

The GPMC access engine can be configured per chip-select to monitor or not the WAIT pin of the external memory device, based on the access type: read or write.

- The GPMC_CONFIG1_i[22] WAITREADMONITORING bit defines whether or not the WAIT pin must be monitored during read accesses.
- The GPMC_CONFIG1_i[21] WAITWRITEMONITORING bit defines whether or not the WAIT pin must be monitored during write accesses.

The GPMC access engine can be configured to monitor the WAIT pin of the external memory device asynchronously or synchronously with the GPMC_CLK clock, depending on the access type: synchronous or asynchronous (the GPMC_CONFIG1_i[29] READTYPE and GPMC_CONFIG1_i[27] WRITETYPE bits).

7.3.4.8.3.1.1 Wait Monitoring During Asynchronous Read Access

When WAIT pin monitoring is enabled for read accesses (WAITREADMONITORING), the effective access time is a logical AND combination of the RDACCESSTIME timing completion and the wait-deasserted state.

During asynchronous read accesses with WAIT pin monitoring enabled, the WAIT pin must be at a valid level (asserted or deasserted) for at least two GPMC clock cycles before RDACCESSTIME completes, to ensure correct dynamic access-time control through WAIT pin monitoring. The advance pipelining of the two GPMC clock cycles is the result of the internal synchronization requirements for the WAIT signal.

In this context, RDACCESSTIME is used as a wait invalid timing window and is set to such a value that the WAIT pin is at a valid state two GPMC clock cycles before RDACCESSTIME completes.

Similarly, during a multiple-access cycle (for example, asynchronous read page mode), the effective access time is a logical AND combination of PAGEBURSTACCESSTIME timing completion and the wait-deasserted state. Wait monitoring pipelining is also applicable to multiple accesses (access within a page).

- Wait monitored as active freezes the CYCLETIME counter. For an access within a page, when the CYCLETIME counter is by definition in a lock state, wait monitored as asserted extends the current access time in the page. Control signals are kept in their current state. The data bus is considered invalid, and no data are captured during this clock cycle.
- Wait monitored as inactive unfreezes the CYCLETIME counter. For an access within a page, when the CYCLETIME counter is by definition in a lock state, wait monitored as inactive completes the current access time and starts the next access phase in the page. The data bus is considered valid, and data are captured during this clock cycle. In case of a single access or if this was the last access in a multiple-access cycle, all signals are controlled according to their related control timing value and according to the CYCLETIME counter status.

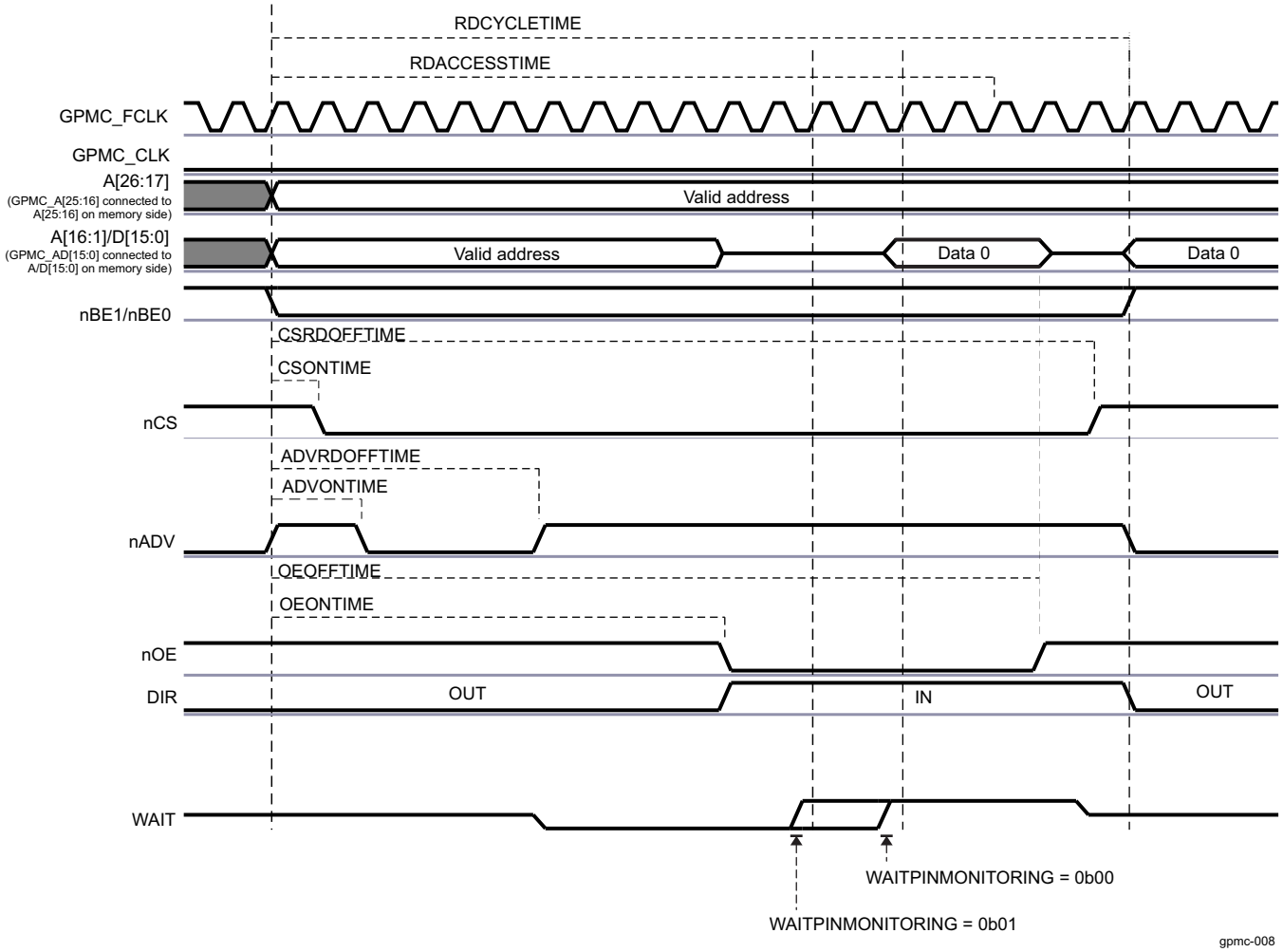
When a delay larger than two GPMC clocks must be observed between wait-pin deactivation time and data valid time (including the required GPMC and the device data setup time), an extra delay can be added between wait-pin deassertion time detection and effective data-capture time and the effective unlock of the CYCLETIME counter. This extra delay can be programmed in the GPMC_CONFIG1_i[19-18] WAITMONITORINGTIME bit field (where i = 0 to 3).

NOTE:

- The WAITMONITORINGTIME parameter does not delay the WAIT pin active or inactive detection, nor does it modify the two GPMC clocks pipelined detection delay.
- This extra delay is expressed as a number of GPMC_CLK clock cycles, even though the access is defined as asynchronous, and no GPMC_CLK clock is provided to the external device. Still, because GPMCFCLKDIVIDER is used as a divider for the GPMC clock, it must be programmed to define the correct WAITMONITORINGTIME delay.

Figure 7-165 shows wait behavior during an asynchronous single read access.

Figure 7-165. Wait Behavior During an Asynchronous Single Read Access (GPMCFCLKDivider = 1)



NOTE: The WAIT signal is active low. GPMC_CONFIG1_i[19-18] WAITMONITORINGTIME = 0b00, or 0b01.

7.3.4.8.3.1.2 Wait Monitoring During Asynchronous Write Access

When WAIT pin monitoring is enabled for write accesses (GPMC_CONFIG1_i[21] WAITWRITEMONITORING bit = 1h), the wait invalid timing window is defined by the WRACCESSTIME field. WRACCESSTIME must be set so that the WAIT pin is at a valid state two GPMC clock cycles before WRACCESSTIME completes. The advance pipelining of the two GPMC clock cycles is the result of the internal synchronization requirements for the WAIT signal.

- Wait monitored as active freezes the CYCLETIME counter. This informs the GPMC that the data bus is not captured by the external device. The control signals are kept in their current state. The data bus still drives the data.
- Wait monitored as inactive unfreezes the CYCLETIME counter. This informs that the data bus is correctly captured by the external device. All signals, including the data bus, are controlled according to their related control timing value and to the CYCLETIME counter status.

When a delay larger than two GPMC clock cycles must be observed between wait-pin deassertion time and the effective data write into the external device (including the required GPMC data setup time and the device data setup time), an extra delay can be added between wait-pin deassertion time detection and effective data write time into the external device and the effective unfreezing of the CYCLETIME counter. This extra delay can be programmed in the GPMC_CONFIG1_i[19-18] WAITMONITORINGTIME bit field (where i = 0 to 3).

NOTE:

- The WAITMONITORINGTIME parameter does not delay the WAIT pin assertion or deassertion detection, nor does it modify the two GPMC clock cycles pipelined detection delay.
- This extra delay is expressed as a number of GPMC_CLK clock cycles, even though the access is defined as asynchronous, and even though no clock is provided to the external device. Still, because the GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER bit field is used as a divider for the GPMC clock, it must be programmed to define the correct WAITMONITORINGTIME delay.

7.3.4.8.3.1.3 Wait Monitoring During Synchronous Read Access

During synchronous accesses with WAIT pin monitoring enabled, the WAIT pin is captured synchronously with GPMC_CLK, using the rising edge of this clock.

The WAIT signal can be programmed to apply to the same clock cycle in which it is captured. Alternatively, it can be sampled one or two GPMC_CLK cycles ahead of the clock cycle to which it applies. This pipelining is applicable to the entire burst access and to all data phases in the burst access. This wait pipelining depth is programmed in the GPMC_CONFIG1_i[19-18] WAITMONITORINGTIME bit field (where i = 0 to 3), and is expressed as a number of GPMC_CLK clock cycles.

In synchronous mode, when WAIT pin monitoring is enabled (the GPMC_CONFIG1_i[22] WAITREADMONITORING bit), the effective access time is a logical AND combination of the RDACCESSTIME timing completion and the wait-deasserted state detection.

Depending on the programmed value of WAITMONITORINGTIME, the WAIT pin must be at a valid level, either asserted or deasserted:

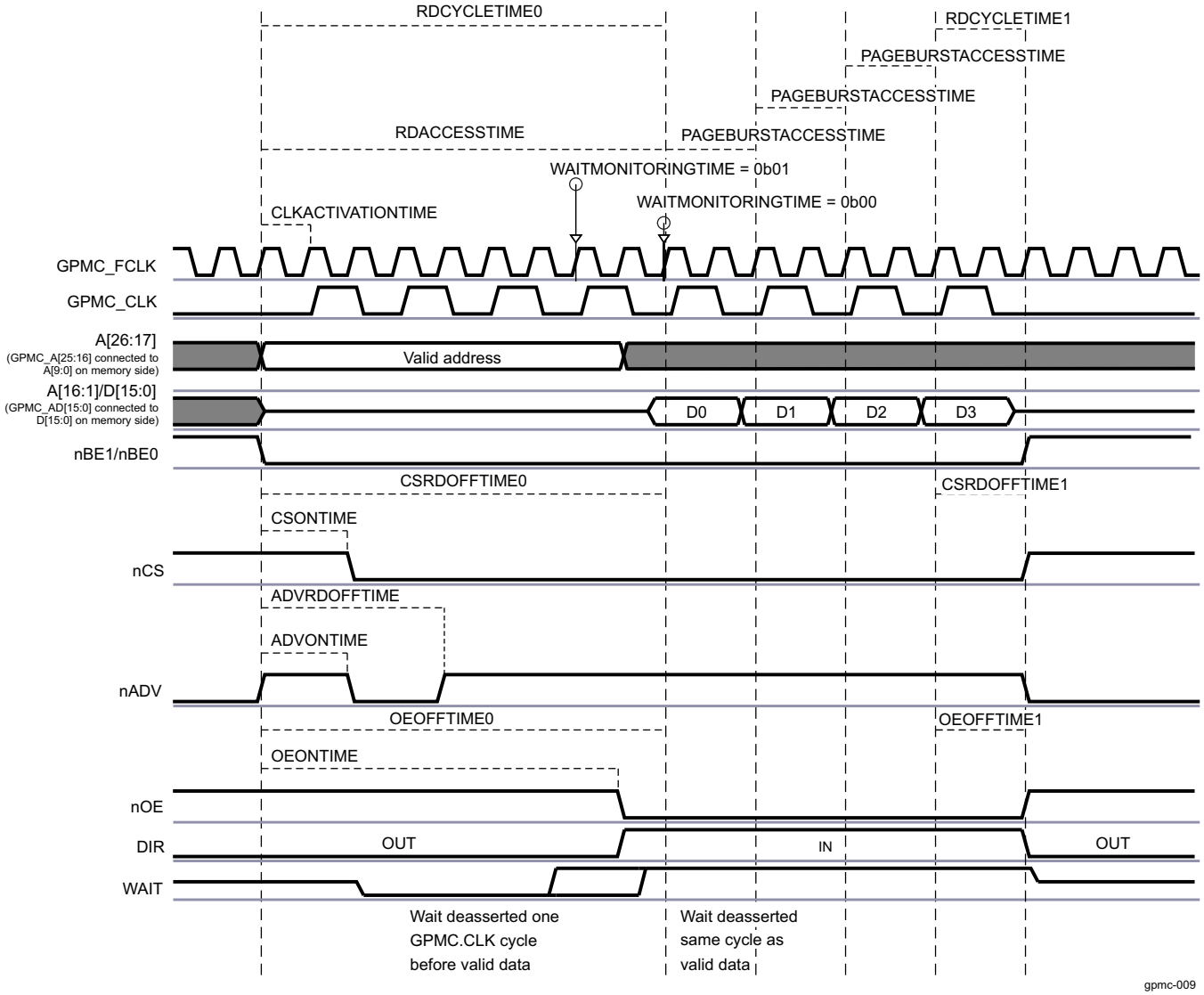
- In the same clock cycle the data is valid if WAITMONITORINGTIME = 0 (at RDACCESSTIME completion)
- In the WAITMONITORINGTIME x (GPMCFCLKDIVIDER + 1) GPMC_FCLK clock cycles before RDACCESSTIME completion if WAITMONITORINGTIME is not equal to 0

Similarly, during a multiple-access cycle (burst mode), the effective access time is a logical AND combination of PAGEBURSTACCESSTIME timing completion and the WAIT-INACTIVE state. The wait pipelining-depth programming applies to the whole burst access.

- Wait monitored as active freezes the CYCLETIME counter. For an access within a burst (when the CYCLETIME counter is by definition in a lock state), wait monitored as active extends the current access time in the burst. Control signals are kept in their current state. The data bus is considered invalid, and no data are captured during this clock cycle.
- Wait monitored as inactive unfreezes the CYCLETIME counter. For an access within a burst (when the CYCLETIME counter is by definition in lock state), wait monitored as inactive completes the current access time and starts the next access phase in the burst. The data bus is considered valid, and data are captured during this clock cycle. In a single access or if this was the last access in a multiple-access cycle, all signals are controlled according to their relative control timing value and the CYCLETIME counter status.

Figure 7-166 shows wait behavior during a synchronous read burst access.

Figure 7-166. Wait Behavior During a Synchronous Read Burst Access



NOTE: The WAIT signal is active low. WAITMONITORINGTIME = 00, 01.

7.3.4.8.3.1.4 Wait Monitoring During Synchronous Write Access

During synchronous accesses with WAIT pin monitoring enabled (the WAITWRITEMONITORING bit), the WAIT pin is captured synchronously with GPMC_CLK, using the rising edge of this clock.

If enabled, external WAIT pin monitoring can be used in combination with WRACCESSTIME to delay the GPMC_CLK capture edge of the effective memory device.

WAIT-monitoring pipelining depth is similar to synchronous read access:

- At WRACCESSTIME completion if WAITMONITORINGTIME = 0
- In the WAITMONITORINGTIME x (GPMCFCLKDIVIDER + 1) GPMC_FCLK cycles before WRACCESSTIME completion if WAITMONITORINGTIME is not equal to 0

Wait-monitoring pipelining definition applies to whole burst accesses:

- Wait monitored as active freezes the CYCLETIME counter. For accesses within a burst, when the CYCLETIME counter is by definition in a lock state, wait monitored as active indicates that the data

bus is not being captured by the external device. Control signals are kept in their current state. The data bus is kept in its current state.

- Wait monitored as inactive unfreezes the CYCLETIME counter. For accesses within a burst, when the CYCLETIME counter is by definition in a lock state, wait monitored as inactive indicates the effective data capture of the bus by the external device and starts the next access of the burst. In case of a single access or if this was the last access in a multiple access cycle, all signals, including the data bus, are controlled according to their related control timing value and the CYCLETIME counter status.

NOTE: WAIT monitoring is supported for all configurations except GPMC_CONFIG1_i[19-18] WAITMONITORINGTIME = 0h (where i = 0 to 3) for write bursts with a clock divider of 1 or 2 (the GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER bit field is equal to 0h or 1h, respectively).

7.3.4.8.3.1.5 Wait With NAND Device

For information about the use of the WAIT pin for communication with a NAND flash external device, see Section 7.3.4.12.2, *NAND Device-Ready Pin*.

7.3.4.8.3.1.6 Idle Cycle Control Between Successive Accesses

7.3.4.8.3.1.6.1 Bus Turnaround (BUSTURNAROUND)

To prevent data-bus contention, an access that follows a read access to a slow memory/device must be delayed (in other words, control the nCS/nOE deassertion to data bus in high-impedance delay).

The bus turnaround is a time-out counter starting after nCS or nOE deassertion time, whichever occurs first, and delays the next access start-cycle time. The counter is programmed through the GPMC_CONFIG6_i[3-0] BUSTURNAROUND bit field (where i = 0 to 3).

After a read access to a chip-select with a nonzero BUSTURNAROUND, the next access is delayed until the BUSTURNAROUND delay completes, if the next access is one of the following:

- A write access to any chip-select (the same or different chip-select from which the data was read)
- A read access to a different chip-select than the chip-select from which the data was read access
- A read or write access to a chip-select associated with an address/data-multiplexed device

NOTE: Bus keeping starts after bus turnaround completion so that DIR changes from IN to OUT after bus turnaround. The bus does not have enough time to go into high-impedance even though it can be driven with the same value before bus turnaround timing.

BUSTURNAROUND delay runs in parallel with GPMC_CONFIG6_i[3-0] CYCLE2CYCLEDELAY bit field delays. BUSTURNAROUND is a timing parameter for the ending chip-select access, while CYCLE2CYCLEDELAY is a timing parameter for the following chip-select access. The effective minimum delay between successive accesses is driven by these delay timing parameters and by the access type of the following access (see Figure 7-167 through Figure 7-169).

Another way to prevent bus contention is to define an earlier nCS or nOE deassertion time for slow devices or to extend the value of RDCYCLETIME. Doing this prevents bus contention, but it also affects all accesses of this specific chip-select.

Figure 7-167. Read-to-Read for an Address-Data Multiplexed Device, on Different Chip-Select, Without Bus Turnaround (nCS Attached to a Fast Device)

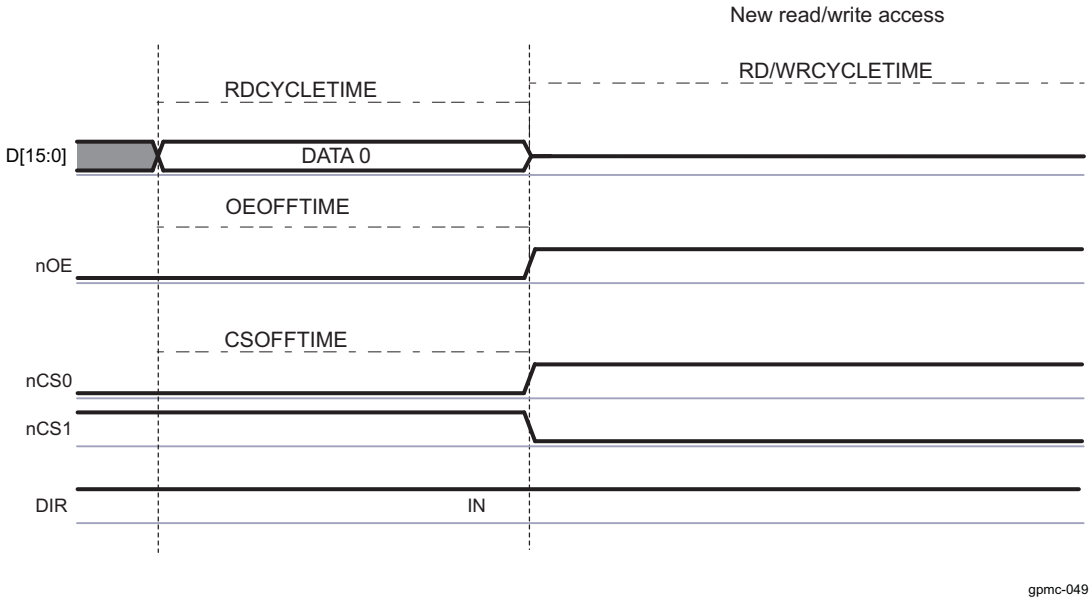


Figure 7-168. Read- to-Read/Write for an Address-Data Multiplexed Device, on Different Chip-Select, With Bus Turnaround

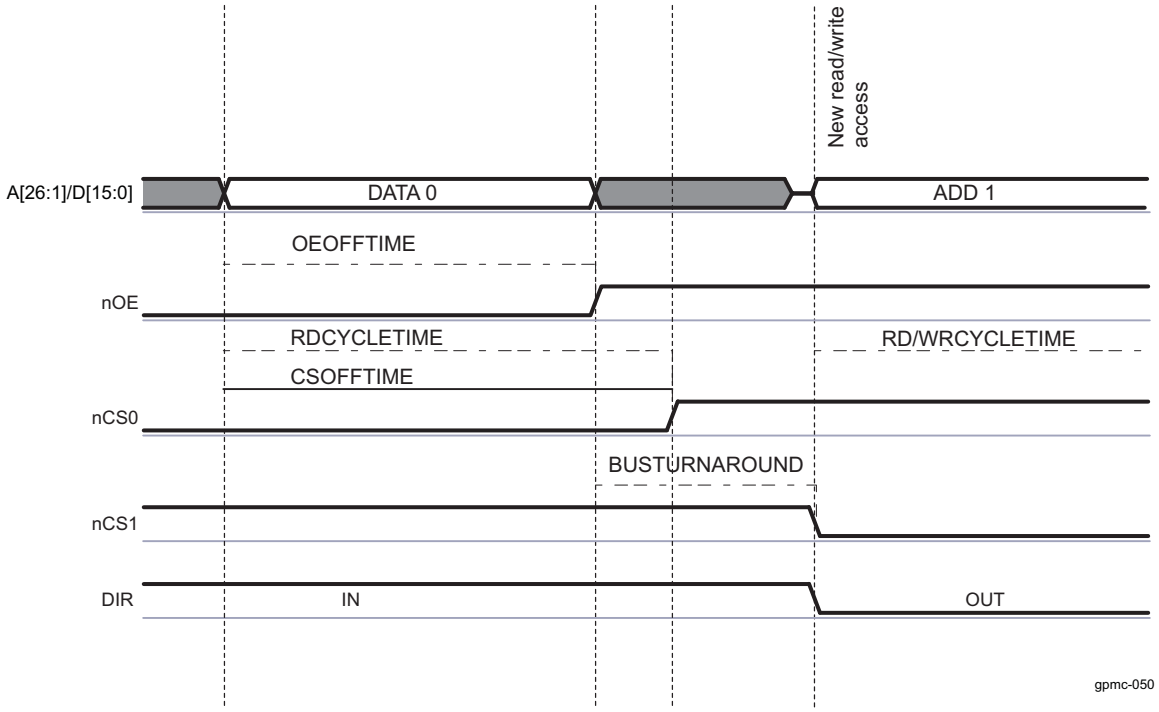
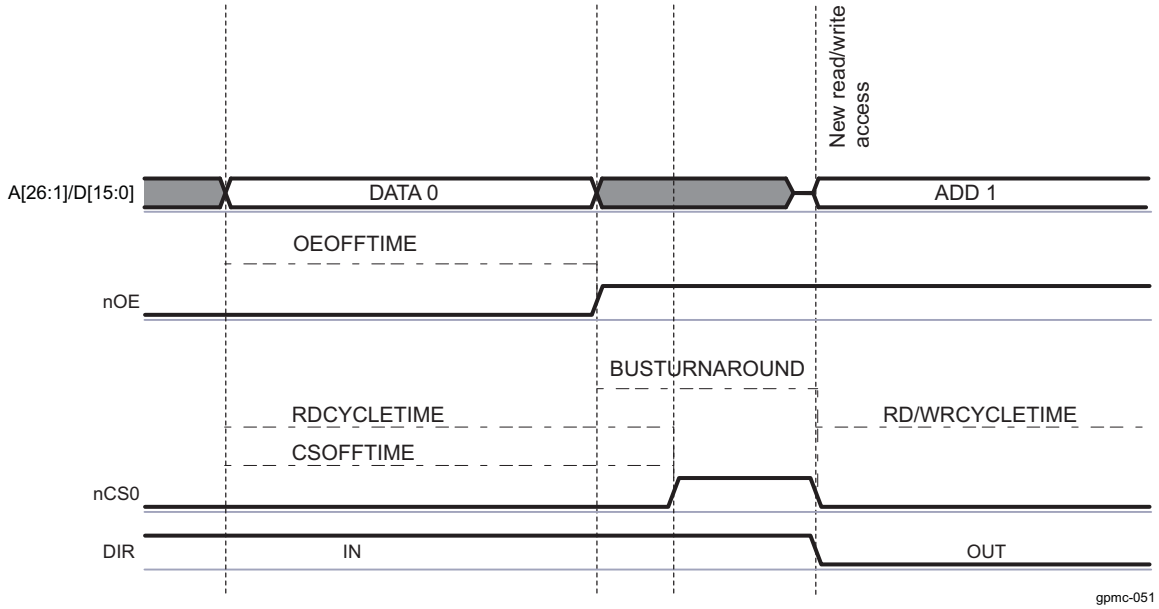


Figure 7-169. Read-to-Read/Write for a Address-Data or AAD-Multiplexed Device, on Same Chip-Select, With Bus Turnaround



7.3.4.8.3.1.6.2 Idle Cycles Between Accesses to Same Chip-Select (CYCLE2CYCLESAMECSEN, CYCLE2CYCLEDELAY)

Some devices require a minimum chip-select signal inactive time between accesses. The GPMC_CONFIG6_i[7] CYCLE2CYCLESAMECSEN bit (where i = 0 to 3) enables insertion of a minimum number of GPMC_FCLK cycles, defined by the GPMC_CONFIG6_i[11-8] CYCLE2CYCLEDELAY bit field, between successive accesses of any type (read or write) to the same chip-select.

If CYCLE2CYCLESAMECSEN is enabled, any subsequent access to the same chip-select is delayed until its CYCLE2CYCLEDELAY completes. The CYCLE2CYCLEDELAY counter starts when CSRDOFFTIME/CSWROFFTIME completes.

The same applies to successive accesses occurring during 32-bit word or burst accesses split into successive single accesses when the single-access mode is used (GPMC_CONFIG1_i[30] READMULTIPLE = 0 or GPMC_CONFIG1_i[28] WRITEMULTIPLE = 0).

All control signals (CS, ADV#/ALE, BE0#/CLE, WE#, and GPMC.CLK) are kept inactive (ADV#/ALE, BE0#/CLE, and GPMC.CLK at low level; and CS, OE#/RE, and WE at high level) during the idle GPMC.FCLK cycles. This prevents back-to-back accesses to the same chip-select without idle cycles between accesses.

7.3.4.8.3.1.6.3 Idle Cycles Between Accesses to Different Chip-Select (CYCLE2CYCLEDIFFCSEN, CYCLE2CYCLEDELAY)

Because of the pipelined behavior of the system, successive accesses to different chip-selects can occur back-to-back with no idle cycles between accesses. Depending on the control signals (nCS, nADV/ALE, nBE0/CLE, nOE/RE, nWE) assertion and deassertion timing parameters and on the device timing parameters, the assertion times of some control signals may overlap between the successive accesses to a different chip-select. Similarly, some control signals (WE, OE/RE) may not respect required transition times.

To work around overlapping and to observe the required control-signal transitions, a minimum of CYCLE2CYCLEDELAY inactive cycles is inserted between the access being initiated to this chip-select and the previous access ending for a different chip-select. This applies to any type of access (read or write).

If the GPMC_CONFIG6_i[6] CYCLE2CYCLEDIFFCSEN bit is enabled, the chip-select access is delayed until CYCLE2CYCLEDELAY cycles have expired since the end of a previous access to a different chip-select. CYCLE2CYCLEDELAY count starts at CSRDOFFTIME/CSWROFFTIME completion. All control signals are kept inactive during the idle GPMC_FCLK cycles.

NOTE: CYCLE2CYCLESAMECSEN and CYCLE2CYCLEDIFFCSEN must be set in the GPMC_CONFIG6_i registers to get idle cycles inserted between accesses on this chip-select and after accesses to a different chip-select, respectively.

The CYCLE2CYCLEDELAY delay runs in parallel with the BUSTURNAROUND delay. The BUSTURNAROUND is a timing parameter defined for the ending chip-select access, whereas CYCLE2CYCLEDELAY is a timing parameter defined for the starting chip-select access. The effective minimum delay between successive accesses is based on the larger delay timing parameter and on access type combination, because bus turnaround does not apply to all access types. For more information about bus turnaround, see Section 7.3.4.8.3.1.6.1, *Bus Turnaround (BUSTURNAROUND)*.

Table 7-471 describes the configuration required for idle cycle insertion.

Table 7-471. Idle Cycle Insertion Configuration

1st Access Type	BUSTURNAROUND Timing Parameter	Second Access Type	Chip-Select	Add/Data Multiplexed	CYCLE2CYCLE SAMECSEN Parameter	CYCLE2CYCLE DIFFCSEN Parameter	Idle Cycle Insertion Between the Two Accesses
R/W	= 0	R/W	Any	Any	0	x	No idle cycles are inserted if the two accesses are well pipelined.
R	> 0	R	Same	Nonmuxed	x	0	No idle cycles are inserted if the two accesses are well pipelined.
R	> 0	R	Different	Nonmuxed	0	0	BUSTURNAROUND cycles are inserted.
R	> 0	R/W	Any	Muxed	0	0	BUSTURNAROUND cycles are inserted.
R	> 0	W	Any	Any	0	0	BUSTURNAROUND cycles are inserted.
W	> 0	R/W	Any	Any	0	0	No idle cycles are inserted if the two accesses are well pipelined.
R/W	= 0	R/W	Same	Any	1	x	CYCLE2CYCLEDELAY cycles are inserted.
R/W	= 0	R/W	Different	Any	x	1	CYCLE2CYCLEDELAY cycles are inserted.
R/W	> 0	R/W	Same	Any	1	x	CYCLE2CYCLEDELAY cycles are inserted. If BTA idle cycles already apply on these two back-to-back accesses, the effective delay is max (BUSTURNAROUND, CYCLE2CYCLEDELAY).
R/W	> 0	R/W	Different	Any	x	1	CYCLE2CYCLEDELAY cycles are inserted. If BTA idle cycles already apply on these two back-to-back accesses, the effective delay is maximum (BUSTURNAROUND, CYCLE2CYCLEDELAY).

7.3.4.8.3.1.7 Slow Device Support (TIMEPARAGRANULARITY Parameter)

All access-timing parameters can be multiplied by 2 by setting the GPMC_CONFIG1_i[4] TIMEPARAGRANULARITY bit (where i stands for the GPMC chip-select i, where i = 0 to 3). Increasing all access timing parameters allows support of slow devices.

7.3.4.8.3.2 DIR Pin

The DIR pin is used to control I/O direction on the GPMC data bus GPMC_D[15:0]. Depending on pad multiplexing, this signal can be output and used externally to the device, if required. The DIR pin is low during transmit (OUT) and high during receive (IN).

For write accesses, the DIR pin stays OUT from start-cycle time to end-cycle time.

For read accesses, the DIR pin goes from OUT to IN at nOE assertion time and stays IN until:

- BUSTURNAROUND is enabled
  - The DIR pin goes from IN to OUT at end-cycle time plus programmable bus turnaround time.
- BUSTURNAROUND is disabled
  - After an asynchronous read access, the DIR pin goes from IN to OUT at RDACCESSTIME + 1 GPMC_FCLK cycle or when RDCYCLETIME completes, whichever occurs last.
  - After a synchronous read access, the DIR pin goes from IN to OUT at RDACCESSTIME + 2 GPMC_FCLK cycles or when RDCYCLETIME completes, whichever occurs last.

Because of the bus-keeping feature of the GPMC, after a read or write access and with no other accesses pending, the default value of the DIR pin is OUT (see Section 7.3.4.9.10). In non-multiplexed devices, the DIR pin stays IN between two successive read accesses to prevent unnecessary toggling.

7.3.4.8.3.3 Reset

No reset signal is sent to the external memory device by the GPMC.

GPMC_RST is the reset signal for the GPMC module. It is connected and controlled by LPSC9. That reset signal initializes the internal state-machine and the internal configuration registers.

7.3.4.8.3.4 Write Protect Signal (nWP)

When connected to the attached memory device, the write protect signal can enable or disable the lockdown function of the attached memory. The nWP output pin value is controlled through the GPMC_CONFIG[4] WRITEPROTECT bit which is common for all chip selects.

7.3.4.8.3.5 Byte Enable (nBE1/nBE0)

Byte enable signals (nBE1/nBE0) are:

- Valid (asserted or nonasserted according to the incoming system request) from access start to access completion for asynchronous and synchronous single accesses
- Asserted low from access start to access completion for asynchronous and synchronous multiple read accesses
- Valid (asserted or nonasserted, according to the incoming system request) synchronously to each written data for synchronous multiple write accesses

7.3.4.8.4 Error Handling

When an error occurs in the GPMC, the error information is stored in the GPMC_ERR_TYPE register and the address of the illegal access is stored in the GPMC_ERR_ADDRESS register. The GPMC keeps only the first error abort information until the GPMC_ERR_TYPE register is reset. Subsequent accesses that cause errors are not logged until the error is cleared by hardware with the GPMC_ERR_TYPE[0] ERRORVALID bit.

- ERRORNOTSUPPADD occurs when an incoming system request address decoding does not match any valid chip-select region, or if two chip-select regions are defined as overlapped, or if a register file access is tried outside the valid address range of 1KB.
- ERRORNOTSUPPMCMD occurs when an unsupported command request is decoded at the interconnect interface.
- ERRORTIMEOUT: A time-out mechanism prevents the system from hanging. The start value of the 9-bit time-out counter is defined in the GPMC_TIMEOUT_CONTROL register and enabled with the GPMC_TIMEOUT_CONTROL[0] TIMEOUTENABLE bit. When enabled, the counter starts at start-

cycle time until it reaches 0 and data is not responded to from memory, and then a time-out error occurs. When data are sent from memory, this counter is reset to its start value. With multiple accesses (asynchronous page mode or synchronous burst mode), the counter is reset to its start value for each data access within the burst.

The GPMC does not generate interrupts on these errors. An interrupt generation is handled at interconnect level.

7.3.4.9 Timing Setting

The GPMC offers maximum flexibility to support various access protocols. Most of the timing parameters of the protocol access used by the GPMC to communicate with attached memories or devices are programmable on a chip-select basis. Assertion and deassertion times of control signals are defined to match the attached memory or device timing specifications and to get maximum performance during accesses. For more information about GPMC_CLK and GPMC_FCLK see Section 7.3.4.9.6, *GPMC_CLK*.

NOTE: In the following sections, the start access time refers to the time at which the access begins.

7.3.4.9.1 Read Cycle Time and Write Cycle Time (RDCYCLETIME / WRCYCLETIME)

The GPMC_CONFIG5_i[4-0] RDCYCLETIME and GPMC_CONFIG5_i[12-8] WRCYCLETIME bit fields (where i = 0 to 3) define the address bus and byte-enable valid times for read and write accesses. To ensure a correct duty cycle of GPMC_CLK between accesses, RDCYCLETIME and WRCYCLETIME are expressed in GPMC_FCLK cycles and must be multiples of the GPMC_CLK cycle. The RDCYCLETIME and WRCYCLETIME bit fields can be set with a granularity of 1 or 2 through the GPMC_CONFIG1_i[4] TIMEPARAGRANULARITY bit.

When RDCYCLETIME or WRCYCLETIME completes, if they are not already deasserted, all control signals (nCS, nADV/ALE, nOE/RE, nWE, and BE0/CLE) are deasserted to their reset values, regardless of their deassertion time parameters.

An exception to this forced deassertion occurs when a pipelined request to the same chip-select or to a different chip-select is pending. In such a case, it is not necessary to deassert a control signal with deassertion time parameters equal to the cycle-time parameter. This exception to forced deassertion prevents any unnecessary glitches. This requirement also applies to BE signals, thus avoiding an unnecessary BE glitch transition when pipelining requests.

NOTE: All control signals (CS, ADV#/ALE, BE0#/CLE, WE#, and GPMC.CLK) are kept inactive (ADV#/ALE, BE0#/CLE, and GPMC.CLK at low level; and CS, OE#/RE, and WE at high level) during the idle GPMC.FCLK cycles.

If no inactive cycles are required between successive accesses to the same chip-select or a different chip-select (GPMC_CONFIG6_i[7] CYCLE2CYCLESAMECSEN = 0 or GPMC_CONFIG6_i[6] CYCLE2CYCLEDIFFCSEN = 0, where i = 0 to 3), and if assertion-time parameters associated with the pipelined access are equal to 0, asserted control signals (nCS, nADV/ALE, nBE0/CLE, nWE, and nOE/RE) are kept asserted. This applies to any read/write to read/write access combination.

If inactive cycles are inserted between successive accesses (that is, CYCLE2CYCLESAMECSEN = 1 or CYCLE2CYCLEDIFFCSEN = 1), the control signals are forced to their respective default reset values for the number of GPMC_FCLK cycles defined in CYCLE2CYCLEDELAY.

7.3.4.9.2 nCS: Chip-Select Signal Control Assertion/Deassertion Time (CSONTIME / CSRDOFFTIME / CSWROFFTIME / CSEXTRADELAY)

The GPMC_CONFIG2_i[3-0] CSONTIME bit field (where i = 0 to 3) defines the nCS signal-assertion time relative to the start access time. It is common for read and write accesses.

The GPMC_CONFIG2_i[12-8] CSRDOFFTIME (read access) and GPMC_CONFIG2_i[20-16] CSWROFFTIME (write access) bit fields define the nCS signal deassertion time relative to start access time.

The CSONTIME, CSRDOFFTIME, and CSWROFFTIME parameters apply to synchronous and asynchronous modes. CSONTIME can be used to control an address and byte-enable setup time before chip-select assertion. CSRDOFFTIME and CSWROFFTIME can be used to control an address and byte-enable hold time after chip-select deassertion.

nCS signal transitions, as controlled through CSONTIME, CSRDOFFTIME, and CSWROFFTIME, can be delayed by a half-GPMC_FCLK period by enabling the GPMC_CONFIG2_i[7] CSEXTRADELAY bit. This half-GPMC_FCLK period provides more granularity on the nCS assertion and deassertion time to ensure proper setup and hold time relative to GPMC_CLK. CSEXTRADELAY is especially useful in configurations where GPMC_CLK and GPMC_FCLK have the same frequency, but it can also be used for all GPMC configurations. If enabled, CSEXTRADELAY applies to all parameters that control nCS transitions.

The CSEXTRADELAY bit must be used carefully to avoid control signal overlap between successive accesses to different chip-selects. This implies the need to program the RDCYCLETIME and WRCYCLETIME bit fields to be greater than the nCS signal-deassertion time, including the extra half-GPMC_FCLK-period delay.

7.3.4.9.3 nADV/ALE: Address Valid/Address Latch Enable Signal Control Assertion/Deassertion Time (ADVONTIME / ADVRDOFFTIME / ADVWROFFTIME / ADVEXTRADELAY/ADVAADMUXONTIME/ADVAADMUXRDOFFTIME/ADVAADMUXWROFFTIME)

The GPMC_CONFIG3_i[3-0] ADVONTIME field (where i = 0 to 3) defines the nADV/ALE signal-assertion time relative to start access time. It is common to read and write accesses.

The GPMC_CONFIG3_i[12-8] ADVRDOFFTIME (read access) and GPMC_CONFIG3_i[20-16] ADVWROFFTIME (write access) bit fields define the nADV/ALE signal-deassertion time relative to start access time.

ADVONTIME can be used to control an address and byte-enable valid setup time control before nADV/ALE assertion. ADVRDOFFTIME and ADVWROFFTIME can be used to control an address and byte-enable valid hold time control after nADV/ALE deassertion. ADVRDOFFTIME and ADVWROFFTIME apply to synchronous and asynchronous modes.

The nADV/ALE signal transitions as controlled through ADVONTIME, ADVRDOFFTIME, and ADVWROFFTIME can be delayed by a half-GPMC_FCLK period by enabling the GPMC_CONFIG3_i[7] ADVEXTRADELAY bit. This half-GPMC_FCLK period provides more granularity on nADV/ALE assertion and deassertion time to ensure proper setup and hold time relative to GPMC_CLK. The ADVEXTRADELAY configuration parameter is especially useful in configurations where GPMC_CLK and GPMC_FCLK have the same frequency, but can be used for all GPMC configurations. If enabled, ADVEXTRADELAY applies to all parameters controlling nADV/ALE transitions.

ADVEXTRADELAY must be used carefully to avoid control-signal overlap between successive accesses to different chip-selects. This implies the need to program the RDCYCLETIME and WRCYCLETIME bit fields to be greater than nADV/ALE signal-deassertion time, including the extra half-GPMC_FCLK-period delay.

GPMC_CONFIG3_i[6-4] ADVAADMUXONTIME, GPMC_CONFIG3_i[26-24] ADVAADMUXRDOFFTIME, and GPMC_CONFIG3_i[30-28] ADVAADMUXWROFFTIME parameters have the same functions as ADVONTIME, ADVRDOFFTIME, and ADVWROFFTIME, but apply to the first address phase in the AAD-multiplexed protocol. The user must ensure that ADVAADMUXxxOFFTIME is programmed to a value less than or equal to ADVxxOFFTIME. Functionality in AAD-multiplexed mode is undefined if the settings do not comply with this requirement. ADVAADMUXxxOFFTIME can be programmed to the same value as ADVONTIME if no high nADV pulse is needed between the two AAD-multiplexed address phases, which is the typical case in synchronous mode. In this configuration, nADV is kept low until it reaches the correct ADVxxOFFTIME.

For more information about the use of ADVONTIME, ADVRDOFFTIME, ADVWROFFTIME, and ADVAADMUXRDOFFTIME and ADVAADMUXWROFFTIME for command latch enable (CLE) and address latch enable (ALE) use for a NAND flash interface, see Section 7.3.4.12, *NAND Access Description*.

7.3.4.9.4 nOE/nRE: Output Enable/Read Enable Signal Control Assertion/Deassertion Time (OEONTIME /

OEOFFTIME / OEEXTRADELAY / OEAADMUXONTIME / OEAADMUXOFFTIME)

The GPMC_CONFIG4_i[3-0] OEONTIME bit field (where i = 0 to 3) defines the nOE/nRE signal assertion time relative to start access time. It applies only to read accesses.

The GPMC_CONFIG4_i[12-8] OEOFFTIME bit field defines the nOE/nRE signal deassertion time relative to start access time. It applies only to read accesses. nOE/nRE is not asserted during a write cycle.

The OEONTIME, OEOFFTIME, OEAADMUXONTIME, and OEAADMUXOFFTIME parameters apply to synchronous and asynchronous modes. OEONTIME can be used to control an address and byte enable valid setup time control before nOE/nRE assertion. OEOFFTIME can be used to control an address and byte-enable valid hold time control after nOE/nRE assertion.

The OEAADMUXONTIME and OEAADMUXOFFTIME parameters have the same functions as OEONTIME and OEOFFTIME, but apply to the first OE assertion in the AAD-multiplexed protocol for a read phase, or to the only OE assertion for a write phase. The user mut ensure that OEAADMUXOFFTIME is programmed to a value less than OEONTIME. Functionality in AAD-multiplexed mode is undefined if the settings do not comply with this requirement. OEAADMUXOFFTIME must never be equal to OEONTIME because the AAD-multiplexed protocol requires a second address phase with the nOE signal deasserted before nOE can be asserted again to define a read command.

The nOE/RE signal transitions as controlled through OEONTIME, OEOFFTIME, OEAADMUXONTIME, and OEAADMUXOFFTIME can be delayed by a half-GPMC_FCLK period by enabling the GPMC_CONFIG4_i[7] OEEXTRADELAY bit. This half-GPMC_FCLK period provides more granularity on the nOE/RE assertion and deassertion time to ensure proper setup and hold time relative to GPMC_CLK. If enabled, OEEXTRADELAY applies to all parameters controlling nOE/nRE transitions.

OEEXTRADELAY must be used carefully, to avoid control-signal overlap between successive accesses to different chip-selects. This implies the need to program RDCYCLETIME and WRCYCLETIME to be greater than the nOE/RE signal-deassertion time, including the extra half-GPMC_FCLK-period delay.

NOTE: When the GPMC generates a read access to an address-/data-multiplexed device, it drives the address bus until nOE assertion time.

7.3.4.9.5 nWE: Write Enable Signal Control Assertion/Deassertion Time (WEONTIME / WEOFFTIME / WEEXTRADELAY)

The GPMC_CONFIG4_i[19-16] WEONTIME bit field (where i = 0 to 3) defines the nWE signal-assertion time relative to start access time. The GPMC_CONFIG4_i[28-24] WEOFFTIME bit field defines the nWE signal-deassertion time relative to start access time. These bit fields apply only to write accesses. nWE is not asserted during a read cycle.

WEONTIME can be used to control an address and byte-enable valid setup time control before nWE assertion. WEOFFTIME can be used to control an address and byte-enable valid hold time control after nWE assertion.

nWE signal transitions as controlled through WEONTIME, and WEOFFTIME can be delayed by a half-GPMC_FCLK period by enabling the GPMC_CONFIG4_i[23] WEEXTRADELAY bit. This half-GPMC_FCLK period provides more granularity on nWE assertion and deassertion time to ensure proper setup and hold time relative to GPMC_CLK. If enabled, WEEXTRADELAY applies to all parameters controlling nWE transitions.

The WEEXTRADELAY bit must be used carefully to avoid control-signal overlap between successive accesses to different chip-selects. This implies the need to program the WRCYCLETIME bit field to be greater than the nWE signal-deassertion time, including the extra half-GPMC_FCLK-period delay.

7.3.4.9.6 GPMC_CLK

GPMC_CLK is the external clock provided to the attached synchronous memory or device.

- The GPMC_CLK clock frequency is the GPMC_FCLK functional clock frequency divided by 1, 2, 3, or 4, depending on the GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER bit field (where i = 0 to 3), with a guaranteed 50-percent duty cycle. For information about the duty cycle error, see the device Data Manual.

- The GPMC_CLK clock is activated only when the access in progress is defined as synchronous (read or write access).
- The GPMC_CONFIG1_i[26-25] CLKACTIVATIONTIME bit field (where i = 0 to 3) defines the number of GPMC_FCLK cycles from start access time to GPMC_CLK activation.
- The GPMC_CLK clock is stopped when cycle time completes and is asserted low between accesses.
- The GPMC_CLK clock is kept low when access is defined as asynchronous.

CAUTION

When the cycle time completes, the GPMC_CLK may be high because of the GPMCFCLKDIVIDER bit field. To ensure correct stoppage of the GPMC_CLK clock within the required 50-percent duty cycle, the user must extend the RDCYCLETIME or WRCYCLETIME value.

NOTE: To ensure a correct external clock cycle, the following rules must be applied:

- (RDCYCLETIME CLKACTIVATIONTIME) must be a multiple of (GPMCFCLKDIVIDER + 1).
- The PAGEBURSTACCESSTIME value must be a multiple of (GPMCFCLKDIVIDER + 1).

7.3.4.9.7 GPMC_CLK and Control Signals Setup and Hold

Control-signal transition (assertion and deassertion) setup and hold values with respect to the GPMC_CLK edge can be controlled in the following ways:

- For the GPMC_CLK signal, the GPMC_CONFIG1_i[26-25] CLKACTIVATIONTIME bit field (where i = 0 to 3) allows setup and hold control of control-signal assertion time.
- The use of a divided GPMC_CLK allows setup and hold control of the control-signal assertion and deassertion times.
- When GPMC_CLK runs at the GPMC_FCLK frequency so that GPMC_CLK edge and control-signal transitions refer to the same GPMC_FCLK edge, the control-signal transitions can be delayed by a half-GPMC_FCLK period to provide minimum setup and hold times. This half-GPMC_FCLK delay is enabled with the CSEXTRADELAY, ADVEXTRADELAY, OEEXTRADELAY, or WEEXTRADELAY parameter. This delay must be used carefully to prevent control-signal overlap between successive accesses to different chip-selects. This implies that the RDCYCLETIME and WRCYCLETIME are greater than the last control-signal deassertion time, including the extra half-GPMC_FCLK cycle.

7.3.4.9.8 Access Time (RDACCESSTIME / WRACCESSTIME)

The read/write access time durations can be programmed independently through the GPMC_CONFIG5_i[20-16] RDACCESSTIME and GPMC_CONFIG6_i[28-24] WRACCESSTIME bit fields (where i = 0 to 3). This allows nOE and GPMC data-capture timing parameters to be independent of nWE and memory device data capture timing parameters. The RDACCESSTIME and WRACCESSTIME bit fields can be set with a granularity of 1 or 2 through the GPMC_CONFIG1_i[4] TIMEPARAGRANULARITY bit.

7.3.4.9.8.1 Access Time on Read Access

In asynchronous read mode, for single and paged accesses, the GPMC_CONFIG5_i[20-16] RDACCESSTIME bit field (where i = 0 to 3) defines the number of GPMC_FCLK cycles from start access time to the GPMC_FCLK rising edge used for the first data capture. RDACCESSTIME must be programmed to the rounded greater value (in GPMC_FCLK cycles) of the read access time of the attached memory device.

In synchronous read mode, for single or burst accesses, RDACCESSTIME defines the number of GPMC_FCLK cycles from the start access time to the GPMC_FCLK rising edge corresponding to the GPMC_CLK rising edge used for the first data capture.

GPMC_CLK, which is sent to the memory device for synchronization with the GPMC controller, is internally retimed to correctly latch the returned data. The GPMC_CONFIG5_i[4-0] RDCYCLETIME bit field must be greater than RDACCESSTIME to let the GPMC latch the last return data using the internally retimed GPMC_CLK.

The external WAIT signal can be used in conjunction with RDACCESSTIME to control the effective GPMC data-capture GPMC_FCLK edge on read access in asynchronous and synchronous modes. For more information about wait monitoring, see Section 7.3.4.8.3.1, *WAIT Pin Monitoring Control*.

7.3.4.9.8.2 Access Time on Write Access

In asynchronous write mode, the GPMC_CONFIG6_i[28-24] WRACCESSTIME timing parameter is not used to define the effective write access time. Instead, it is used as a wait invalid timing window and must be set to a correct value so that the GPMC_WAIT pin is at a valid state two GPMC_CLK cycles before WRACCESSTIME completes. For more information about wait monitoring, see Section 7.3.4.8.3.1, *WAIT Pin Monitoring Control*.

In synchronous write mode , for single or burst accesses, WRACCESSTIME defines the number of GPMC_FCLK cycles from the start access time to the GPMC_CLK rising edge used by the memory device for the first data capture.

The external WAIT signal can be used in conjunction with WRACCESSTIME to control the effective memory device data-capture GPMC_CLK edge for a synchronous write access. For more information about wait monitoring, see Section 7.3.4.8.3.1, *WAIT Pin Monitoring Control*.

7.3.4.9.9 Page Burst Access Time (PAGEBURSTACCESSTIME)

The GPMC_CONFIG5_i[27-24] PAGEBURSTACCESSTIME bit field (where i = 0 to 3) can be set with a granularity of 1 or 2 through theGPMC_CONFIG1_i[4] TIMEPARAGRANULARITY bit.

7.3.4.9.9.1 Page Burst Access Time on Read Access

In asynchronous page read mode, the delay between successive word captures in a page is controlled through the PAGEBURSTACCESSTIME bit field. The PAGEBURSTACCESSTIME parameter must be programmed to the rounded greater value (in GPMC_FCLK cycles) of the read access time of the attached device.

In synchronous burst read mode, the delay between successive word captures in a burst is controlled through the PAGEBURSTACCESSTIME bit field.

The external WAIT signal can be used in conjunction with PAGEBURSTACCESSTIME to control the effective GPMC data-capture GPMC_FCLK edge on read access. For more information about wait monitoring, see Section 7.3.4.8.3.1, *WAIT Pin Monitoring Control*.

7.3.4.9.9.2 Page Burst Access Time on Write Access

Asynchronous page write mode is not supported. PAGEBURSTACCESSTIME is irrelevant in this case.

In synchronous burst write mode, PAGEBURSTACCESSTIME controls the delay between successive memory device word captures in a burst.

The external WAIT signal can be used in conjunction with PAGEBURSTACCESSTIME to control the effective memory device data capture GPMC_CLK edge in synchronous write mode. For more information about wait monitoring, see Section 7.3.4.8.3.1, *WAIT Pin Monitoring Control*.

7.3.4.9.10 Bus Keeping Support

At the end cycle time of a read access, if no other access is pending, the GPMC drives the bus with the last data read after RDCYCLETIME completes to prevent bus floating and reduce power consumption.

After a write access, if no other access is pending, the GPMC keeps driving the data bus after WRCYCLETIME completes with the same data to prevent bus floating and power consumption.

7.3.4.10 NOR Access Description

For each chip-select configuration, the read access can be specified as asynchronous or synchronous access through the GPMC_CONFIG1_i[29] READTYPE bit (where i = 0 to 3). For each chip-select configuration, the write access can be specified as synchronous or asynchronous access through the GPMC_CONFIG1_i[27] WRITETYPE bit where (i = 0 to 3).

Asynchronous and synchronous read and write access time and related control signals are controlled through timing parameters that refer to GPMC_FCLK. The primary difference of synchronous mode is the availability of a configurable clock interface (GPMC_CLK) to control the external device. Synchronous mode also affects data-capture and wait-pin monitoring schemes in read access.

For more information about asynchronous and synchronous access, see the descriptions of GPMC_CLK, RdAccessTime, WrAccessTime, and WAIT pin monitoring.

For more information about timing-parameter settings, see the sample timing diagrams in this chapter.

NOTE: The address bus and nBE[1:0] are fixed for the duration of a synchronous burst read access, but they are updated for each beat of an asynchronous page-read access.

7.3.4.10.1 Asynchronous Access Description

This section describes:

- Asynchronous single-read operation on an address/data multiplexed device
- Asynchronous single write operation on an address/data-multiplexed device
- Asynchronous single read operation on an AAD-multiplexed device
- Asynchronous single write operation on an AAD-multiplexed device
- Asynchronous multiple (page) read operation on a non-multiplexed device

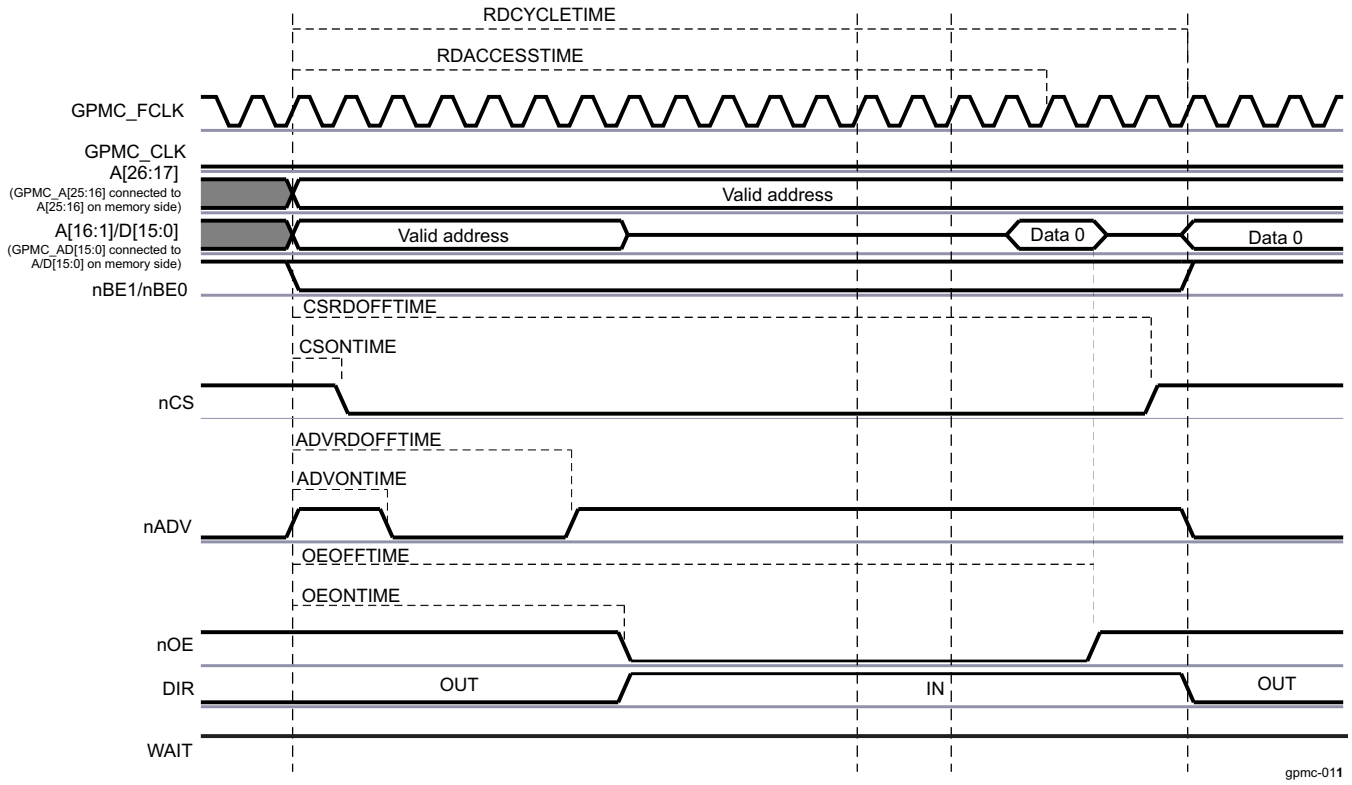
In asynchronous operations GPMC_CLK is not provided outside the GPMC and is kept low.

7.3.4.10.1.1 Access on Address/Data Multiplexed Devices

7.3.4.10.1.1.1 Asynchronous Single-Read Operation on an Address/Data Multiplexed Device

Figure 7-170 shows an asynchronous single read operation on an address/data-multiplexed device.

Figure 7-170. Asynchronous Single Read on an Address/Data-Multiplexed Device



For formulas to calculate timing parameters, see Section 7.3.5.6.1, *GPMC Timing Parameters Formulas*.

Table 7-502 lists the timing bit fields to set up to configure the GPMC in asynchronous single-read mode.

When the GPMC generates a read access to an address/data-multiplexed device, it drives the address bus until nOE assertion time. For more information, see Section 7.3.4.8.2.3, *Address/Data-Multiplexing Interface*.

Address bits (A[16:1] from a GPMC perspective, A[15:0] from an external device perspective) are placed on the address/data bus, and the remaining address bits GPMC_A[27:16] are placed on the address bus. The address phase ends at nOE assertion, when the DIR signal goes from OUT to IN.

- Chip-select signal nCS:
  - nCS assertion time is controlled by the GPMC_CONFIG2_i[3-0] CSONTIME bit field. It controls the address setup time to nCS assertion.
  - nCS deassertion time is controlled by the GPMC_CONFIG2_i[12-8] CSRDOFFTIME bit field. It controls the address hold time from nCS deassertion.
- Address valid signal nADV:
  - nADV assertion time is controlled by the GPMC_CONFIG3_i[3-0] ADVONTIME bit field.
  - nADV deassertion time is controlled by the GPMC_CONFIG3_i[12-8] ADVRDOFFTIME bit field.
- Output enable signal nOE:
  - nOE assertion indicates a read cycle.
  - nOE assertion time is controlled by the GPMC_CONFIG4_i[3-0] OEONTIME bit field.
  - nOE deassertion time is controlled by the GPMC_CONFIG4_i[12-8] OEOFFTIME bit field.

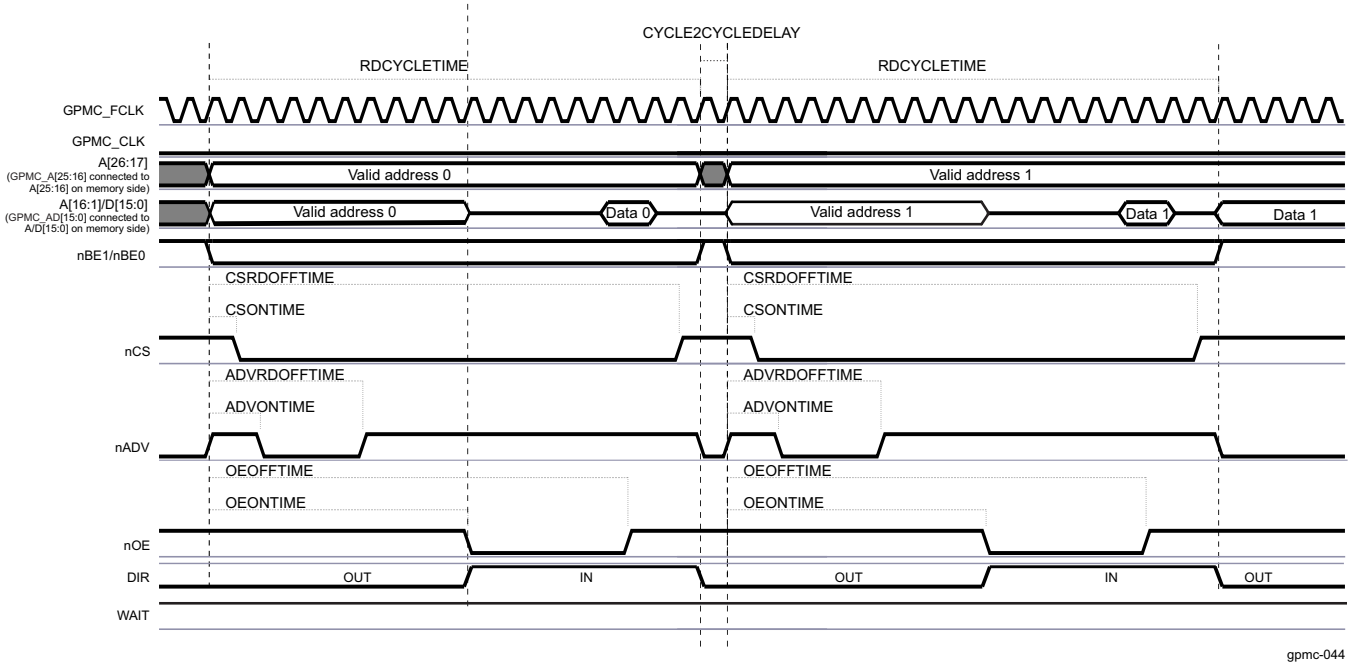
- Read data is latched when RDACCESSTIME completes. Access time is defined in the GPMC_CONFIG5_i[20-16] RDACCESSTIME bit field.
- Direction signal DIR: DIR goes from OUT to IN at the same time that nOE is asserted.
- The end of the access is defined by the GPMC_CONFIG5_i[4-0] RDCYCLETIME parameter.

In the GPMC, when a 16-bit wide device is attached to the controller, a 32-bit word write access is split into two 16-bit word write accesses. For more information about GPMC access size and type adaptation, see Section 7.3.4.10.5, *System Burst Versus External Device Burst Support*.

Between two successive accesses, if an nCS pulse is needed:

- The GPMC_CONFIG6_i[11-8] CYCLE2CYCLEDELAY bit field can be programmed with the GPMC_CONFIG6_i[7] CYCLE2CYCLESAMECSEN bit enabled.
- The CSWROFFTIME and CSONTIME parameters also allow a chip-select pulse, but this affects all other types of access.

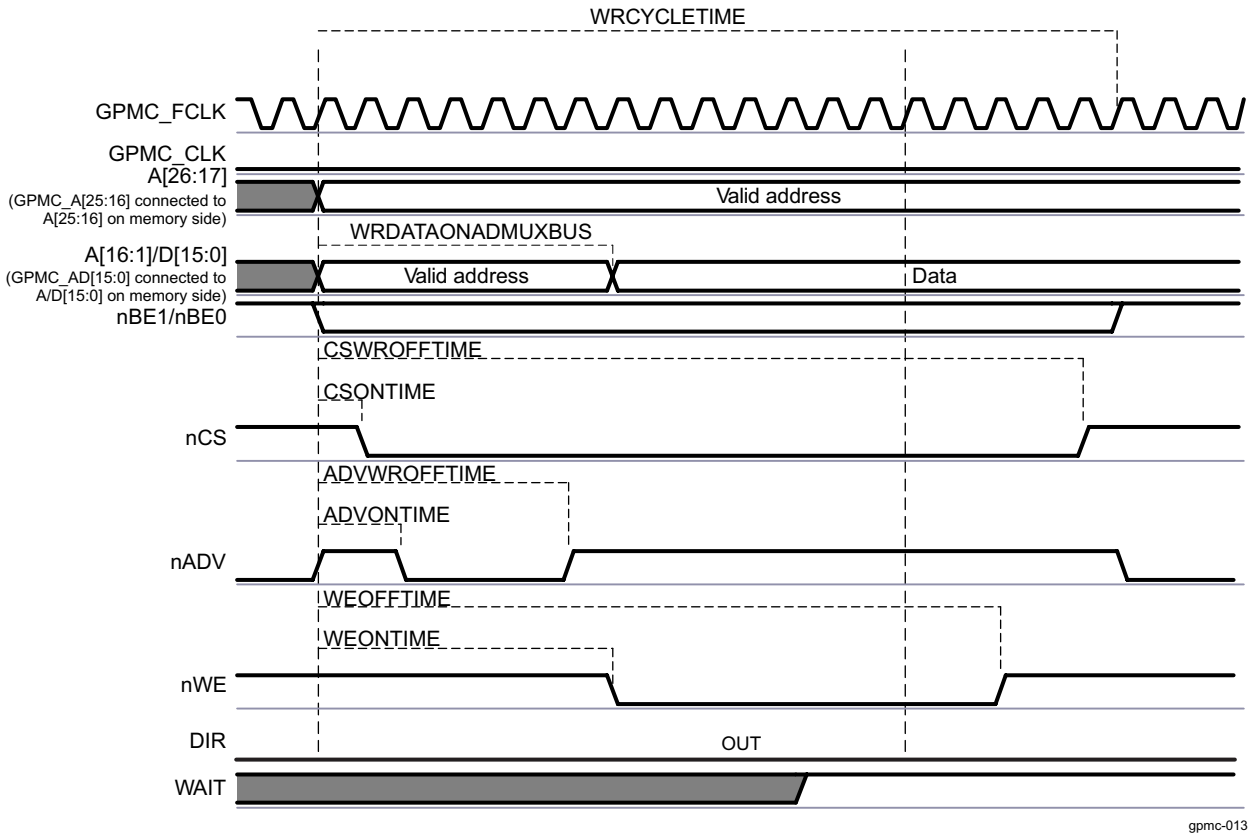
Figure 7-171. Two Asynchronous Single-Read Accesses on an Address/Data-Multiplexed Device (32-Bit Read Split Into 2 x 16-Bit Read)



7.3.4.10.1.1.2 Asynchronous Single-Write Operation on an Address/Data-Multiplexed Device

Figure 7-172 shows an asynchronous single-write operation on an address/data-multiplexed device.

Figure 7-172. Asynchronous Single-Write on an Address/Data-Multiplexed Device



For formulas to calculate timing parameters, see Section 7.3.5.6.1, *GPMC Timing Parameters Formulas*.

Table 7-502 lists the timing bit fields to set up to configure the GPMC in asynchronous single-write mode.

When the GPMC generates a write access to an address/data-multiplexed device, it drives the address bus until nWE assertion time. For more information, see Section 7.3.4.8.2.3, *Address/Data-Multiplexing Interface*.

The nCS and nADV signals are controlled in the same way as for a asynchronous single-read operation on an address/data-multiplexed device.

- Write enable signal nWE:
  - nWE assertion indicates a write cycle.
  - nWE assertion time is controlled by the GPMC_CONFIG4_i[19-16] WEONTIME bit field.
  - nWE deassertion time is controlled by the GPMC_CONFIG4_i[28-24] WEOFFTIME bit field.
- Direction signal DIR: DIR signal is OUT during the entire access.
- The end of the access is defined by the GPMC_CONFIG5_i[12-8] WRCYCLETIME parameter.

Address bits A[16:1] (GPMC point of view) are placed on the address/data bus at the start of cycle time, and the remaining address bits A[26:17] are placed on the address bus.

Data is driven on the address/data bus at a GPMC_CONFIG6_i[19-16] WRDATAONADMUXBUS time.

NOTE: Write multiple access in asynchronous mode is not supported. If WRITEMULTIPLE is enabled with WRITETYPE as asynchronous, the GPMC processes single asynchronous accesses.

After a write operation, if no other access (read or write) is pending, the data bus keeps its previous value. See Section 7.3.4.9.10, *Bus Keeping Support*.

7.3.4.10.1.1.3 Asynchronous Multiple (Page) Write Operation on an Address/Data-Multiplexed Device

Write multiple (page) access in asynchronous mode is not supported for address/data-multiplexed devices.

If the GPMC_CONFIG1_i[28] WRITEMULTIPLE bit is enabled (1h) with the GPMC_CONFIG1_i[27] WRITETYPE bit as asynchronous (0h), the GPMC processes single asynchronous accesses.

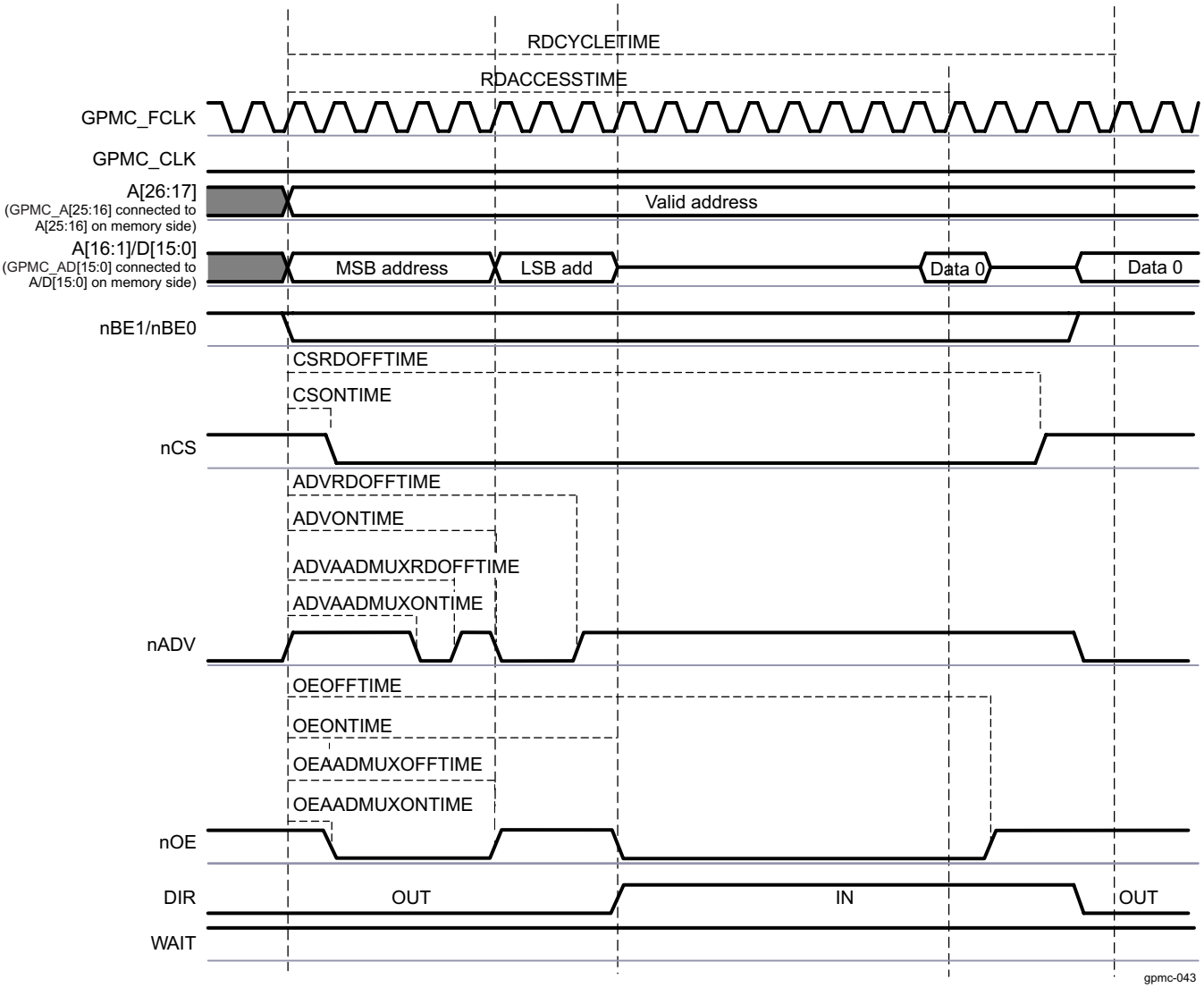
For accesses on non-multiplexed devices, see Section 7.3.4.10.3, *Asynchronous and Synchronous Accesses in non-multiplexed Mode*.

7.3.4.10.1.2 Access on Address/Address/Data-Multiplexed Devices

7.3.4.10.1.2.1 Asynchronous Single Read Operation on an AAD-Multiplexed Device

Figure 7-173 shows an asynchronous single-read operation on an AAD-multiplexed device.

Figure 7-173. Asynchronous Single Read on an AAD-Multiplexed Device



For formulas to calculate timing parameters, see Section 7.3.5.6.1, *GPMC Timing Parameters Formulas*.

Table 7-502 lists the timing bit fields to set up to configure the GPMC in asynchronous single write mode.

When the GPMC generates a read access to an AAD-multiplexed device, all address bits are driven onto the address/data bus in two separate phases. The first phase is used for the MSB address and is qualified with nOE driven low. The first address phase ends at the first nOE deassertion time. The second phase for LSB address is qualified with nOE driven high. The second address phase ends at the second nOE assertion time, when the DIR signal goes from OUT to IN.

The nCS and DIR signals are controlled in the same way as for an asynchronous single-read operation on an address/data-multiplexed device.

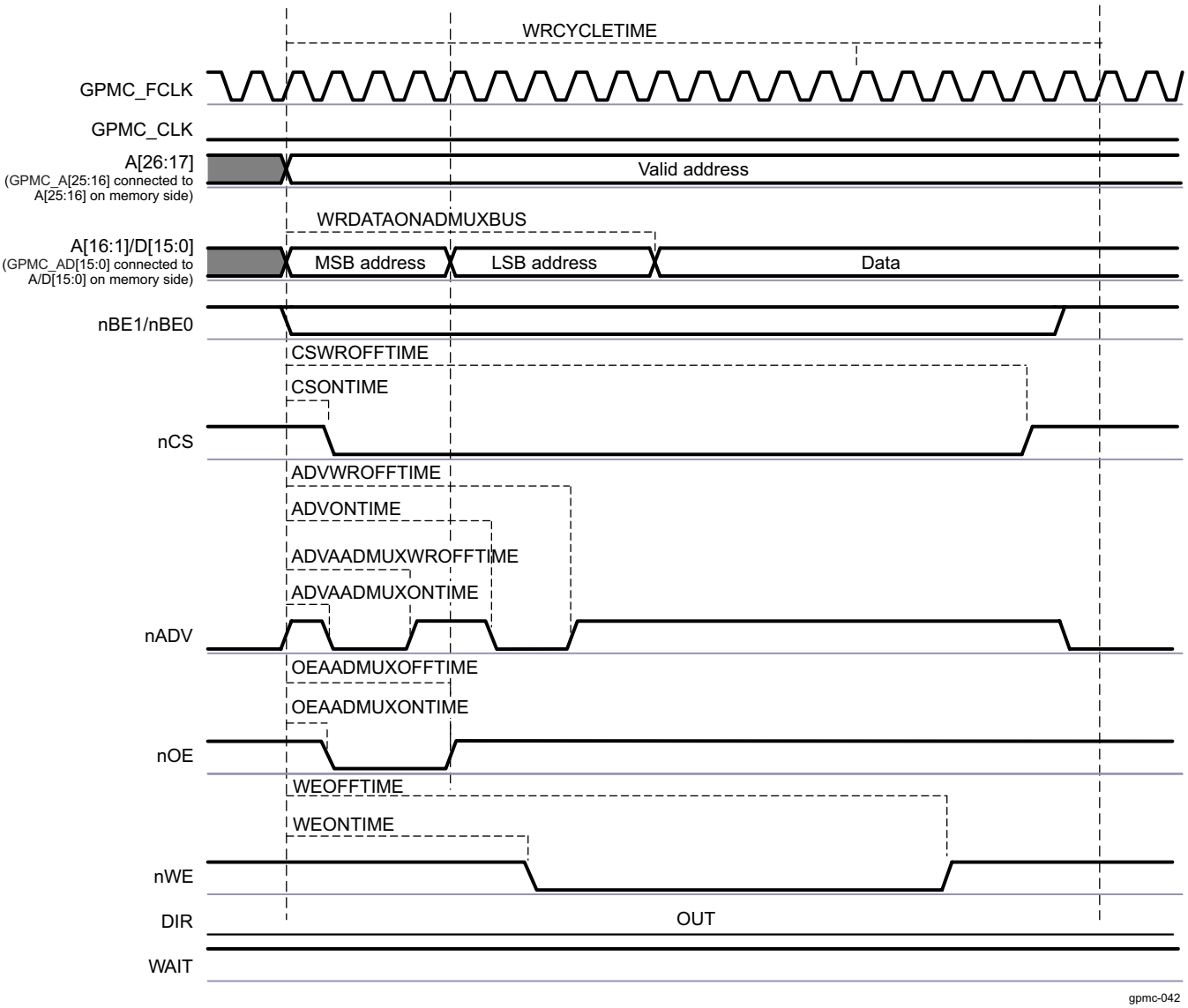
- Address valid signal nADV. nADV is asserted and deasserted twice during a read transaction:
  - nADV first assertion time is controlled by the GPMC_CONFIG3_i[6-4] ADVAADMUXONTIME bit field.

  - nADV first deassertion time is controlled by the GPMC_CONFIG3_i[26-24] ADVAADMUXRDOFFTIME bit field.
  - nADV second assertion time is controlled by the GPMC_CONFIG3_i[3-0] ADVONTIME bit field.
  - nADV second deassertion time is controlled by the GPMC_CONFIG3_i[12-8] ADVRDOFFTIME bit field.
- Output Enable signal nOE. nOE is asserted and deasserted twice during a read transaction (nOE second assertion indicates a read cycle):
  - nOE first assertion time is controlled by the GPMC_CONFIG4_i[6-4] OEAADMUXONTIME bit field.
  - nOE first deassertion time is controlled by the GPMC_CONFIG3_i[15-13] OEAADMUXOFFTIME bit field.
  - nOE second assertion time is controlled by the GPMC_CONFIG4_i[3-0] OEONTIME bit field.
  - nOE second deassertion time is controlled by the GPMC_CONFIG4_i[12-8] OEOFFTIME bit field.

7.3.4.10.1.2.2 Asynchronous Single-Write Operation on an AAD-Multiplexed Device

Figure 7-174 shows an asynchronous single-write operation on an AAD-multiplexed device.

Figure 7-174. Asynchronous Single Write on an AAD-Multiplexed Device



For formulas to calculate timing parameters, see Section 7.3.5.6.1, *GPMC Timing Parameters Formulas*.

Table 7-502 lists the timing bit fields to set up to configure the GPMC in asynchronous single-write mode.

When the GPMC generates a write access to an AAD-multiplexed device, all address bits are driven onto the address/data bus in two separate phases. The first phase is used for the MSB address and is qualified with nOE driven low. The second phase for LSB address is qualified with nOE driven high. The address phase ends at nWE assertion time.

The nCS, nWE, and DIR signals are controlled in the same way as for an asynchronous single-write operation on an address/data-multiplexed device. See Table 7-493.

- Address valid signal nADV is asserted and deasserted twice during a write transaction:
  - nADV first assertion time is controlled by the GPMC_CONFIG3_i[6-4] ADVAADMUXONTIME bit field.
  - nADV first deassertion time is controlled by the GPMC_CONFIG3_i[30-28] ADVAADMUXWROFFTIME bit field.
  - nADV second assertion time is controlled by the GPMC_CONFIG3_i[3-0] ADVONTIME bit field.
  - nADV second deassertion time is controlled by the GPMC_CONFIG3_i[20-16] ADVWROFFTIME bit field.
- Output enable signal nOE is asserted during the address phase of a write transaction:
  - nOE assertion time is controlled by the GPMC_CONFIG4_i[6-4] OEAADMUXONTIME bit field.
  - nOE deassertion time is controlled by the GPMC_CONFIG3_i[15-13] OEAADMUXOFFTIME bit field.

The address bits for the first address phase are driven onto the data bus until nOE deassertion. Data is driven onto the address/data bus at the clock edge defined by the GPMC_CONFIG6_i[19-16] WRDATAONADMUXBUS parameter.

7.3.4.10.1.2.3 Asynchronous Multiple (Page) Read Operation on an AAD-Multiplexed Device

Write multiple (page) access in asynchronous mode is not supported for AAD-multiplexed devices.

If the GPMC_CONFIG1_i[28] WRITEMULTIPLE bit is enabled (1h) with the GPMC_CONFIG1_i[27] WRITETYPE bit as asynchronous (0h), the GPMC processes single asynchronous accesses.

For accesses on non-multiplexed devices, see Section 7.3.4.10.3, *Asynchronous and Synchronous Accessed in non-multiplexed Mode*.

7.3.4.10.2 Synchronous Access Description

This section describes read and write synchronous accesses on address/data-multiplexed devices. All information in this section can be applied to any type of memory (non-multiplexed, address and data-multiplexed, or AAD-multiplexed) with the difference limited to the address phase. For accesses on non-multiplexed devices, see Section 7.3.4.10.3, *Asynchronous and Synchronous Accessed in non-multiplexed Mode*.

In synchronous operations:

- The GPMC_CLK clock is provided outside the GPMC when accessing the memory device.
- The GPMC_CLK clock is derived from the GPMC_FCLK clock using the GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER bit field. In the following section i stands for the chip-select number, i = 0 to 3.
- The GPMC_CONFIG1_i[26-25] CLKACTIVATIONTIME bit field specifies that the GPMC_CLK is provided outside the GPMC for 0 to 2 GPMC_FCLK cycles after start access time until RDCYCLETIME or WRCYCLETIME completes.

7.3.4.10.2.1 Synchronous Single Read

Figure 7-175 and Figure 7-176 show a synchronous single-read operation with GPMCFCLKDIVIDER equal to 0 and 1, respectively.

Figure 7-175. Synchronous Single Read (GPMCFCLKDIVIDER = 0)

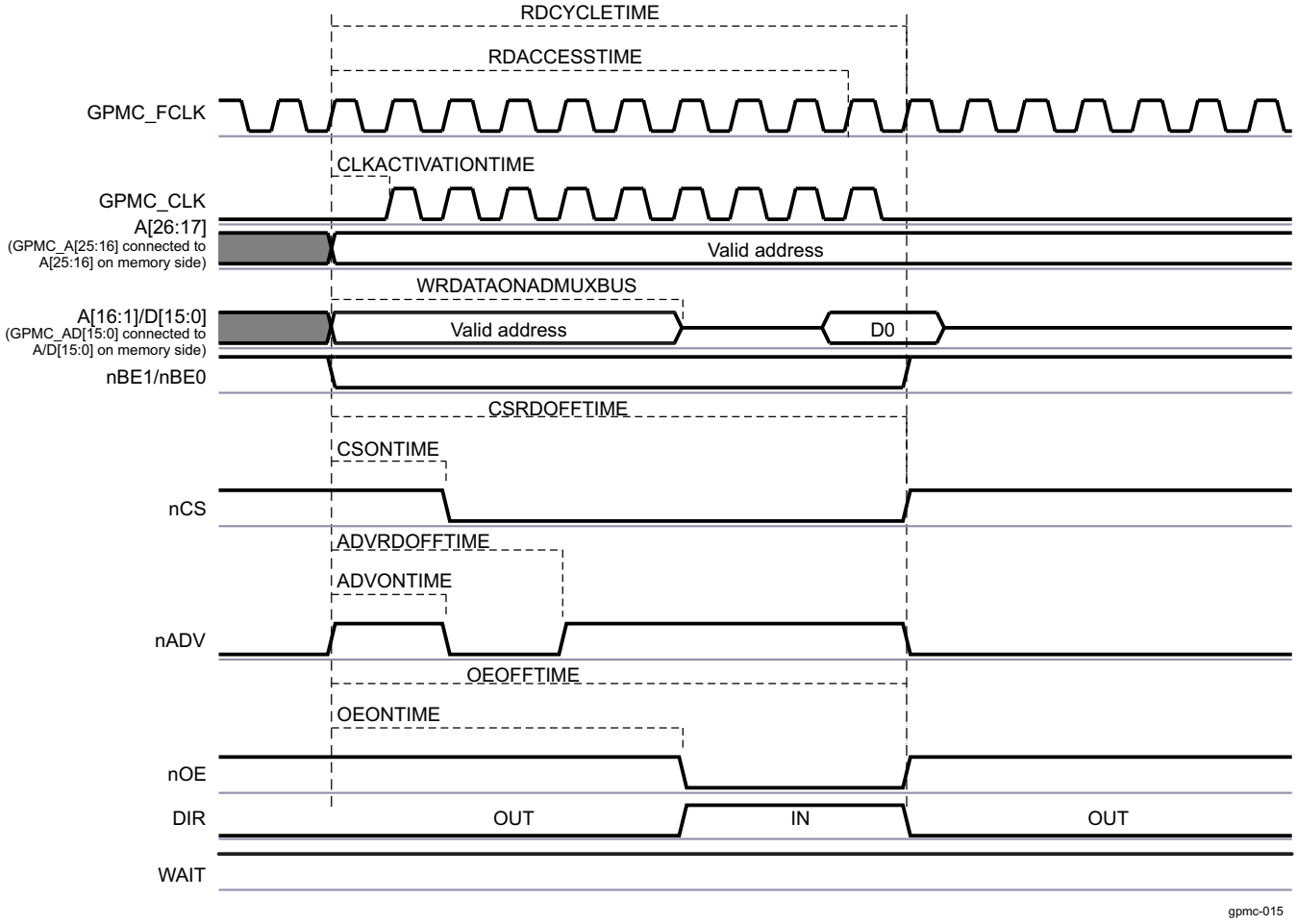
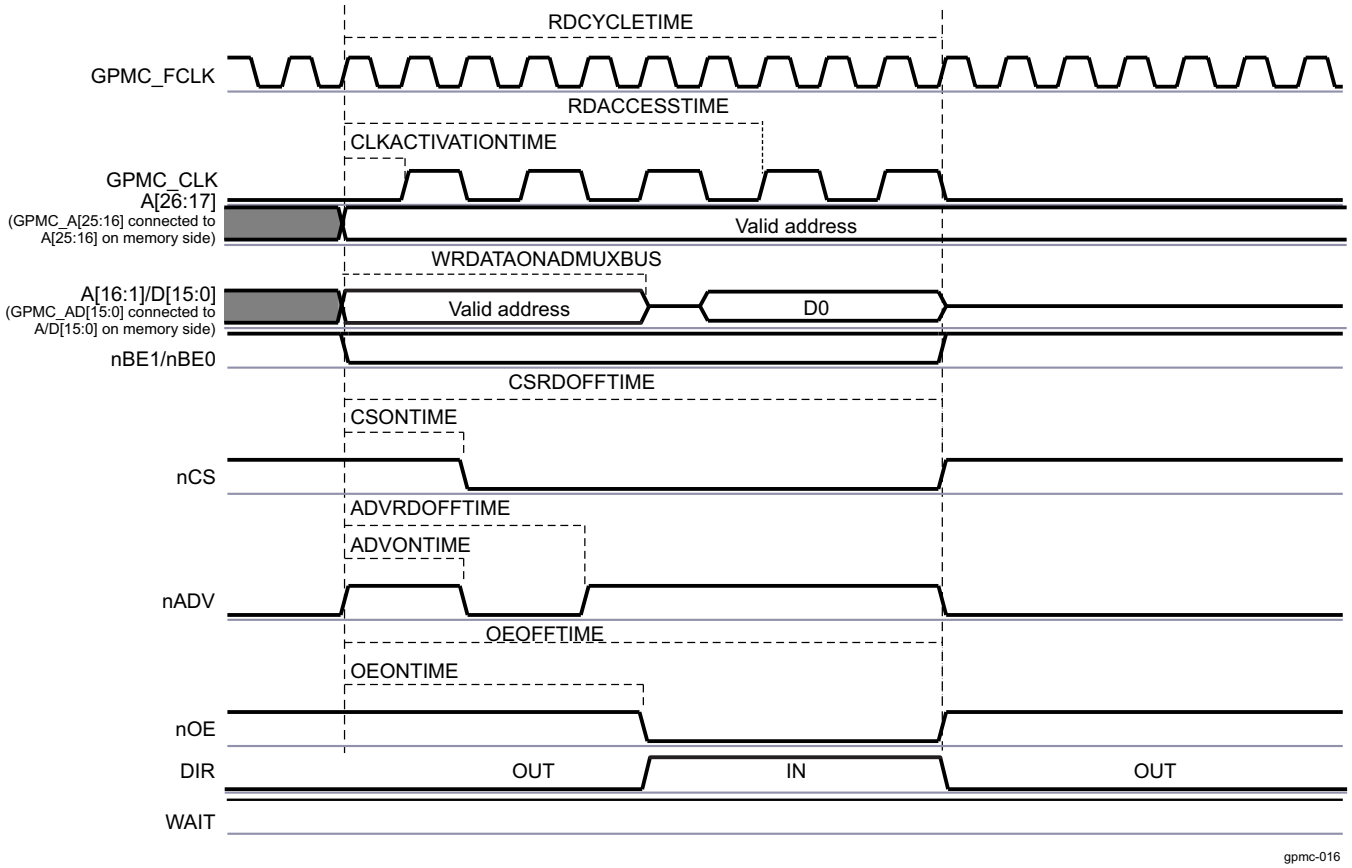


Figure 7-176. Synchronous Single Read (GPMCFCLKDIVIDER = 1)



For formulas to calculate timing parameters, see Section 7.3.5.6.1, *GPMC Timing Parameters Formulas*.

Table 7-502 lists the timing bit fields to set up to configure the GPMC in asynchronous single-read mode.

When the GPMC generates a read access to an address/data-multiplexed device, it drives the address bus until nOE assertion time. For more information, see Section 7.3.4.8.2.3, *Address/Data-Multiplexing Interface*.

- Chip-select signal nCS:
  - nCS assertion time is controlled by the GPMC_CONFIG2_i[3-0] CSONTIME bit field and ensures address setup time to nCS assertion.
  - nCS deassertion time is controlled by the GPMC_CONFIG2_i[12-8] CSRDOFFTIME bit field and ensures address hold time to nCS deassertion.
- Address valid signal nADV:
  - nADV assertion time is controlled by the GPMC_CONFIG3_i[3-0] ADVONTIME bit field.
  - nADV deassertion time is controlled by the GPMC_CONFIG3_i[12-8] ADVRDOFFTIME bit field.
- Output enable signal nOE:
  - nOE assertion indicates a read cycle.
  - nOE assertion time is controlled by the GPMC_CONFIG4_i[3-0] OEONTIME bit field.
  - nOE deassertion time is controlled by the GPMC_CONFIG4_i[12-8] OEOFFTIME bit field.
- Initial latency for the first read data is controlled by GPMC_CONFIG5_i[20-16] RDACCESSTIME bit field or by monitoring the WAIT signal.
- Total access time (the GPMC_CONFIG5_i[4-0] RDCYCLETIME bit field) corresponds to RDACCESSTIME plus the address hold time from nCS deassertion, plus time from RDACCESSTIME to CSRDOFFTIME.

- Direction signal DIR: DIR goes from OUT to IN at the same time as nOE assertion.

When the GPMC generates a write access to an AAD-multiplexed device, all address bits are driven onto the address/data bus in two separate phases. The first phase is used for the MSB address and is qualified with nOE driven low. The second phase for LSB address is qualified with nOE driven high. The address phase ends at nWE assertion time.

The nCS and DIR signals are controlled in the same way as for a synchronous single-read operation on an address/data-multiplexed device.

- Address valid signal nADV is asserted and deasserted twice during a read transaction:
  - nADV first assertion time is controlled by the GPMC_CONFIG3_i[6-4] ADVAADMUXONTIME bit field.
  - nADV first deassertion time is controlled by the GPMC_CONFIG3_i[26-24] ADVAADMUXRDOFFTIME bit field.
  - nADV second assertion time is controlled by the GPMC_CONFIG3_i[3-0] ADVONTIME bit field.
  - nADV second deassertion time is controlled by the GPMC_CONFIG3_i[12-8] ADVRDOFFTIME bit field.
- Output Enable signal nOE is asserted and deasserted twice during a read transaction (nOE second assertion indicates a read cycle):
  - nOE first assertion time is controlled by the GPMC_CONFIG4_i[6-4] OEAADMUXONTIME bit field.
  - nOE first deassertion time is controlled by the GPMC_CONFIG3_i[15-13] OEAADMUXOFFTIME bit field.
  - nOE second assertion time is controlled by the GPMC_CONFIG4_i[3-0] OEONTIME bit field.
  - nOE second deassertion time is controlled by the GPMC_CONFIG4_i[12-8] OEOFFTIME bit field.

After a read operation, if no other access (read or write) is pending, the data bus is driven with the previous read value. See Section 7.3.4.9.10, *Bus Keeping Support*.

7.3.4.10.2.2 Synchronous Multiple (Burst) Read (4-, 8-, 16-Word16 Burst With Wraparound Capability)

Figure 7-177 and Figure 7-178 show a synchronous multiple-read operation with GPMCFCLKDivider equal to 0 and 1, respectively.

Figure 7-177. Synchronous Multiple (Burst) Read (GPMCFCLKDIVIDER = 0)

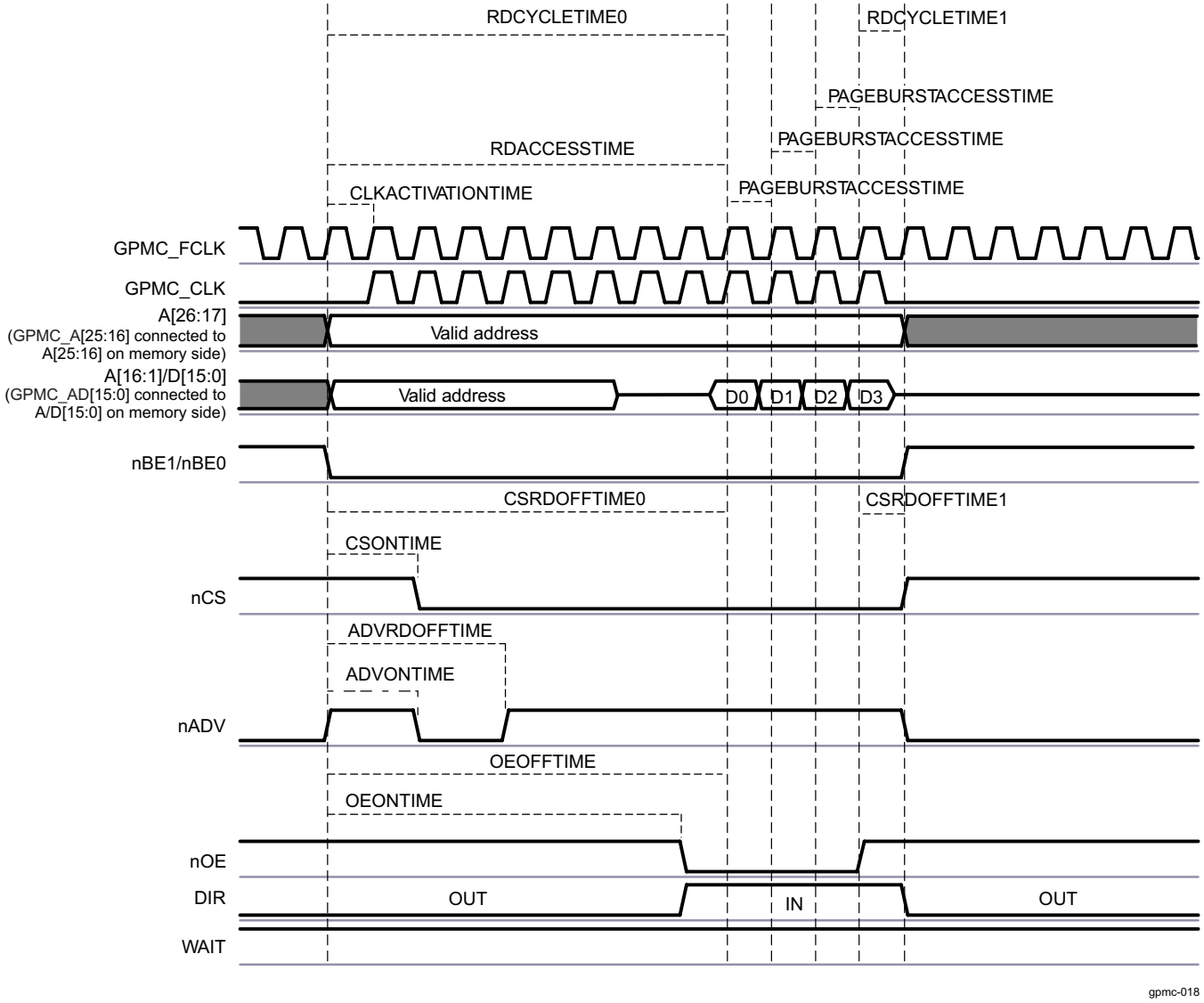
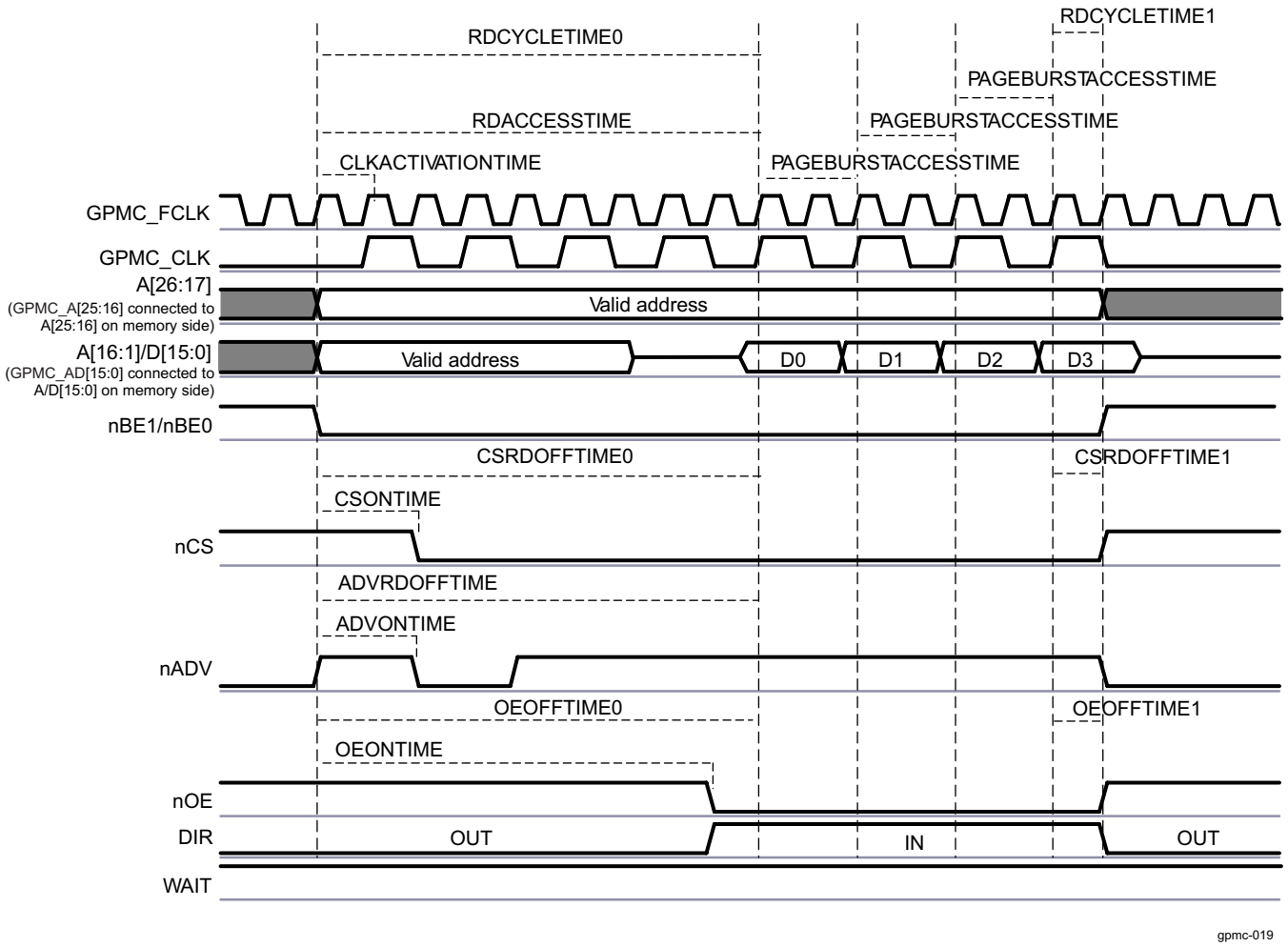


Figure 7-178. Synchronous Multiple (Burst) Read (GPMCFCLKDIVIDER = 1)



When the GPMC_CONFIG5_i[20-16] RDACCESSTIME bit field completes, control-signal timings are frozen during the multiple data transactions, corresponding to the GPMC_CONFIG5_i[27-24] PAGEBURSTACCESSTIME bit field multiplied by the number of remaining data transactions.

The nCS, nADV, nOE, and DIR signals are controlled in the same way as for a synchronous single-read operation. See Table 7-488.

Initial latency for the first read data is controlled by RDACCESSTIME or by monitoring the WAIT signal. Successive read data are provided by the memory device every one or two GPMC_CLK cycles. The PAGEBURSTACCESSTIME parameter must be set accordingly with the GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER bit field and the memory-device internal configuration. Depending on the device page length, the GPMC checks the device page crossing during a new burst request and purposely inserts initial latency (of RDACCESSTIME) when required.

Total access time GPMC_CONFIG5_i[4-0] RDCYCLETIME corresponds to RDACCESSTIME plus the address hold time from nCS deassertion. In Figure 7-178, the programmed value of RDCYCLETIME equals RDCYCLETIME0 + RDCYCLETIME1.

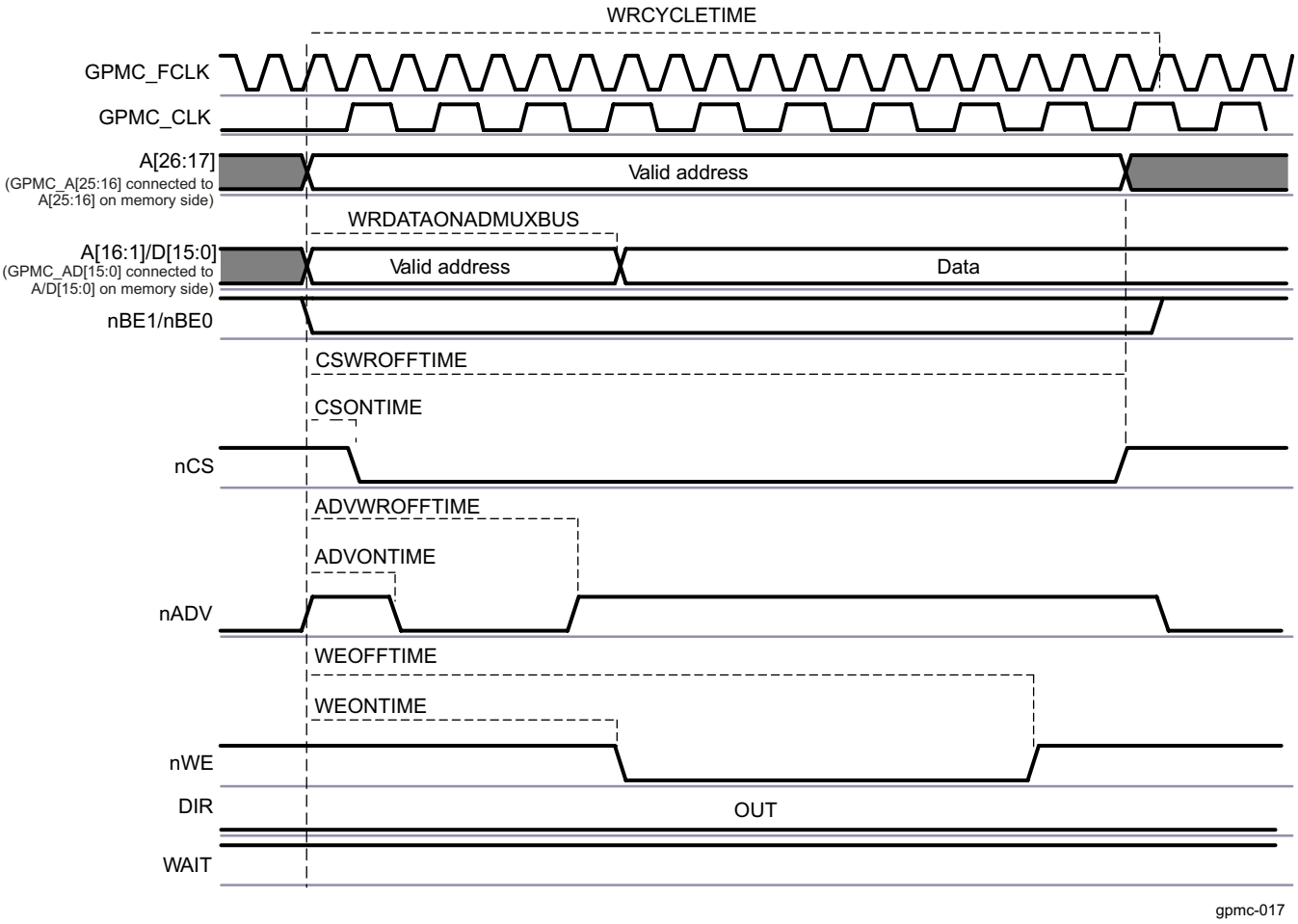
After a read operation, if no other access (read or write) is pending, the data bus is driven with the previous read value. See Section 7.3.4.9.10, *Bus Keeping Support*.

Burst wraparound is enabled through the GPMC_CONFIG1_i[31] WRAPBURST bit and allows a 4-, 8-, or 16-Word16 linear burst access to wrap within its burst-length boundary through the GPMC_CONFIG1_i[24-23] ATTACHEDDEVICEPAGELENGTH bit field.

7.3.4.10.2.3 Synchronous Single Write

Burst write mode is used for synchronous single or burst accesses.

Figure 7-179. Synchronous Single Write on an Address/Data-Multiplexed Device



When the GPMC generates a write access to an address/data-multiplexed device, it drives the data bus (with address bits A[16:1]) until the GPMC_CONFIG6_i[19-16] WRDATAONADMUXBUS bit field time. The first data of the burst is driven on the address/data bus at WRDATAONADMUXBUS time.

7.3.4.10.2.4 Synchronous Multiple (Burst) Write

Synchronous burst write mode provides synchronous single or consecutive accesses.

Figure 7-180 shows a synchronous burst write access when the chip-select is configured in address/data-multiplexed mode.

Figure 7-180. Synchronous Multiple Write (Burst Write) in Address/Data-Multiplexed Mode

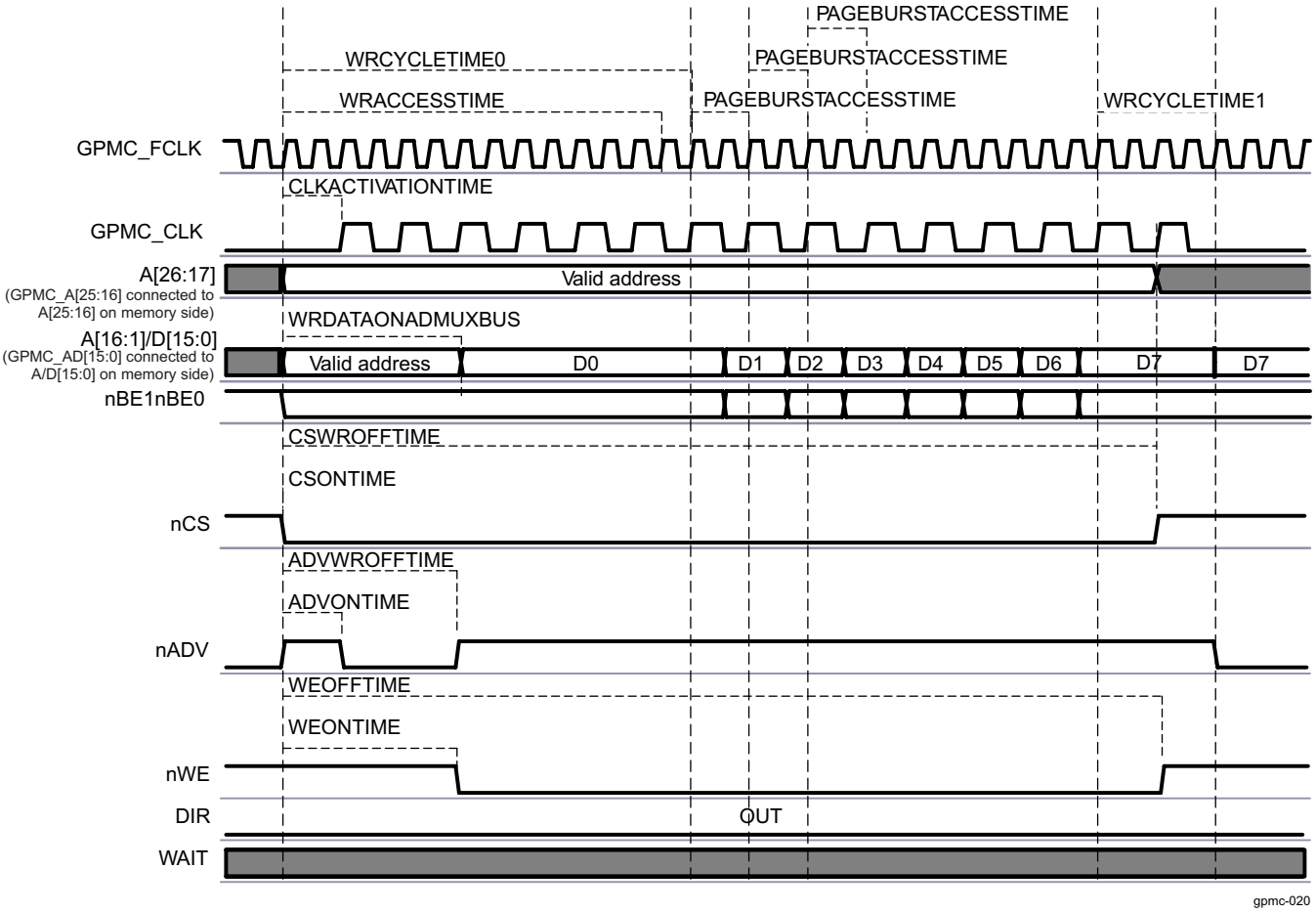
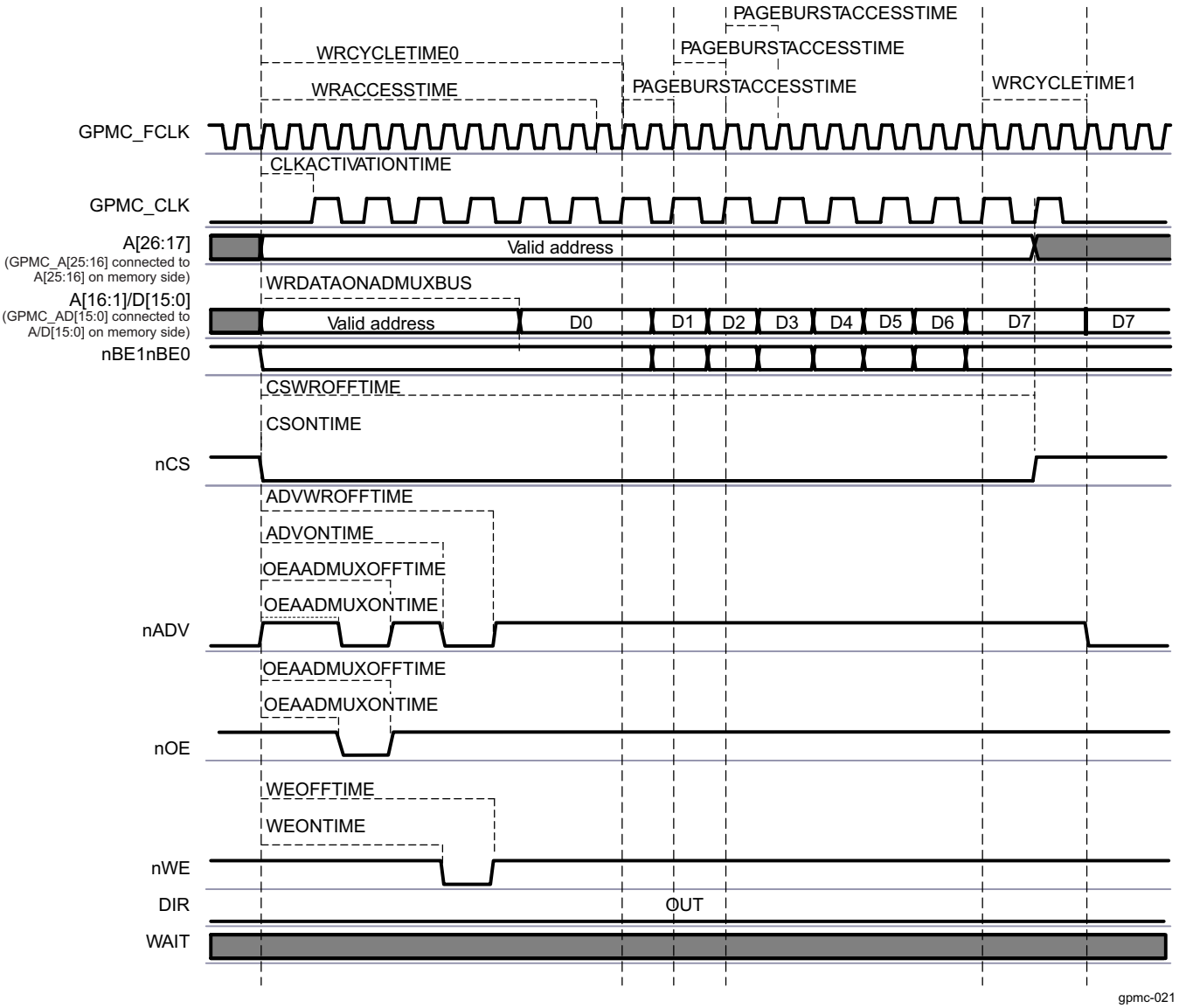


Figure 7-181 shows the same synchronous burst write access when the chip-select is configured in address/address/data-multiplexed (AAD-multiplexed) mode.

Figure 7-181. Synchronous Multiple Write (Burst Write) in Address/Address/Data-Multiplexed Mode



The first data of the burst is driven on the A/D bus at the GPMC_CONFIG6_i[19-16] WRDATAONADMUXBUS bit field.

When WRACCESSTIME completes, control-signal timings are frozen during the multiple data transactions, corresponding to the GPMC_CONFIG5_i[27-24] PAGEBURSTACCESSTIME bit field multiplied by the number of remaining data transactions.

When the GPMC generates a read access to an address/data-multiplexed device, it drives the address bus until nOE assertion time. For more information, see Section 7.3.4.8.2.3, *Address/Data-Multiplexing Interface*.

- Chip-select signal nCS:
  - nCS assertion time is controlled by the GPMC_CONFIG2_i[3-0] CSONTIME bit field (where i = 0 to 3) and ensures address setup time to nCS assertion.
  - nCS deassertion time controlled by the GPMC_CONFIG2_i[20-16] CSWROFFTIME bit field and ensures address hold time to nCS deassertion.
- Address valid signal nADV:
  - nADV assertion time is controlled by the GPMC_CONFIG3_i[3-0] ADVONTIME bit field.

  - nADV deassertion time is controlled by the GPMC_CONFIG3_i[20-16] ADVWROFFTIME bit field.
- Write enable signal nWE:
  - nWE assertion indicates a read cycle.
  - nWE assertion time is controlled by the GPMC_CONFIG4_i[19-16] WEONTIME bit field.
  - nWE deassertion time is controlled by the GPMC_CONFIG4_i[28-24] WEOFFTIME bit field.

NOTE: The nWE falling edge must not be used to control the time when the burst first data is driven in the address/data bus, because some new devices require the nWE signal to be low during the address phase.

- Direction signal DIR is OUT during the entire access.

When the GPMC generates a write access to an AAD-multiplexed device, all address bits are driven onto the address/data bus in two separate phases. The first phase is used for the MSB address and is qualified with nOE driven low. The second phase for LSB address is qualified with nOE driven high. The address phase ends at nWE assertion time.

The nCS, and DIR signals are controlled as previously described.

- Address valid signal nADV is asserted and deasserted twice during a read transaction:
  - nADV first assertion time is controlled by the GPMC_CONFIG3_i[6-4] ADVAADMUXONTIME bit field.
  - nADV first deassertion time is controlled by the GPMC_CONFIG3_i[26-24] ADVAADMUXRDOFFTIME bit field.
  - nADV second assertion time is controlled by the GPMC_CONFIG3_i[3-0] ADVONTIME bit field.
  - nADV second deassertion time is controlled by the GPMC_CONFIG3_i[12-8] ADVRDOFFTIME bit field.
- Output Enable signal nOE is asserted and deasserted twice during a read transaction (nOE second assertion indicates a read cycle):
  - nOE first assertion time is controlled by the GPMC_CONFIG4_i[6-4] OEAADMUXONTIME bit field.
  - nOE first deassertion time is controlled by the GPMC_CONFIG4_i[15-13] OEAADMUXOFFTIME bit field.
  - nOE second assertion time is controlled by the GPMC_CONFIG4_i[3-0] OEONTIME bit field.
  - nOE second deassertion time is controlled by the GPMC_CONFIG4_i[12-8] OEOFFTIME bit field.

First write data is driven by the GPMC at GPMC_CONFIG6_i[19-16] WRDATAONADMUXBUS, when in address/data-multiplexed configuration. The next write data of the burst is driven on the bus at WRACCESSTIME + 1 during GPMC_CONFIG5_i[27-24] PAGEBURSTACCESSTIME GPMC_FCLK cycles. The last data of the synchronous burst write is driven until GPMC_CONFIG5_i[12-8] WRCYCLETIME completes.

- WRACCESSTIME is defined in the GPMC_CONFIG6_i[28-24] bit field.
- The PAGEBURSTACCESSTIME parameter must be set accordingly with GPMCFCLKDIVIDER and the memory-device internal configuration.

Total access time GPMC_CONFIG5_i[12-8] WRCYCLETIME corresponds to WRACCESSTIME plus the address hold time from nCS deassertion. In Figure 7-180, the programmed value of WRCYCLETIME equals WRCYCLETIME0 + WRCYCLETIME1. WRCYCLETIME0 and WRCYCLETIME1 delays are not actual parameters and are only a graphical representation of the full WRCYCLETIME value.

After a write operation, if no other access (read or write) is pending, the data bus keeps the previous value. See Section 7.3.4.9.10, *Bus Keeping Support*.

7.3.4.10.3 Asynchronous and Synchronous Accesses in non-multiplexed Mode

Page mode is available only in non-multiplexed mode.

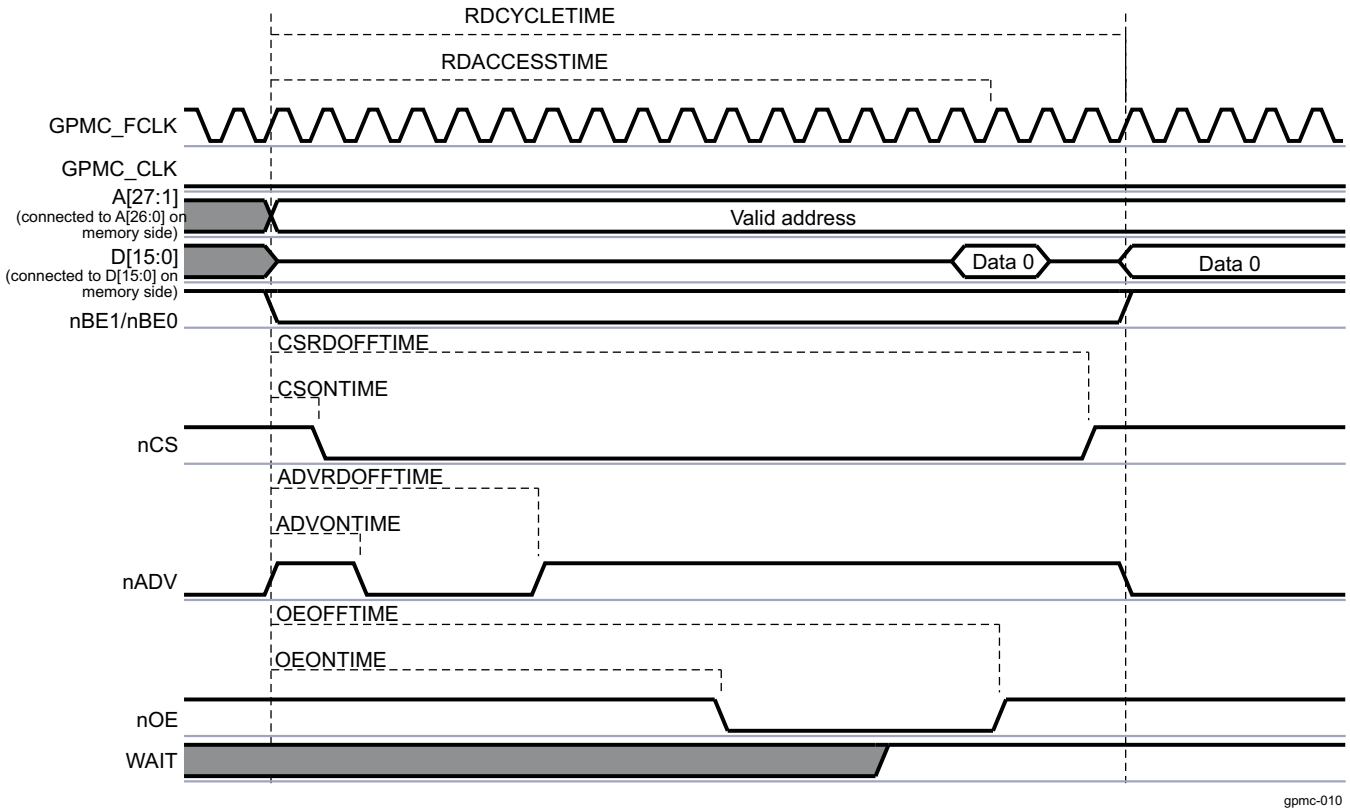
- Asynchronous single-read operation on a non-multiplexed device
- Asynchronous single-write operation on a non-multiplexed device
- Asynchronous multiple- (page mode) read operation on a non-multiplexed device

- Synchronous operations on a non-multiplexed device

7.3.4.10.3.1 Asynchronous Single-Read Operation on non-multiplexed Device

Figure 7-182 shows an asynchronous single-read operation on a non-multiplexed device.

Figure 7-182. Asynchronous Single Read on an Address/Data-non-multiplexed Device



The 27-bit address (For a 16-bit data memory device, hence GPMC A[0] is not necessary to be output) is driven onto the address bus A[27:1] and the 16-bit data is driven onto the data bus D[15:0].

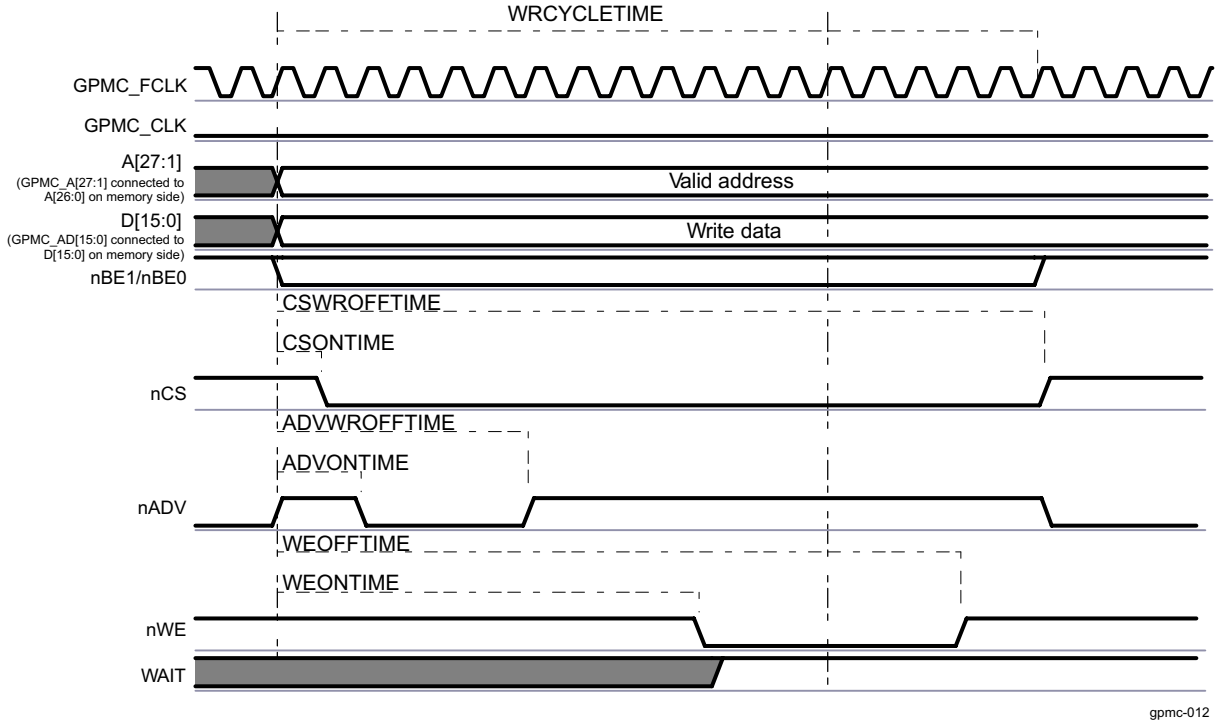
Read data is latched at GPMC_CONFIG1_5[20-16] RDACCESSTIME completion time. The end of the access is defined by the GPMC_CONFIG1_5[4-0] RDCYCLETIME parameter.

The nCS, nADV, nOE, and DIR signals are controlled in the same way as address/data-multiplexed accesses (see Table 7-493).

7.3.4.10.3.2 Asynchronous Single-Write Operation on non-multiplexed Device

Figure 7-183 shows an asynchronous single-write operation on a non-multiplexed device.

Figure 7-183. Asynchronous Single Write on an Address/Data-non-multiplexed Device

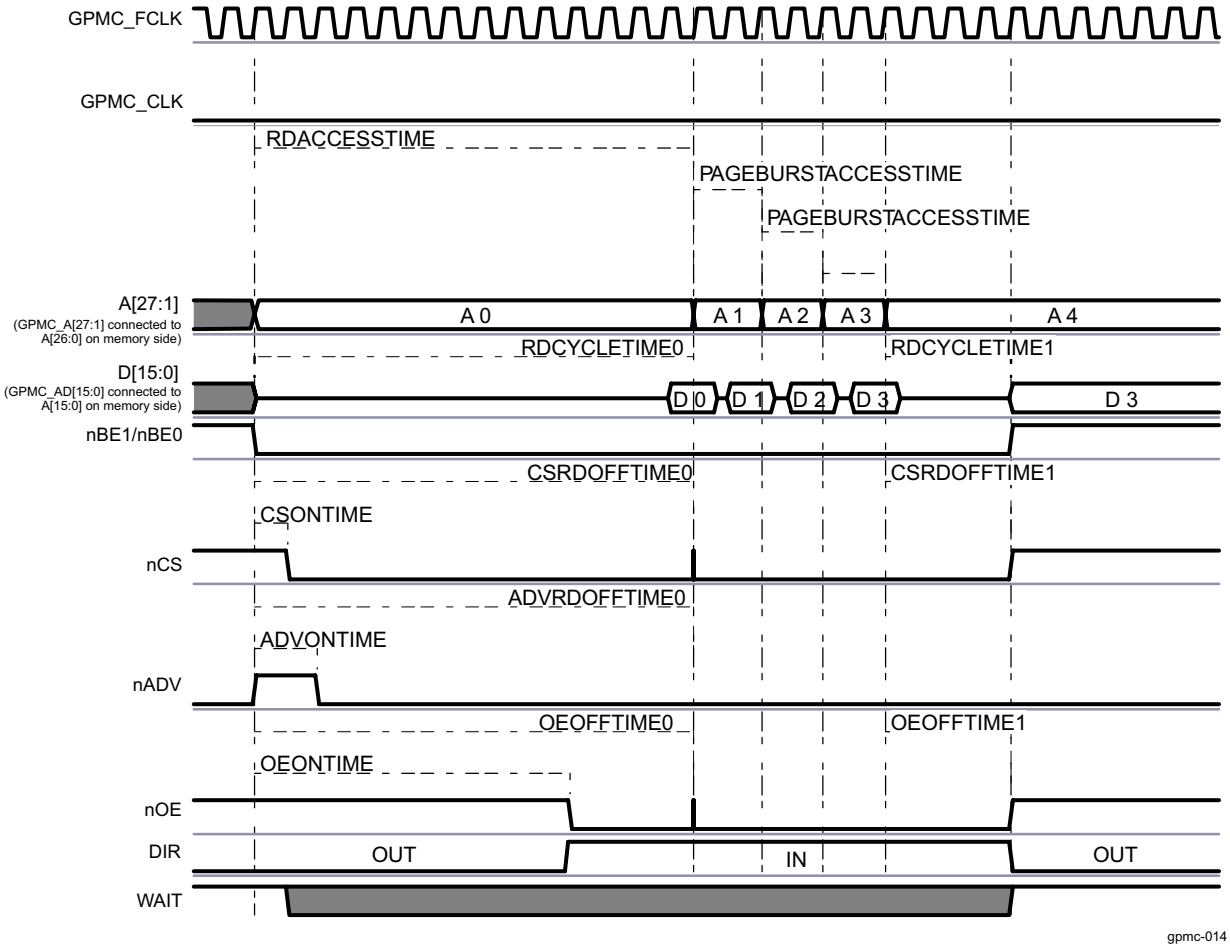


The nCS, nADV, nWE, and DIR signals are controlled in the same way as address/data-multiplexed accesses (see Table 7-493).

7.3.4.10.3.3 Asynchronous Multiple (Page Mode) Read Operation on non-multiplexed Device

Figure 7-184 shows an asynchronous multiple-read operation on a non-multiplexed device in which two word32 host read accesses to the GPMC are split into one multiple- (page mode of 4 word16) read access to the attached device.

Figure 7-184. Asynchronous Multiple (Page Mode) Read



NOTE: The WAIT signal is active low.

The nCS, nADV, nOE, and DIR signals are controlled in the same way as address/data-multiplexed accesses (see Table 7-493).

When RDACCESSTIME completes, control signal timings are frozen during the multiple data transactions, corresponding to PAGEBURSTACCESSTIME multiplied by the number of remaining data transactions.

Read data is latched at GPMC_CONFIG5_i[20-16] RDACCESSTIME completion time (where i = 0 to 3). The end of the access is defined by the GPMC_CONFIG5_i[4-0] RDCYCLETIME parameter.

During consecutive accesses, the GPMC increments the address after each data read completes.

Delay between successive read data in the page is controlled by the GPMC_CONFIG5_i[27-24] PAGEBURSTACCESSTIME parameter. Depending on the device page length, the GPMC can control device page crossing during a burst request and insert initial RDACCESSTIME latency. Page crossing is possible only with a new burst access, meaning a new initial access phase is initiated.

Total access time RDCYCLETIME corresponds to RDACCESSTIME, plus the address hold time, starting from the nCS deassertion.

- The read cycle time is defined in the GPMC_CONFIG5_i[4-0] RDCYCLETIME bit field.
- In Figure 7-184, the programmed value of RDCYCLETIME equals RDCYCLETIME0 (before paged accesses) + RDCYCLETIME1 (after paged accesses).

7.3.4.10.3.4 Synchronous Operations on a non-multiplexed Device

All information for this section is equivalent to similar operations for address/data-multiplexed or AAD-multiplexed accesses. The only difference resides in the address phase. See Section 7.3.5.3, *GPMC Configuration in NOR Mode*.

7.3.4.10.4 Page and Burst Support

Each chip-select can be configured to process system single or burst requests into successive single accesses or asynchronous page/synchronous burst accesses, with appropriate access size adaptation.

Depending on the external device page or burst capability, read and write accesses can be independently configured through the GPMC. The GPMC_CONFIG1_i[30] READMULTIPLE and GPMC_CONFIG1_i[28] WRITMULTIPLE bits (where i = 0 to 3) are associated with the READTYPE and WRITETYPE parameters.

NOTE:

- Asynchronous write page mode is not supported.
- 8-bit-wide device support is limited to nonburstable devices (READMULTIPLE and WRITEMULTIPLE are ignored).
- Not applicable to NAND device interfacing

7.3.4.10.5 System Burst vs External Device Burst Support

The device system can issue the following requests to the GPMC:

- Byte, 16-bit word, 32-bit word requests (byte-enable-controlled). This is always a single request from the interconnect point of view.
- Incrementing fixed-length bursts of two, four, and eight words
- Wrapped (critical word access first) fixed-length burst of two, four, or eight words

To process a system request with the optimal protocol, the READMULTIPLE (and READTYPE) and WRITEMULTIPLE (and WRITETYPE) parameters must be set according to the burstable capability (synchronous or asynchronous) of the attached device.

The GPMC access engine issues only fixed-length bursts. The maximum length that can be issued is defined per chip-select by the GPMC_CONFIG1_i[24-23] ATTACHEDDEVICEPAGELENGTH bit field (where i = 0 to 3). When the value of ATTACHEDDEVICEPAGELENGTH is less than the length of the system burst request (including the appropriate access size adaptation according to the device width), the GPMC splits the system burst request into multiple bursts. Within the specified 4-, 8-, or 16-word value, the value of the ATTACHEDDEVICEPAGELENGTH bit field must correspond to the maximum length burst supported by the memory device configured in fixed-length burst mode (as opposed to continuous burst mode).

To get optimal performance from memory devices that natively support 16 Word16-length-wrapping burst capability (critical word access first), the ATTACHEDDEVICEPAGELENGTH parameter must be set to 16 words and the GPMC_CONFIG1_i[31] WRAPBURST bit (where i = 0 to 3) must be set to 1. Similarly DEVICEPAGELENGTH is set to 4 and 8 for memories supporting 4 and 8 Word16-length-wrapping burst, respectively.

When the memory device does not offer (or is not configured to offer) native 16 Word16-length-wrapping burst, the WRAPBURST parameter must be cleared, and the GPMC access engine emulates the wrapping burst by issuing the appropriate burst sequences according to the value of ATTACHEDDEVICEPAGELENGTH.

When the memory device does not support native-wrapping burst, there is usually no difference in behavior between a fixed-burst length mode and a continuous-burst mode configuration (except for a potential power increase from a memory-speculative data prefetch in a continuous burst read). However, even though continuous burst mode is compatible with GPMC behavior, because the GPMC access engine issues only fixed-length burst and does not benefit from continuous burst mode, it is best to configure the memory device in fixed-length burst mode.

The memory device maximum-length burst (configured in fixed-length burst wrap or nonwrap mode) usually corresponds to the memory device data buffer size. Memory devices with a minimum of 16 half-word buffers are the most appropriate (especially with wrap support), but memory devices with smaller buffer size (4 or 8) are also supported, assuming that the GPMC_CONFIG1_i[24-23] ATTACHEDDEVICEPAGELENGTH bit field is set accordingly to 4 or 8 words.

The device system issues only requests with addresses or starting addresses for nonwrapping burst requests; that is, the request size boundary is aligned. In case of an eight-word-wrapping burst, the wrapping address always occurs on the eight-word boundary. As a consequence, all words requested must be available from the memory data buffer when the buffer size is equal to or greater than the value of ATTACHEDDEVICEPAGELENGTH. This usually means that data can be read from or written to the buffer at a constant rate (number of cycles between data) without wait-states between data accesses. If the memory does not behave this way (nonzero wait-state burstable memory), WAIT pin monitoring must be enabled to dynamically control data access completion within the burst.

NOTE: When the system burst request length is less than the value of ATTACHEDDEVICEPAGELENGTH, the GPMC proceeds with the required accesses.

7.3.4.11 pSRAM Access Specificities

pSRAM devices are SRAM-pin-compatible low-power memories that contain a self-refreshed DRAM memory array. The GPMC_CONFIG1_i[11-10] DEVICETYPE bit field (where i = 0 to 3) must be set to 0b00.

The pSRAM device uses the NOR protocol. It supports the following operations:

- Asynchronous single read
- Asynchronous page read
- Asynchronous single write
- Synchronous single read and write
- Synchronous burst read
- Synchronous burst write (not supported by NOR flash memory)

pSRAM devices must be powered up and initialized in a predefined manner according to the specifications of the attached device.

pSRAM devices can be programmed to use either mode: fixed or variable latency. pSRAM devices can automatically schedule autorefresh operations, which force the GPMC to use its WAIT signal capability when read or write operations occur during an internal self-refresh operation, or they can automatically include the autorefresh operation in the access time. These devices do not require additional WAIT signal capability or a minimum nCS high pulse width between consecutive accesses to ensure that the correct internal refresh operation is scheduled.

7.3.4.12 NAND Access Description

NOTE: Due to device limitation, NAND functionality of the GPMC is not supported and the NAND functionality described in Section 7.3, *General-Purpose Memory Controller (GPMC)* is not applicable.

NAND (8-bit and 16-bit) memory devices using a standard NAND asynchronous address/data-multiplexing scheme can be supported on any chip-select with the appropriate asynchronous configuration settings.

As for any other type of memory compatible with the GPMC interface, accesses to a chip-select allocated to a NAND device can be interleaved with accesses to chip-selects allocated to other external devices. This interleaved capability limits the system to *chip enable don't care* NAND devices, because the chip-select allocated to the NAND device must be deasserted if accesses to other chip-selects are requested.

7.3.4.12.1 NAND Memory Device in Byte or 16-bit Word Stream Mode

NAND devices require correct command and address programming before data array read or write accesses. The GPMC does not include specific hardware to translate a random address system request into a NAND-specific multiphase access. In that sense, GPMC NAND support, as opposed to random memory-map device support, is data stream-oriented (byte or 16-bit word).

The GPMC NAND programming model depends on a software driver for address and command formatting with the correct data address pointer value according to the block and page structure. Because of NAND structure and protocol interface diversity, the GPMC does not support automatic command and address phase programming, and software drivers must access the NAND device ID to ensure that correct command and address formatting are used for the identified device.

NAND device data read and write accesses are achieved through an asynchronous read or write access. The associated chip-select signal timing control must be programmed according to the NAND device timing specification.

Any chip-select region can be qualified as a NAND region to constrain the nADV/ALE signal as ALE (ALE active high, default state value at low) during address program access, and the nBE0/CLE signal as CLE (CLE active high, default state value at low) during command program access. GPMC address lines are not used (the previous value is not changed) during NAND access.

7.3.4.12.1.1 Chip-Select Configuration for NAND Interfacing in Byte or Word Stream Mode

The GPMC_CONFIG7_i register (where i = 0 to 3) associated with a NAND device region interfaced in byte or word stream mode can be initialized with a minimum size of 16MB, because any address location in the chip-select memory region can be used to access a NAND data array. The NAND flash protocol specifies an address sequence where address bits are passed through the data bus in a series of write accesses with the ALE pin asserted. After this address phase, all operations are streamed and the system requests address is irrelevant.

CAUTION

To allow correct command, address, and data-access controls, the GPMC_CONFIG1_i register associated with a NAND device region must be initialized in asynchronous read and write modes with the parameters listed in Table 7-472. Failure to comply with these settings corrupts the NAND interface protocol.

Table 7-472. Chip-Select Configuration for NAND Interfacing

Bit Field	Register	Value	Comments
WRAPBURST	GPMC_CONFIG1_i[31] (1)	0	No wrap
READMULTIPLE	GPMC_CONFIG1_i[30]	0	Single access
READTYPE	GPMC_CONFIG1_i[29]	0	Asynchronous mode
WRITEMULTIPLE	GPMC_CONFIG1_i[28]	0	Single access
WRITETYPE	GPMC_CONFIG1_i[27]	0	Asynchronous mode
CLKACTIVATIONTIME	GPMC_CONFIG1_i[26-25]	0b00	
ATTACHEDDEVICEPAGELENGTH	GPMC_CONFIG1_i[24-23]	Don't care	Single-access mode
WAITREADMONITORING	GPMC_CONFIG1_i[22]	0	Wait not monitored by GPMC access engine
WAITWRITEMONITORING	GPMC_CONFIG1_i[21]	0	Wait not monitored by GPMC access engine
WAITMONITORINGTIME	GPMC_CONFIG1_i[19-18]	Don't care	Wait not monitored by GPMC access engine

(1) i = 0 to 3

Table 7-472. Chip-Select Configuration for NAND Interfacing (continued)

Bit Field	Register	Value	Comments
WAITPINSELECT	GPMC_CONFIG1_i[17-16]		Select which wait is monitored by edge detectors
DEVICESIZE	GPMC_CONFIG1_i[13-12]	0b00 or 0b01	8- or 16-bit interface
DEVICETYPE	GPMC_CONFIG1_i[11-10]	0b10	NAND device in stream mode
MUXADDDATA	GPMC_CONFIG1_i[9-8]	0b00	non-multiplexed mode
TIMEPARAGRANULARITY	GPMC_CONFIG1_i[4]	0	Timing achieved with best GPMC clock granularity
GPMCFCLKDIVIDER	GPMC_CONFIG1_i[1-0]	Don't care	Asynchronous mode

The GPMC_CONFIG1_i to GPMC_CONFIG4_i registers (where i = 0 to 3) associated with a NAND device region must be initialized with the correct control-signal timing value according to the NAND device timing parameters.

7.3.4.12.1.2 NAND Device Command and Address Phase Control

NAND devices require multiple address programming phases. The software driver must issue the correct number of command and address program accesses, according to the device command set and the device address-mapping scheme.

NAND device-command and address-phase programming is achieved through write requests to the GPMC_NAND_COMMAND_i and GPMC_NAND_ADDRESS_i register locations (where i = 0 to 3) with the correct command and address values. These locations are mapped in the associated chip-select register region. The associated chip-select signal timing control must be programmed according to the NAND device timing specification.

Command and address values are not latched during the access and cannot be read back at the register location.

- Only write accesses must be issued to these locations, but the GPMC does not discard any read access. Accessing a NAND device with nOE and CLE or ALE asserted (read access) can produce undefined results.
- Write accesses to the GPMC_NAND_COMMAND_i and GPMC_NAND_ADDRESS_i register locations must be posted for faster operations (where i = 0 to 3). The GPMC_CONFIG[0] NANDFORCEPOSTEDWRITE bit enables write accesses to these locations as posted, even if they are defined as nonposted.

A write buffer is used to store write transaction information before the external device is accessed:

- Up to eight consecutive posted write accesses can be accepted and stored in the write buffer.
- For nonposted write, the pipeline is one deep.
- An GPMC_STATUS[0] EMPTYWRITEBUFFERSTATUS bit stores the empty status of the write buffer.

The GPMC_NAND_COMMAND_i and GPMC_NAND_ADDRESS_i registers (where i = 0 to 3) are 32-bit word locations, which means any 32-or 16-bit word access is split into 4- or 2-byte accesses if an 8-bit-wide NAND device is attached. For multiple-command phase or multiple-address phase, the software driver can use 32- or 16-bit word access to these registers, but it must consider the splitting and little-endian ordering scheme. When only one byte command or address phase is required, only byte write access to the GPMC_NAND_COMMAND_i and GPMC_NAND_ADDRESS_i registers can be used, and any of the four byte locations of the registers is valid.

The same applies to a GPMC_NAND_COMMAND_i and a GPMC_NAND_ADDRESS_i (where i = 0 to 3) 32-bit word write access to a 16-bit-wide NAND device (split into two 16-bit word accesses). In the case of a 16-bit word write access, the MSByte of the 16-bit word value must be set according to the NAND device requirement (usually 0). Either 16-bit word location or any one of the four byte locations of the registers is valid.

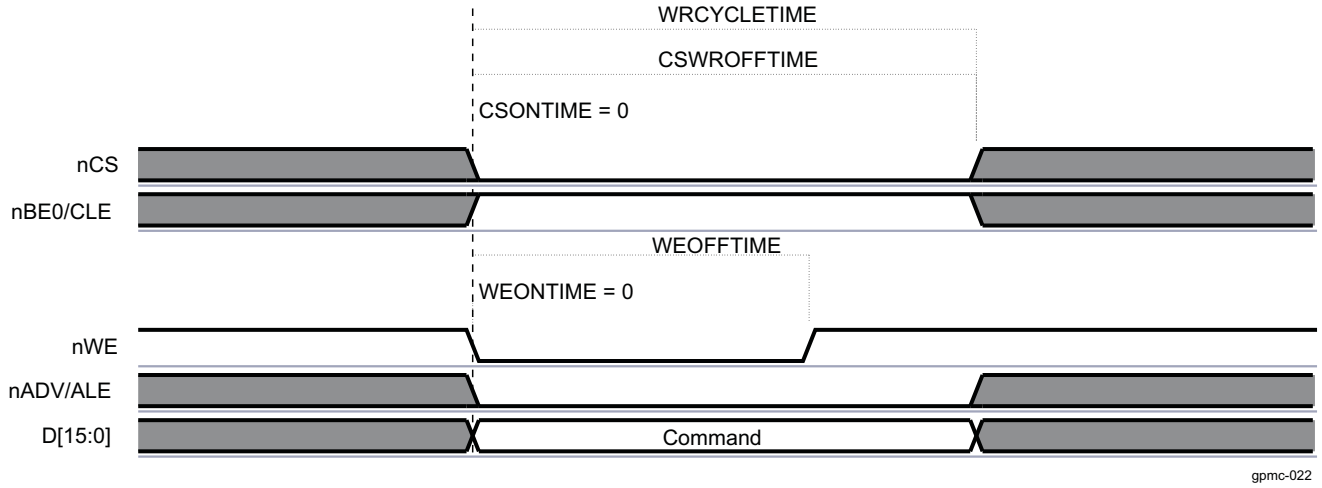
7.3.4.12.1.3 Command Latch Cycle

Writing data at the GPMC_NAND_COMMAND_i location (where i = 0 to 3) places the data as the NAND command value on the bus, using a regular asynchronous write access.

- nCE is controlled by the CSONTIME and CSWROFFTIME timing parameters.
- CLE is controlled by the ADVONTIME and ADVWROFFTIME timing parameters.
- nWE is controlled by the WEONTIME and WEOFFTIME timing parameters.
- ALE and nRE (nOE) are maintained inactive.

Figure 7-185 shows the NAND command latch cycle.

Figure 7-185. NAND Command Latch Cycle



NOTE: CLE is shared with the nBE0 output signal and has an inverted polarity from BE0. The NAND qualifier deals with this. During the asynchronous NAND data access cycle, nBE0 (also nBE1) must not toggle, because it is shared with CLE.

NAND flash memories do not use byte-enable signals.

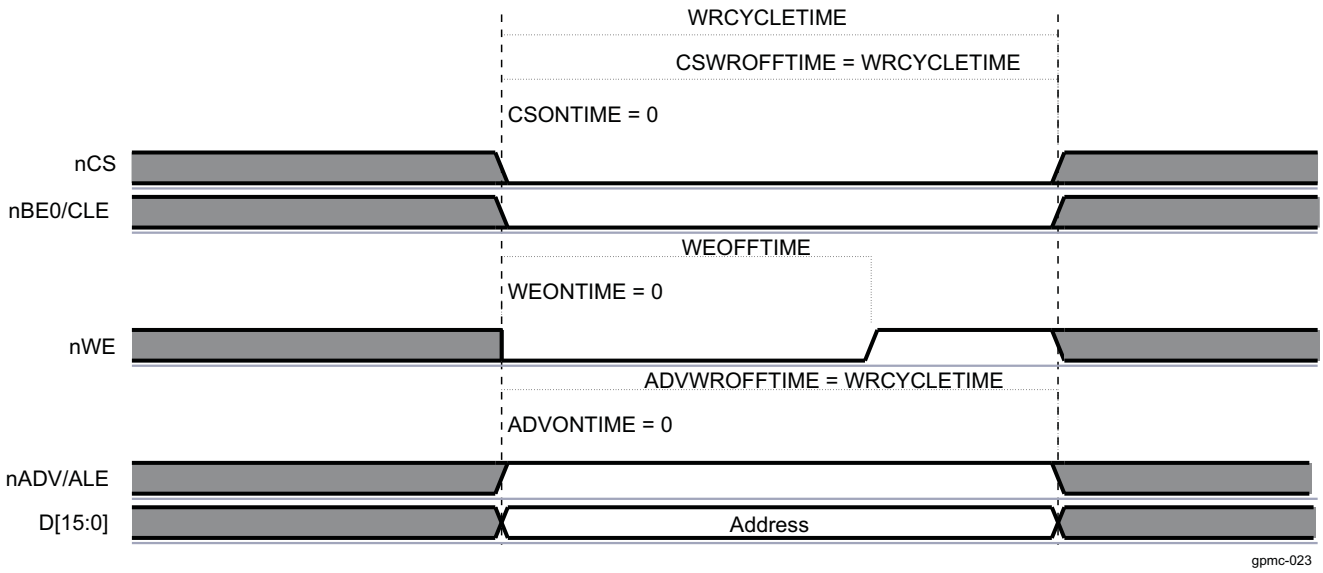
7.3.4.12.1.4 Address Latch Cycle

Writing data at the GPMC_NAND_ADDRESS_i location (where i = 0 to 3) places the data as the NAND partial address value on the bus, using a regular asynchronous write access.

- nCS is controlled by the CSONTIME and CSWROFFTIME timing parameters.
- ALE is controlled by the ADVONTIME and ADVWROFFTIME timing parameters.
- nWE is controlled by the WEONTIME and WEOFFTIME timing parameters.
- CLE and nRE (nOE) are maintained inactive.

Figure 7-186 shows the NAND address latch cycle.

Figure 7-186. NAND Address Latch Cycle



NOTE: ALE is shared with the nADV output signal and has an inverted polarity from ADV. The NAND qualifier deals with this. During the asynchronous NAND data access cycle, ALE is kept stable.

7.3.4.12.1.5 NAND Device Data Read and Write Phase Control in Stream Mode

NAND device data read and write accesses are achieved through a read or write request to the chip-select-associated memory region at any address location in the region or through a read or write request to the GPMC_NAND_DATA_i location (where i = 0 to 3) mapped in the chip-select-associated control register region. GPMC_NAND_DATA_i is not a true register, but an address location to enable nRE or nWE signal control. The associated chip-select signal timing control must be programmed according to the NAND device timing specification.

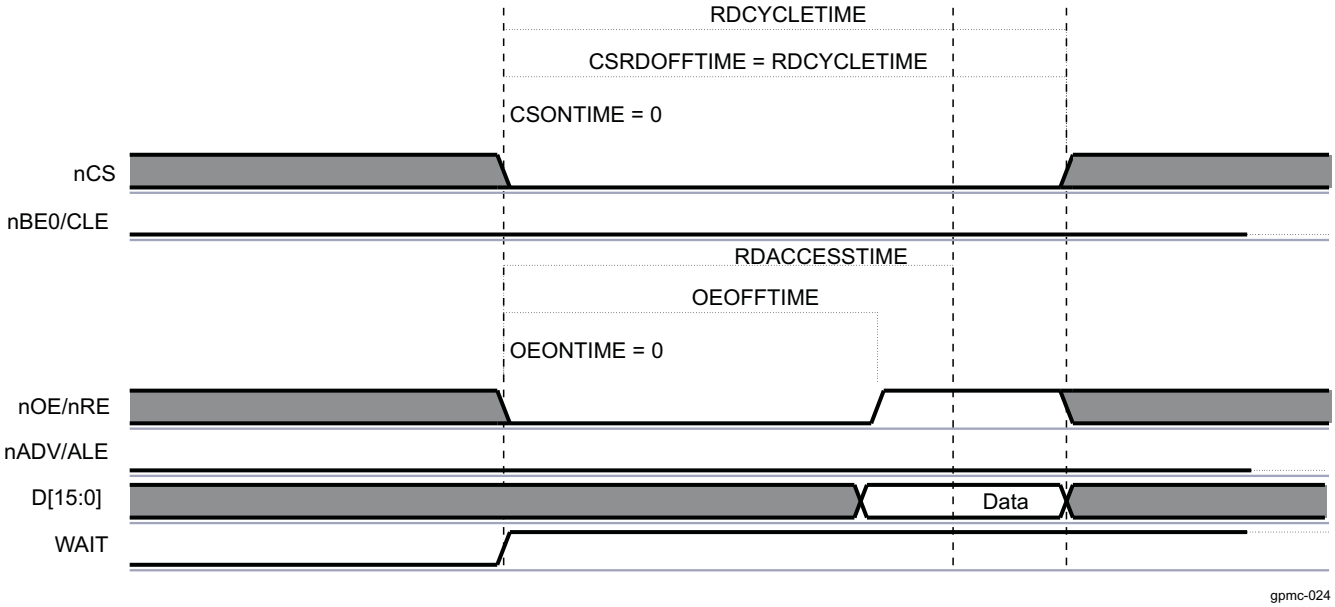
Reading data from the GPMC_NAND_DATA_i location or from any location in the associated chip-select memory region activates an asynchronous read access.

- nCS is controlled by the CSONTIME and CSRDOFFTIME timing parameters.
- nRE is controlled by the OEONTIME and OEOFFTIME timing parameters.
- To take advantage of nRE high-to-data invalid minimum timing value, RDACCESSTIME can be set so that data are effectively captured after nRE deassertion. This allows optimization of NAND read access cycle time completion. For optimal timing parameter settings, see the NAND device and the device timing parameters.

ALE, CLE, and nWE are maintained inactive.

Figure 7-187 shows the NAND data read cycle.

Figure 7-187. NAND Data Read Cycle

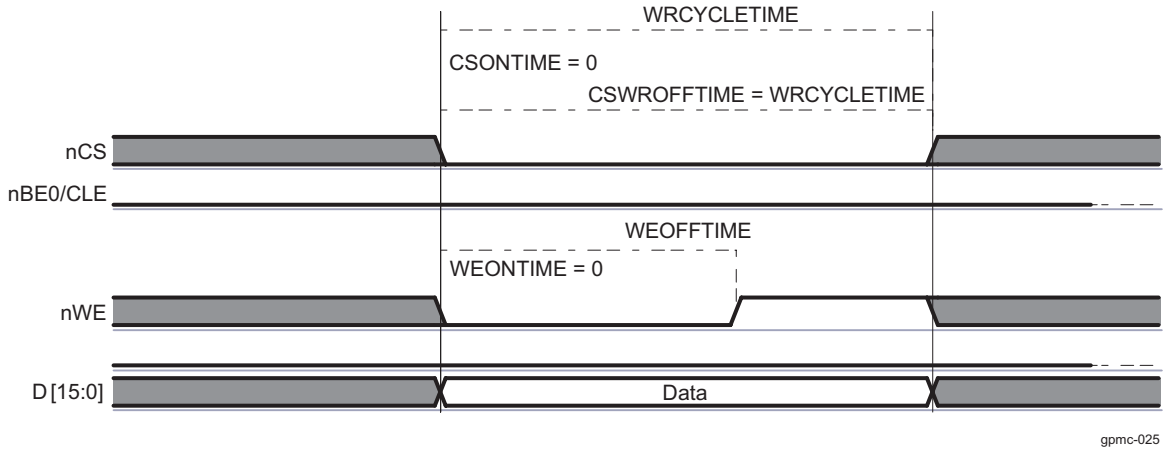


Writing data to the GPMC_NAND_DATA_i location or to any location in the associated chip-select memory region activates an asynchronous write access.

- nCS is controlled by the CSONTIME and CSWROFFTIME timing parameters.
- nWE is controlled by the WEONTIME and WEOFFTIME timing parameters.
- ALE, CLE, and nRE (nOE) are maintained inactive.

Figure 7-188 shows the NAND data write cycle.

Figure 7-188. NAND Data Write Cycle



7.3.4.12.1.6 NAND Device General Chip-Select Timing Control Requirement

For most NAND devices, read data access time is dominated by nCS-to-data-valid timing and has faster nRE-to-data-valid timing. Successive accesses with nCS deassertions between accesses are affected by this timing constraint. Because accesses to a NAND device can be interleaved with other chip-select accesses, there is no certainty that nCS always stays low between two accesses to the same chip-select. Moreover, an nCS deassertion time between the same chip-select NAND accesses is likely to be required as follows: the nCS deassertion requires programming CYCLETIME and RDACCESSTIME according to the nCS-to-data-valid critical timing.

To get full performance from NAND read and write accesses, the prefetch engine can dynamically reduce the following on back-to-back NAND accesses (to the same memory) and suppress the minimum nCS high pulse width between accesses:

- RDCYCLETIME
- WRCYCLETIME
- RDACCESSTIME
- WRACCESSTIME
- CSRDOFFTIME
- CSWROFFTIME
- ADVRDOFFTIME
- ADVWROFFTIME
- OEOFFTIME
- WEOFFTIME

For more information about optimal prefetch engine access, see Section 7.3.4.12.4, *Prefetch and Write-Posting Engine*.

Some NAND devices require minimum write-to-read idle time, especially for device-status read accesses following status-read command programming (write access). If such write-to-read transactions are used, a minimum nCS high pulse width must be set. For this, CYCLE2CYCLESAMECSEN and CYCLE2CYCLEDELAY must be set according to the appropriate timing requirement to prevent any timing violation.

NAND devices usually have an important nRE high-to-data bus in three-state mode. This requires a bus turnaround setting (BUSTURNAROUND = 1) so that the next access to a different chip-select is delayed until the BUSTURNAROUND delay completes. Back-to-back NAND read accesses to the same NAND flash are not affected by the programmed bus turnaround delay.

7.3.4.12.1.7 Read and Write Access Size Adaptation

7.3.4.12.1.7.1 8-Bit-Wide NAND Device

Host 16- and 32-bit word read and write access requests to a chip-select associated with an 8-bit-wide NAND device are split into successive read and write byte accesses to the NAND memory device. Byte access is ordered according to little-endian organization. A NAND 8-bit-wide device must be interfaced on the D0D7 interface bus lane. GPMC data accesses are justified on this bus lane when the cs is associated with an 8-bit-wide NAND device.

7.3.4.12.1.7.2 16-Bit-Wide NAND Device

Host 32-bit word read and write access requests to a chip-select associated with a 16-bit-wide NAND device are split into successive read and write 16-bit word accesses to the NAND memory device. 16-bit word access is ordered according to little-endian organization.

Host byte read and write access requests to a 16-bit-wide NAND device are completed as 16-bit accesses on the device itself, because there is no byte-addressing capability on 16-bit-wide NAND devices. This means that the NAND device address pointer is incremented on a 16-bit word basis and not on a byte basis. For a read access, only the requested byte is given back to the host, but the remaining byte is not stored or saved by the GPMC, and the next byte or 16-bit word read access gets the next 16-bit word NAND location. For a write access, the invalid byte part of the 16-bit word is driven to FF, and the next byte or 16-bit word write access programs the next 16-bit word NAND location.

Generally, byte access to a 16-bit-wide NAND device must be avoided, especially when ECC calculation is enabled. 8- or 16-bit ECC-based computations are corrupted by a byte read to a 16-bit-wide NAND device, because the nonrequested byte is considered invalid on a read access (not captured on the external data bus; FF is fed to the ECC engine) and is set to FF on a write access.

Host requests (read/write) issued in the chip-select memory region are translated in successive single or split accesses (read/write) to the attached device. Therefore, incrementing 32-bit burst requests are translated in multiple 32-bit sequential accesses following the access adaptation of the 32-bit to 8- or 16-bit device.

7.3.4.12.2 NAND Device-Ready Pin

The NAND memory device provides a ready pin to indicate data availability after a block/page opening and to indicate that data programming is complete. The ready pin can be connected to one of the wait GPMC input pins; data read accesses must not be tried when the ready pin is sampled inactive (device is not ready) even if the associated chip-select WAITREADMONITORING bit field is set. The duration of the NAND device busy state after the block/page opening is so long (up to 50 micro second) that accesses occurring when the ready pin is sampled inactive can stall GPMC access and eventually cause a system time-out.

NOTE: If a read access to a NAND flash is done using WAIT monitoring mode, the device is blocked during a page opening, and so is the GPMC. If the correct settings are used, other chip-selects can be used while the memory processes the page opening command.

To avoid a time-out caused by a block/page opening delay in NAND flash, disable the WAIT pin monitoring for read and write accesses (that is, set the GPMC_CONFIG1_i[21] WAITWRITEMONITORING and GPMC_CONFIG1_i[22] WAITREADMONITORING bits to 0, where i = 0 to 3), and use one of the following methods instead:

- Use software to poll the WAITxSTATUS bit (where x = 0 to 2) of the GPMC_STATUS .
- Configure an interrupt that is generated on the WAIT signal change (through the GPMC_IRQENABLE register bits[11-8]).

Even if the READWAITMONITORING bit is not set, the external memory nR/B pin status is captured in the programmed wait bit in the GPMC_STATUS register.

The READWAITMONITORING bit method must be used for other memories than NAND flash, if they require the use of a WAIT signal.

7.3.4.12.2.1 Ready Pin Monitored by Software Polling

The ready signal state can be monitored through the GPMC_STATUS WAITxSTATUS bit (where x = 0 to 2). Software must monitor the ready pin only when the signal is declared valid. Refer to the NAND device timing parameters to set the correct software temporization to monitor ready only after the invalid window is complete from the last read command written to the NAND device.

7.3.4.12.2.2 Ready Pin Monitored by Hardware Interrupt

Each GPMC_WAIT input pin can generate an interrupt when a wait-to-no-wait transition is detected. Depending on whether the GPMC_CONFIG WAITxPINPOLARITY bits (where x = 0 to 2) is active low or active high, the wait-to-no-wait transition is a low-to-high external WAIT signal transition or a high-to-low external WAIT signal transition, respectively.

The wait transition pin detector must be cleared before any transition detection. This is done by writing 1 to the WAITxEDGEDETECTIONSTATUS bit (where x = 0 to 2) of the GPMC_IRQSTATUS register according to the GPMC_WAIT pin used for the NAND device-ready signal monitoring. To detect a wait-to-no-wait transition, the transition detector requires a wait active time detection of a minimum of two GPMC_FCLK cycles. Software must incorporate precautions to clear the wait transition pin detector before wait (busy) time completes.

A wait-to-no-wait transition detection can issue a GPMC interrupt if the WAITxEDGEDETECTIONENABLE bit in the GPMC_IRQENABLE register is set and if the WAITxEDGEDETECTIONSTATUS bit field in the GPMC_IRQSTATUS register is set.

The WAITMONITORINGTIME bit field does not affect wait-to-no-wait transition time detection.

It is also possible to poll the WAITxEDGEDETECTIONSTATUS bit field in the GPMC_IRQSTATUS register according to the GPMC_WAIT pin used for NAND device ready signal monitoring.

7.3.4.12.3 ECC Calculator

The GPMC includes an error code correction (ECC) calculator circuitry that enables ECC calculation on the fly during data read or data program (that is, write) operations. The page size supported by the ECC calculator in one calculation/context is 512 bytes.

The user can choose from two different algorithms with different error correction capabilities through the GPMC_ECC_CONFIG[16] ECCALGORITHM bit:

1. Hamming code for 1-bit error code correction on 8- or 16-bit NAND flash organized with page size greater than 512 bytes
2. Bose-Chaudhurl-Hocquenghem (BCH) code for 4- to 16-bit error correction

The GPMC does not handle the error code correction directly. During writes, the GPMC computes parity bits. During reads, the GPMC provides enough information for the processor to correct errors without reading the data buffer all over again.

The Hamming code ECC is based on a 2-dimensional (2D) (row and column) bit parity accumulation. This parity accumulation is accomplished on the programmed number of bytes or 16-bit words read from the memory device, or is written to the memory device in stream mode.

Because the ECC engine includes only one accumulation context, it can be allocated to only one chip-select at a time through the GPMC_ECC_CONFIG[3-1] ECCCS bit field. Even if two chip-selects use different ECC algorithms, one the Hamming code and the other a BCH code, they must define separate ECC contexts because some of the ECC registers are common to all types of algorithms.

7.3.4.12.3.1 Hamming Code

All references to ECC in this subsection refer to the 1-bit error correction Hamming code.

The ECC is based on a 2D (row and column) bit parity accumulation known as the Hamming code. The parity accumulation is done for a programmed number of bytes or 16-bit word read from the memory device or written to the memory device in stream mode.

There is no automatic error detection or correction, and the software NAND driver must read the multiple ECC calculation results, compare them to the expected code value, and take the appropriate corrective actions according to the error handling strategy (ECC storage in spare byte, error correction on read, block invalidation).

The ECC engine includes a single accumulation context. It can be allocated to a single designated chip-select at a time, and parallel computations on different chip-selects are not possible. Because it is allocated to a single chip-select, the ECC computation is not affected by interleaved GPMC accesses to other chip-selects and devices. The ECC accumulation is sequentially processed in the order of data read from or written to the memory on the designated chip-select. The ECC engine does not differentiate read accesses from write accesses and does not differentiate data from command or status information. Software must ensure that only relevant data are passed to the NAND flash memory while the ECC computation engine is active.

The starting NAND page location must be programmed first, followed by an ECC accumulation context reset with an ECC enabling, if required. The NAND device accesses discussed in the following sections must be limited to data read or write until the specified number of ECC calculations is complete.

7.3.4.12.3.1.1 ECC Result Register and ECC Computation Accumulation Size

The GPMC includes up to nine ECC result registers (GPMC_ECCj_RESULT, where j = 1 to 9) to store ECC computation results when the specified number of bytes or 16-bit words has been computed.

The ECC result registers are used sequentially: one ECC result is stored in one ECC result register on the list, the next ECC result is stored in the next ECC result register on the list, and so forth, until the last ECC computation. The value of the GPMC_ECCj_RESULT register is valid only when the programmed number of bytes or 16-bit words has been accumulated, which means that the same number of bytes or 16-bit words has been read from or written to the NAND device in sequence.

The GPMC_ECC_CONTROL[3-0] ECCPOINTER bit field must be set to the correct value to select the ECC result register to be used first in the list for the incoming ECC computation process. The ECCPointer can be read to determine which ECC register is used in the next ECC result storage for the ongoing ECC computation. The value of the GPMC_ECCj_RESULT register (where j = 1 to 9) can be considered valid when ECCPOINTER equals j + 1. When the GPMC_ECCj_RESULT register (where j = 9) is updated, ECCPOINTER is frozen at 10, and ECC computing is stopped (ECCENABLE = 0).

The ECC accumulator must be reset before any ECC computation accumulation process. The GPMC_ECC_CONTROL[8] ECCCLEAR bit must be set to 1 (nonpersistent bit) to clear the accumulator and all ECC result registers.

For each ECC result (each GPMC_ECCj_RESULT register, where j = 1 to 9), the number of bytes or 16-bit words used for ECC computing accumulation can be selected from between two programmable values.

The ECCjRESULTSIZE bits (where j = 1 to 9) in the GPMC_ECC_SIZE_CONFIG register select which programmable size value (ECCSIZE0 or ECCSIZE1) must be used for this ECC result (stored in the GPMC_ECCj_RESULT register).

The ECCSIZE0 and ECCSIZE1 bit fields allow selection of the number of bytes or 16-bit words used for ECC computation accumulation. Any even values from 2 to 512 are allowed.

Flexibility in the number of ECCs computed and the number of bytes or 16-bit words used in the successive ECC computations enables different NAND page error-correction strategies. Usually based on 256 or 512 bytes and on 128 or 256 16-bit word, the number of ECC results required is a function of the NAND device page size. Specific ECC accumulation size can be used when computing the ECC on the NAND spare byte.

For example, with a 2-KB data page, 8-bit-wide NAND device, eight ECCs accumulated on 256 bytes can be computed and added to one extra ECC computed on the 24 spare bytes area where the eight ECC results used for comparison and correction with the computed data page ECC are stored. The GPMC then provides nine GPMC_ECCj_RESULT registers (j = 1 to 9) to store the results. In this case, ECCSIZE0 is set to 256, and ECCSIZE1 is set to 24; the ECC[1-8] RESULTSIZE bits are set to 0, and the ECC9RESULTSIZE bit is set to 1.

7.3.4.12.3.1.2 ECC Enabling

The GPMC_ECC_CONFIG[3-1] ECCCS bit field selects the allocated chip-select. The GPMC_ECC_CONFIG[0] ECCENABLE bit enables ECC computation on the next detected read or write access to the selected chip-select.

The following fields must not be changed or cleared while an ECC computation is in progress:

- CCPOINTER
- ECCCLEAR
- ECCSIZE
- ECCjRESULTSIZE (where j = 1 to 9)
- ECC16B
- ECCCS

The ECC accumulator and ECC result register must not be changed or cleared while an ECC computation is in progress.

Table 7-473 describes the ECC enable settings.

Table 7-473. ECC Enable Settings

Bit Field	Register	Value	Comments
ECCCS	GPMC_ECC_CONFIG	0–3	Selects the chip-select where ECC is computed
ECC16B	GPMC_ECC_CONFIG	0/1	Selects column number for ECC calculation
ECCCLEAR	GPMC_ECC_CONTROL	0–7	Clears all ECC result registers
ECCPOINTER	GPMC_ECC_CONTROL	0–7	A write to this bit field selects the ECC result register where the first ECC computation is stored. Set to 1 by default.

Table 7-473. ECC Enable Settings (continued)

Bit Field	Register	Value	Comments
ECCSIZE1	GPMC_ECC_SIZE_CONFIG	00h–FFh	Defines ECCSIZE1
ECCSIZE0	GPMC_ECC_SIZE_CONFIG	00h–FFh	Defines ECCSIZE0
ECCjRESULTSIZE (j from 1 to 9)	GPMC_ECC_SIZE_CONFIG	0/1	Selects the size of ECCn result register
ECCENABLE	GPMC_ECC_CONFIG	1	Enables the ECC computation

7.3.4.12.3.1.3 ECC Computation

The ECC algorithm is a multiple parity bit accumulation computed on the odd and even bit streams extracted from the byte or Word16 streams. The parity accumulation is split into row and column accumulations, as shown in Figure 7-189 and Figure 7-190. The intermediate row and column parities are used to compute the upper level row and column parities. Only the final computation of each parity bit is used for ECC comparison and correction.

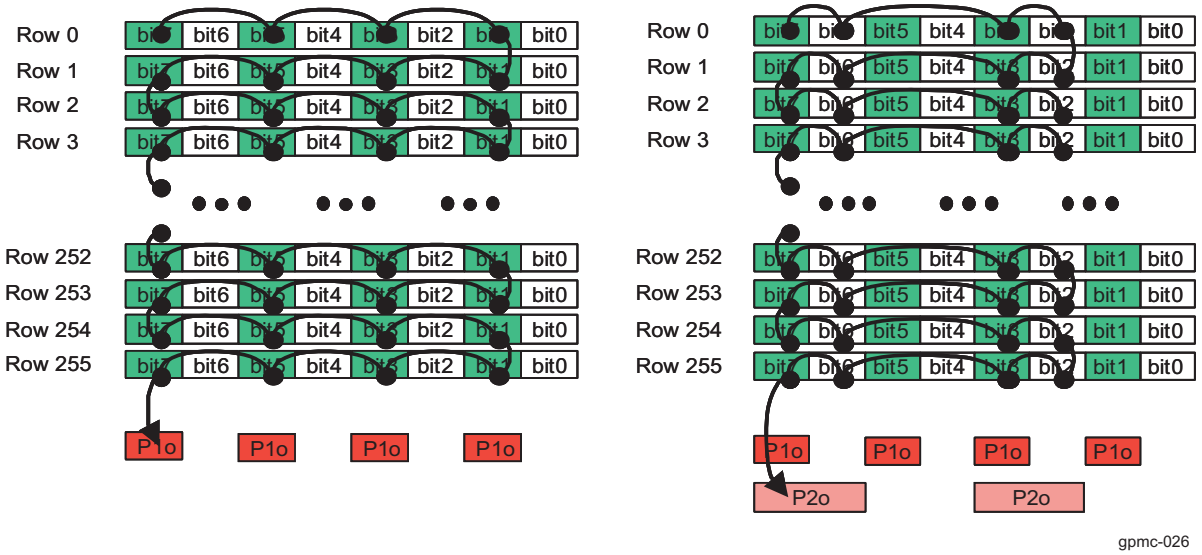
P1o = bit7 XOR bit5 XOR bit3 XOR bit1 on each byte of the data stream
P1e = bit6 XOR bit4 XOR bit2 XOR bit0 on each byte of the data stream

P2o = bit7 XOR bit6 XOR bit3 XOR bit2 on each byte of the data stream
P2e = bit5 XOR bit4 XOR bit1 XOR bit0 on each byte of the data stream

P4o = bit7 XOR bit6 XOR bit5 XOR bit4 on each byte of the data stream
P4e = bit3 XOR bit2 XOR bit1 XOR bit0 on each byte of the data stream

Each column parity bit is XORed with the previous accumulated value.

Figure 7-189. Hamming Code Accumulation Algorithm (1/2)

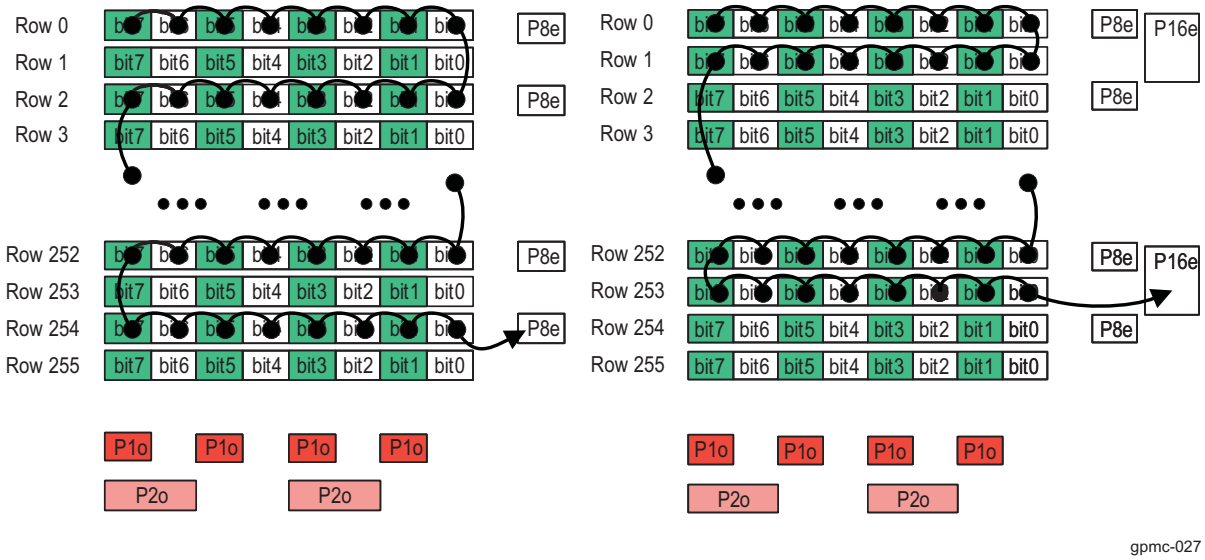


For line parities, the bits of each new data are XORed together, and line parity bits are computed as described below:

P8e = row0 XOR row2 XOR row4 XOR ... XOR row254
P8o = row1 XOR row3 XOR row5 XOR ... XOR row255

P16e = row0 XOR row1 XOR row4 XOR row5 XOR ... XOR row252 XOR row 253
P16o = row2 XOR row3 XOR row6 XOR row7 XOR ... XOR row254 XOR row 255

Figure 7-190. Hamming Code Accumulation Algorithm (2/2)



Unused parity bits in the result registers are set to 0.

Figure 7-191 shows ECC computation for a 256-byte data stream (read or write). The result includes six column parity bits (P1o-P2o-P4o for odd parities, and P1e-P2e-P4e for even parities) and sixteen row parity bits (P8o-P16o-P32o--P1024o for odd parities, and P8e-P16e-P32e--P1024e for even parities).

Figure 7-191. ECC Computation for a 256-Byte Data Stream (Read or Write)

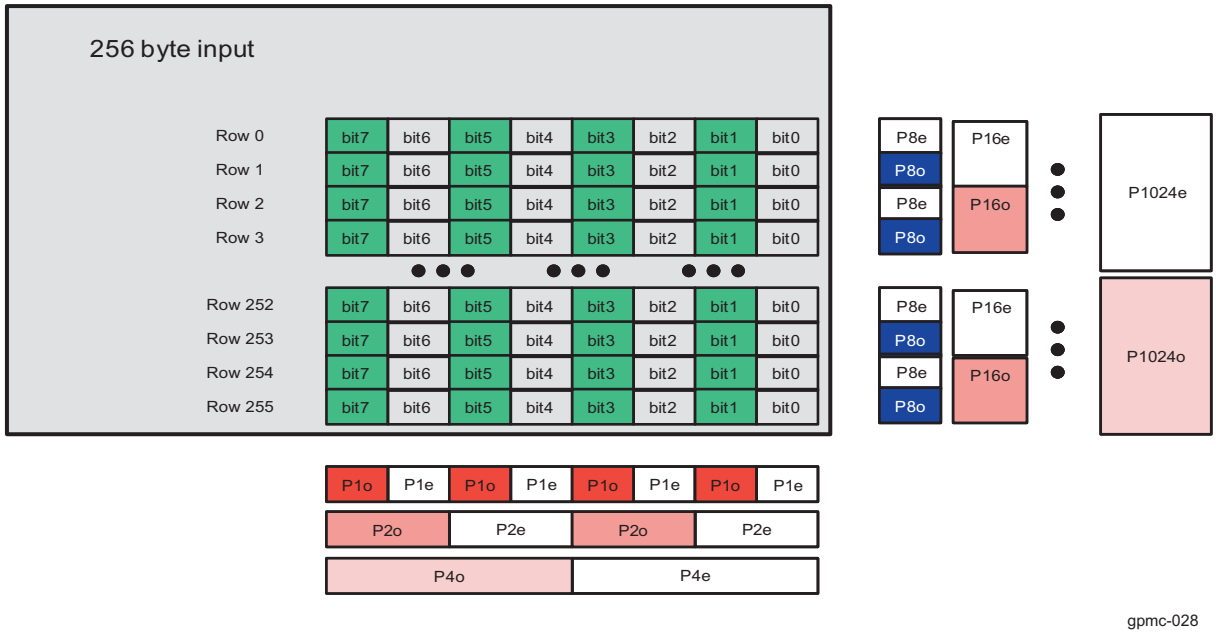
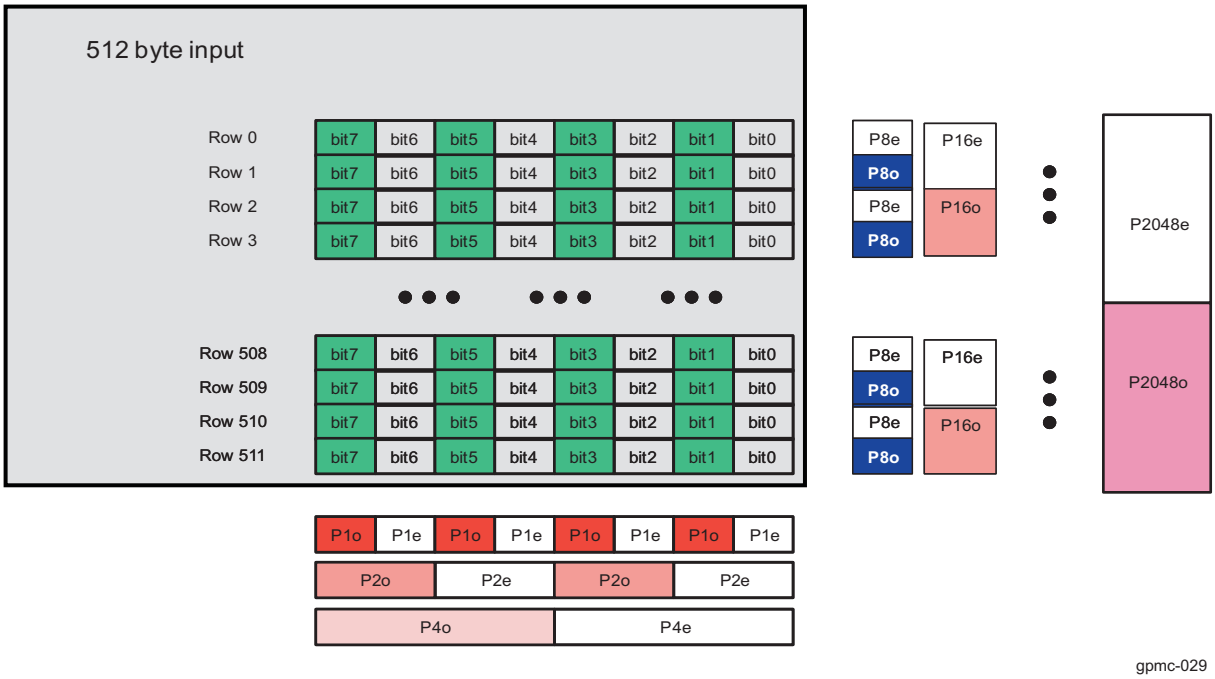


Figure 7-192 shows ECC computation for a 512-byte data stream (read or write). The result includes six column parity bits (P1o-P2o-P4o for odd parities, and P1e-P2e-P4e for even parities) and eighteen row parity bits (P8o-P16o-P32o--P1024o- - P2048o for odd parities, and P8e-P16e-P32e--P1024e- P2048e for even parities).

Figure 7-192. ECC Computation for a 512-Byte Data Stream (Read or Write)



For a 2-KB page, four 512 bytes ECC calculations plus 1 for the spare area are required. Results are stored in the GPMC_ECCj_RESULT registers (where j = 1 to 9).

7.3.4.12.3.1.4 ECC Comparison and Correction

To detect an error, the computed ECC result must be XORed with the parity value stored in the spare area of the accessed page.

- If the result of this logical XOR is all 0s, no error is detected and the read data is correct.
- If every second bit in the parity result is a 1, 1 bit is corrupted and is located at bit address (P2048o, P1024o, P512o, P256o, P128o, P64o, P32o, P16o, P8o, P4o, P2o, P1o). Software must correct the corresponding bit.
- If only 1 bit in the parity result is 1, it is an ECC error and the read data is correct.

7.3.4.12.3.1.5 ECC Calculation Based on 8-Bit Word

The 8-bit-based ECC computation is used for 8-bit-wide NAND device interfacing.

The 8-bit-based ECC computation can be used for 16-bit-wide NAND device interfacing to get backward compatibility on the error-handling strategy used with 8-bit-wide NAND devices. In this case, the 16-bit-wide data read from or written to the NAND device is fragmented into 2 bytes. According to little-endian access, the LSB of the 16-bit-wide data is ordered first in the byte stream used for 8-bit-based ECC computation.

7.3.4.12.3.1.6 ECC Calculation Based on 16-Bit Word

ECC computation based on an 16-bit word is used for 16-bit-wide NAND device interfacing. This ECC computation is not supported when interfacing an 8-bit-wide NAND device, and the GPMC_ECC_CONFIG[7] ECC16B bit must be set to 0 when interfacing an 8-bit-wide NAND device.

The parity computation based on 16-bit words affects the row and column parity mapping. The main difference is that the odd and even parity bits P8o and P8e are computed on rows for an 8-bit-based ECC and on columns for a 16-bit based ECC. Figure 7-193 and Figure 7-194 show a 128 Word16 ECC computation scheme and a 256 16-bit word ECC computation scheme.

Figure 7-193. 128 Word16 ECC Computation

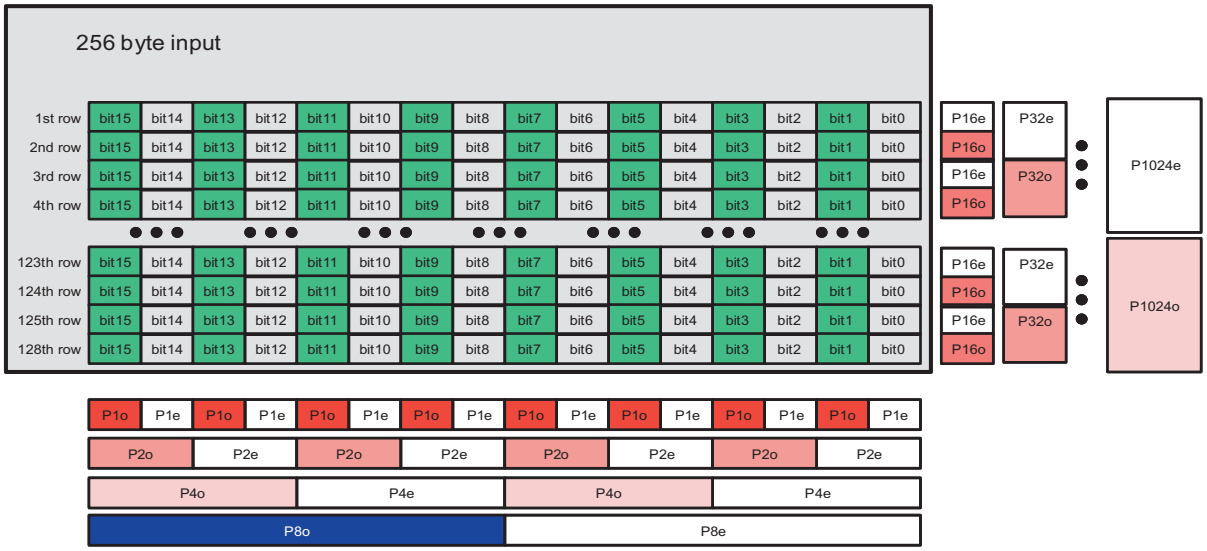
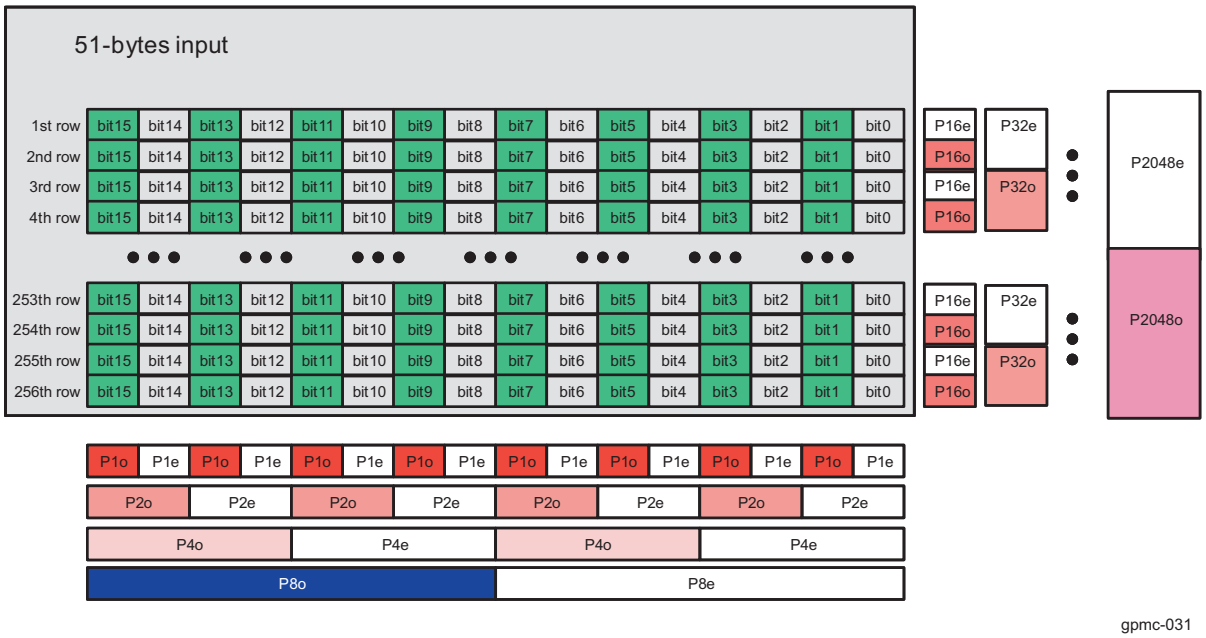


Figure 7-194. 256 Word16 ECC Computation



7.3.4.12.3.2 BCH Code

All references to ECC in this subsection refer to the 4- to 16-bit error correction BCH code.

7.3.4.12.3.2.1 Requirements

1. Read and write accesses to a NAND flash take place by whole pages, in a predetermined sequence: first the data byte page itself, and then some spare bytes, including the BCH ECC (and other information). The NAND device can cache a full page, including spares, for read and write accesses.
   - Typical page write sequence:
     - Sequential write to NAND cache of main data plus spare data, for a page. ECC is calculated on the fly. Calculated ECC can be inserted on the fly in the spares or replaced by dummy

accesses.
- When the calculated ECC is replaced by dummy accesses, it must be written to the cache in a second, separate phase. The ECC module is disabled during that time.
- NAND writes its cache line (page) to the array.

- Typical page read sequence:
  - Sequential read of a page. ECC is calculated on the fly.
  - The status of the ECC module buffers determines the presence of errors.

2. Accesses to several memories can be interleaved by the GPMC, but only one of those memories at a time can be a NAND using the BCH engine; in other words, only one BCH calculation (for example, for a single page) can be ongoing at any time. The sequential nature of NAND accesses ensures that the data is always written or read out in the same order. BCH-relevant accesses are selected by the chip-selects of the GPMC.
3. Each page can hold up to 4KB of data, spare bytes not included. This means up to 8 × 512-byte BCH messages. Because all the data is written or read out first, followed by the BCH ECC, the BCH engine must be able to hold eight 104-bit remainders or syndromes (or smaller, 52-bit ones) at the same time.

   The BCH module can store all remainders internally. After the page start, an internal counter is used to detect the 512-byte sector boundaries. On those boundaries, the current remainder is stored and the divider reset for the next calculation. At the end of the page, the BCH module contains all remainders.
4. NAND access cycles hold 8 or 16 bits of data each (1 or 2 bytes); Each NAND cycle takes at least four cycles of the GPMC internal clock. This means the NAND flash timing parameters must define a RDCYCLETIME and a WRCYCLETIME of at least four clock cycles after optimization when using the BCH calculator.
5. The spare area is assumed to be large enough to hold the BCH ECC; that is, to have a message of at least 13 bytes available per 512-byte sector of data. The zone of unused spare area by the ECC may or may not be protected by the same ECC scheme, by extending the BCH message beyond 512 bytes (the maximum codeword is 1023 bytes long, ECC included, which leaves much space to cover spare bytes).

7.3.4.12.3.2.2 Memory Mapping of BCH Codeword

BCH encoding considers a block of data to protect as a polynomial message M(x). In a standard case, 512 bytes of data (that is, 2^{12} bits = 4096 bits) are seen as a polynomial of degree $2^{12} – 1 = 4095$, with parameters ranging from M0 to M4095. For 512 bytes of data, 52 bits are required for 4-bit error correction, 104 bits are required for 8-bit error correction, and 207 bits are required for 16-bit error correction. The ECC is a remainder polynomial R(x) of degree 103 (or 51, depending on the selected mode). The complete codeword C(x) is the concatenation of M(x) and R(x), as described in Table 7-474.

Table 7-474. Flattened BCH Codeword Mapping (512 Bytes + 104 Bits)

	Message M(x)			ECC R(x)		
Bit Number	M4095	...	M0	R103	...	R0

If the message is extended by the addition of spare bytes to be protected by the same ECC, the principle is still valid. For example, a 3-byte extension of the message gives a polynomial message M(x) of degree ((512 + 3) × 8) – 1 = 4119, for a total of 3 + 13 = 16 spare bytes of spare, all protected as part of the same codeword.

The message and the ECC bits are manipulated and mapped in the GPMC byte-oriented system. The ECC bits are stored in the following registers (where i = 0 to 3):

- GPMC_BCH_RESULT0_i
- GPMC_BCH_RESULT1_i
- GPMC_BCH_RESULT2_i
- GPMC_BCH_RESULT3_i

7.3.4.12.3.2.2.1 Memory Mapping of Data Message

The data message mapping must follow the following rules:

- Bit endianness within a byte is little-endian; that is, the bytes LSB is also the lowest-degree polynomial parameter: a byte b7-b0 (with b0 the LSB) represents a segment of polynomial $b7 * x^{(7+i)} + b6 * x^{(6+i)} + ... + b0 * x^{i}$
- The message is mapped in the NAND starting with the highest-order parameters, that is, in the lowest addresses of a NAND page.
- Byte endianness within the 16-bit words in the NAND is big-endian (that is, the same message mapped in 8- and 16-bit memories has the same content at the same byte address).

NOTE: The BCH module has no visibility over actual addresses. The most important point is the sequence of data words the BCH sees. However, the NAND page is always scanned incrementally in read and write accesses, which produces the mapping patterns described in the following.

Table 7-475 and Table 7-476 describe the mapping of the same 512-byte vector (typically, a BCH message) in the NAND memory space. The byte address is only an offset modulo 512 (200h), because the same page may contain several contiguous 512-byte sectors (BCH blocks). The LSB and MSB are, respectively, the bits M0 and $M(2^{12}–1)$ of the codeword mapping discussed previously. In both cases the data vectors are aligned; that is, their boundaries coincide with the RAM data word boundaries.

Table 7-475. Aligned Message Byte Mapping in 8-Bit NAND

Byte Offset	8-Bit Word
000h	(MSB) Byte 511 (1FFh)
001h	Byte 510 (1FEh)
...	...
1FFh	Byte 0 (0h) (LSB)

Table 7-476. Aligned Message Byte Mapping in 16-Bit NAND

Byte Offset	16-Bit Word MSB	16-Bit Word LSB
000h	Byte 510 (1FEh)	(MSB) Byte 511 (1FFh)
002h	Byte 508 (1FCh)	Byte 509 (1FDh)
...	...	...
1FEh	Byte 0 (0h)	(LSB) Byte 1 (1h)

Table 7-477 through Table 7-482 list the mapping in memory of arbitrarily-sized messages, starting on access (byte or 16-bit word) boundaries for more clarity. Note that message may actually start and stop on arbitrary nibbles. A nibble is a 4-bit entity. The unused nibbles are not discarded, and they can still be used by the BCH module, but as part of the next message section (for example, on the ECC of another sector).

Table 7-477. Aligned Nibble Mapping of Message in 8-Bit NAND

Byte Offset	8-Bit Word	
	4-Bit Most Significant Nibble	4-Bit Least Significant Nibble
1	(MSB) Nibble S-1	Nibble S-2
2	Nibble S-3	Nibble S-4
...	...	...
S/2 – 2	Nibble 3	Nibble 2
S/2 – 1	Nibble 1	Nibble 0 (LSB)

Table 7-478. Misaligned Nibble Mapping of Message in 8-Bit NAND

Byte Offset	8-Bit Word	
	4-Bit Most Significant Nibble	4-Bit Least Significant Nibble
1	(MSB) Nibble S-1	Nibble S-2
2	Nibble S-3	Nibble S-4
...	...	...
(S + 1) / 2 – 2	Nibble 2	Nibble 1
(S + 1) / 2 – 1	Nibble 0 (LSB)	

Table 7-479. Aligned Nibble Mapping of Message in 16-Bit NAND

Byte Offset	16-Bit Word			
	4-Bit Most Significant Nibble		4-Bit Least Significant Nibble	
0	Nibble S-3	Nibble S-4	(MSB) Nibble S-1	Nibble S-2
2	Nibble S-7	Nibble S-8	Nibble S-5	Nibble S-6
...	...		...	
S/2 – 4	Nibble 5	Nibble 4	Nibble 7	Nibble 6
S/2 – 2	Nibble 1	Nibble 0 (LSB)	Nibble 3	Nibble 2

Table 7-480. Misaligned Nibble Mapping of Message in 16-Bit NAND (1 Unused Nibble)

Byte Offset	16-Bit Word			
	4-Bit Most Significant Nibble		4-Bit Least Significant Nibble	
0	Nibble S-3	Nibble S-4	(MSB) Nibble S-1	Nibble S-2
2	Nibble S-7	Nibble S-8	Nibble S-5	Nibble S-6
...	...		...	
(S + 1) / 2 – 4	Nibble 4	Nibble 3	Nibble 6	Nibble 5
(S + 1) / 2 – 2	Nibble 0 (LSB)		Nibble 2	Nibble 1

Table 7-481. Misaligned Nibble Mapping of Message in 16-Bit NAND (2 Unused Nibbles)

Byte Offset	16-Bit Word			
	4-Bit Most Significant Nibble		4-Bit Least Significant Nibble	
0	Nibble S-3	Nibble S-4	(MSB) Nibble S-1	Nibble S-2
2	Nibble S-7	Nibble S-8	Nibble S-5	Nibble S-6
...	...		...	
(S + 2) / 2 – 4	Nibble 3	Nibble 2	Nibble 5	Nibble 4
(S + 2) / 2 – 2			Nibble 1	Nibble 0 (LSB)

Table 7-482. Misaligned Nibble Mapping of Message in 16-Bit NAND (3 Unused Nibbles)

Byte Offset	16-Bit Word			
	4-Bit Most Significant Nibble		4-Bit Least Significant Nibble	
0	Nibble S-3	Nibble S-4	(MSB) Nibble S-1	Nibble S-2
2	Nibble S-7	Nibble S-8	Nibble S-5	Nibble S-6
...	...		...	
(S + 3) / 2 – 4	Nibble 2	Nibble 1	Nibble 4	Nibble 3
(S + 3) / 2 – 2			Nibble 0 (LSB)	

Many other cases exist than those given in the previous tables; for example, where the message does not start on a word boundary.

7.3.4.12.3.2.2.2 Memory-Mapping of the ECC

The ECC (or remainder) is presented by the BCH module as a single 104-bit (or 52-bit), little-endian vector. Software must fetch those 13 bytes (or 6 bytes) from the module interface and then store them to the spare area (page write) in the NAND or to an intermediate buffer for comparison with the stored ECC (page read). There are no constraints on the ECC mapping inside the spare area: it is software-controlled.

It is advised, however, to maintain a coherence in the respective formats of the message or the ECC remainder once they have been read out of the NAND. The error correction algorithm works from the complete codeword (concatenated message and remainder) once an error is detected. The creation of this codeword must be made as straightforward as possible.

There are cases in which the same NAND access contains both data and the ECC protecting that data. This is the case when the data/ECC boundary (which can be on any nibble) does not coincide with an access boundary. The ECC is calculated on the fly following the write. In that case, the write must also contain part of the ECC because it is impossible to insert the ECC on the fly. Instead:

- During the initial page write (BCH encoding), the ECC is replaced by dummy bits. The BCH encoder is by definition turned OFF during the ECC section, so the BCH result is unmodified.
- During a second phase, the ECC is written to the correct location, next to the actual data.
- The completed line buffer is then written to the NAND array.

7.3.4.12.3.2.2.3 Wrapping Modes

For a given wrapping mode, the module automatically goes through a specific number of sections as data is being fed into the module. For each section, the BCH core can be enabled (in which case the data is fed to the BCH divider) or not (in which case the BCH simply counts to the end of the section). When enabled, the data is added to the ongoing calculation for a given sector number (for example, number 0).

Wrapping modes are described as follows. To better understand and see the real-life read and write sequences implemented with each mode, see Section 7.3.4.12.3.2.3, *Supported NAND Page Mappings and ECC Schemes*.

For each mode:

- A sequence describes the mode in pseudo language, with, for each section, the size and the buffer used for ECC processing (if ON). The programmable lengths are size, size0, and size1.
- A checksum condition is given. If the checksum condition is not respected for a given mode, the module behavior is unpredictable. S is the number of sectors in the page; size0 and size1 are the section sizes programmed for the mode, in nibbles.

Wrapping modes 8 through 11 insert a 1-nibble padding where the BCH processing is OFF. This is intended for t = 4 ECC, where ECC is 6 bytes long and the ECC area is expected to include (at least) 1 unused nibble to remain byte-aligned.

7.3.4.12.3.2.2.4 Manual Mode (0h)

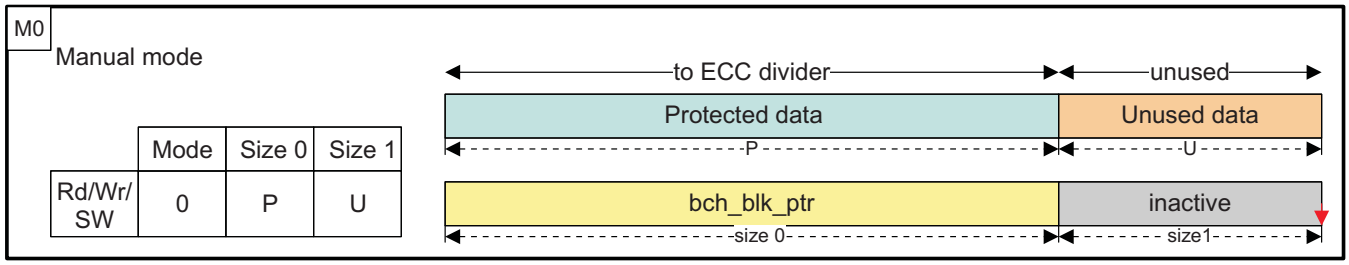
This mode is intended for short sequences, added manually to a given buffer through the software data port input. A complete page may be built out of several such sequences.

To process an arbitrary sequence of 4-bit nibbles, accesses to the software data port, containing the appropriate data, must be made. If the sequence end does not coincide with an access boundary (for example, to process 5 nibbles = 20 bits in 16-bit access mode) and those nibbles must be skipped, a number of unused nibbles must be programmed in GPMC_ECC_SIZE_CONFIG[29-22] ECCSIZE1 In the same example, 5 nibbles to process + 3 to discard = 8 nibbles = 2 × 16-bit accesses. Software must set:

- GPMC_ECC_SIZE_CONFIG[19-12] ECCSIZE0 = 5h
- GPMC_ECC_SIZE_CONFIG[29-22] ECCSIZE1 = 3h

NOTE: In the following figures, size and size0 are the same parameter.

Figure 7-195. Manual Mode Sequence and Mapping



gpmc-032

Section processing sequence:

- One time with buffer
  - size0 nibbles of data, processing ON
  - size1 nibbles of unused data, processing OFF

Checksum: size0 + size1 nibbles must fit in a whole number of accesses.

In the following sections, S is the number of sectors in the page.

7.3.4.12.3.2.2.5 Mode 1h

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- Repeat with buffer 0 to S-1
  - size0 nibbles spare, processing ON
  - size1 nibbles spare, processing OFF

Checksum: Spare area size (nibbles) = S – (size0 + size1)

7.3.4.12.3.2.2.6 Mode Ah (10)

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- Repeat with buffer 0 to S-1
  - size0 nibbles spare, processing ON
  - 1 nibble pad spare, processing OFF
  - size1 nibbles spare, processing OFF

Checksum: Spare area size (nibbles) = S – (size0 + 1 + size1)

7.3.4.12.3.2.2.7 Mode 2h

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- Repeat with buffer 0 to S-1
  - size0 nibbles spare, processing OFF
  - size1 nibbles spare, processing ON

Checksum: Spare area size (nibbles) = S – (size0 + size1)

7.3.4.12.3.2.2.8 Mode 3h

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- One time with buffer 0
  - size0 nibbles spare, processing ON
- Repeat with buffer 0 to S-1
  - size1 nibbles spare, processing ON

Checksum: Spare area size (nibbles) = size0 + (S – size1)

7.3.4.12.3.2.2.9 Mode 7h

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- One time with buffer 0
  - size0 nibbles spare, processing ON
- Repeat S times (no buffer used)
  - size1 nibbles spare, processing OFF

Checksum: Spare area size (nibbles) = size0 + (S – size1)

7.3.4.12.3.2.2.10 Mode 8h

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- One time with buffer 0
  - size0 nibbles spare, processing ON
- Repeat with buffer 0 to S-1
  - 1 nibble padding spare, processing OFF
  - size1 nibbles spare, processing ON

Checksum: Spare area size (nibbles) = size0 + (S – (1 + size1))

7.3.4.12.3.2.2.11 Mode 4h

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- One time (no buffer used)

  - size0 nibbles spare, processing OFF
- Repeat with buffer 0 to S-1
  - size1 nibbles spare, processing ON

Checksum: Spare area size (nibbles) = size0 + (S – size1)

7.3.4.12.3.2.2.12 Mode 9h

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- One time (no buffer used)
  - size0 nibbles spare, processing OFF
- Repeat with buffer 0 to S-1
  - 1 nibble padding spare, processing OFF
  - size1 nibbles spare, processing ON

Checksum: Spare area size (nibbles) = size0 + (S – (1 + size1))

7.3.4.12.3.2.2.13 Mode 5h

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- Repeat with buffer 0 to S-1
  - size0 nibbles spare, processing ON
- Repeat with buffer 0 to S-1
  - size1 nibbles spare, processing ON

Checksum: Spare area size (nibbles) = S – (size0 + size1)

7.3.4.12.3.2.2.14 Mode Bh (11)

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- Repeat with buffer 0 to S-1
  - size0 nibbles spare, processing ON
- Repeat with buffer 0 to S-1
  - 1 nibble padding spare, processing OFF
  - size1 nibbles spare, processing ON

Checksum: Spare area size (nibbles) = S – (size0 + 1 + size1)

7.3.4.12.3.2.2.15 Mode 6h

Page processing sequence:

- Repeat with buffer 0 to S-1
  - 512-byte data, processing ON
- Repeat with buffer 0 to S-1
  - size0 nibbles spare, processing ON
- Repeat S times (no buffer used)
  - size1 nibbles spare, processing OFF

Checksum: Spare area size (nibbles) = S – (size0 + size1)

7.3.4.12.3.2.3 Supported NAND Page Mappings and ECC Schemes

The following rules apply to the entire mapping description:

- Main data area (sectors) size is hardcoded to 512 bytes.
- Spare area size is programmable.
- All page sections (of main area data bytes, protected spare bytes, unprotected spare bytes, and ECC) are defined as explained in Section 7.3.4.12.3.2.2.1, *Memory Mapping of Data Message*.

Each of the following sections gives a NAND page mapping example (per-sector spare mappings, pooled spare mapping, per-sector spare mapping, with ECC separated at the end of the page).

In the mapping diagrams, sections that belong to the same BCH codeword have the same color (blue or green); unprotected sections are not covered (orange) by the BCH scheme.

Below each mapping diagram, a write (encoding) and read (decoding: syndrome generation) sequence is given, with the number of the active buffers at each point in time (yellow). In the inactive zones (grey), no computing is taking place but the data counter is still active.

In Figure 7-196 through Figure 7-198, the tables on the left summarize the mode, size0, and size1 parameters to program for, respectively, write and read processing of a page, with the given mapping, where:

- P is the size of spare byte section Protected by the ECC (in nibbles)
- U is the size of spare byte section Unprotected by the ECC (in nibbles)
- E is the size of the ECC (in nibbles)
- S is the number of Sectors per page (two in the current diagrams)

Each time the processing of a BCH block is complete (ECC calculation for write/encoding, syndrome generation for read/decoding, indicated by red arrows), the update pointer is pulsed. The processing for block 0 can be the first or the last to complete, depending on the NAND page mapping and operation (read or write). All examples show a page size of 1 KB + spares; that is, S = 2 sectors of 512 bytes. The same principles can be extended to larger pages by adding more sectors.

The actual BCH codeword size is used during the error location work to restrict the search range: by definition, errors can happen only in the codeword that was actually written to the NAND, not in the mathematical codeword of $n = 2^{13} – 1 = 8191$ bits; that codeword (higher-order bits) is all-zero and implicit during computations.

The actual BCH codeword size depends on the mode, the programmed sizes, and the sector number (all sizes in nibbles):

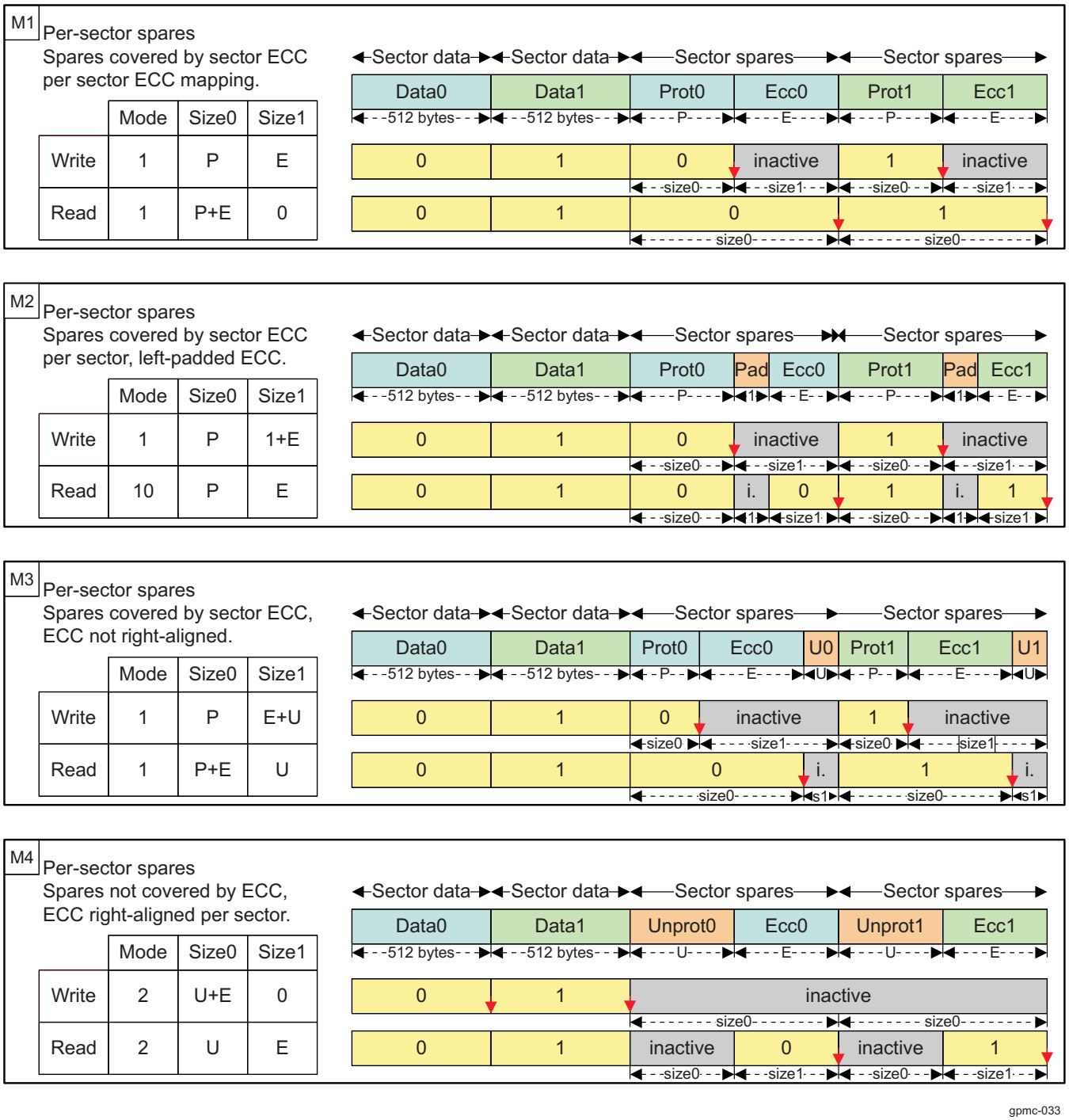
- Spares mapped and protected per sector (below: see M1-M2-M3-M9-M10):
  - All sectors: (512) + P + E
- Spares pooled and protected by sector 0 (below: see M5-M6):
  - Sector 0 codeword: (512) + P + E
  - Other sectors: (512) + E
- Unprotected spares (below: see M4-M7-M8-M11-M12):
  - All codewords (512) + E

7.3.4.12.3.2.3.1 Per-Sector Spare Mappings

In these schemes, each 512-byte sector of the main area has its own dedicated section of the spare area. The spare area of each sector is composed of:

- ECC, which must be located after the data it protects
- Other data, which may or may not be protected by the ECC for its sector.

Figure 7-196. NAND Page Mapping and ECC: Per-Sector Schemes

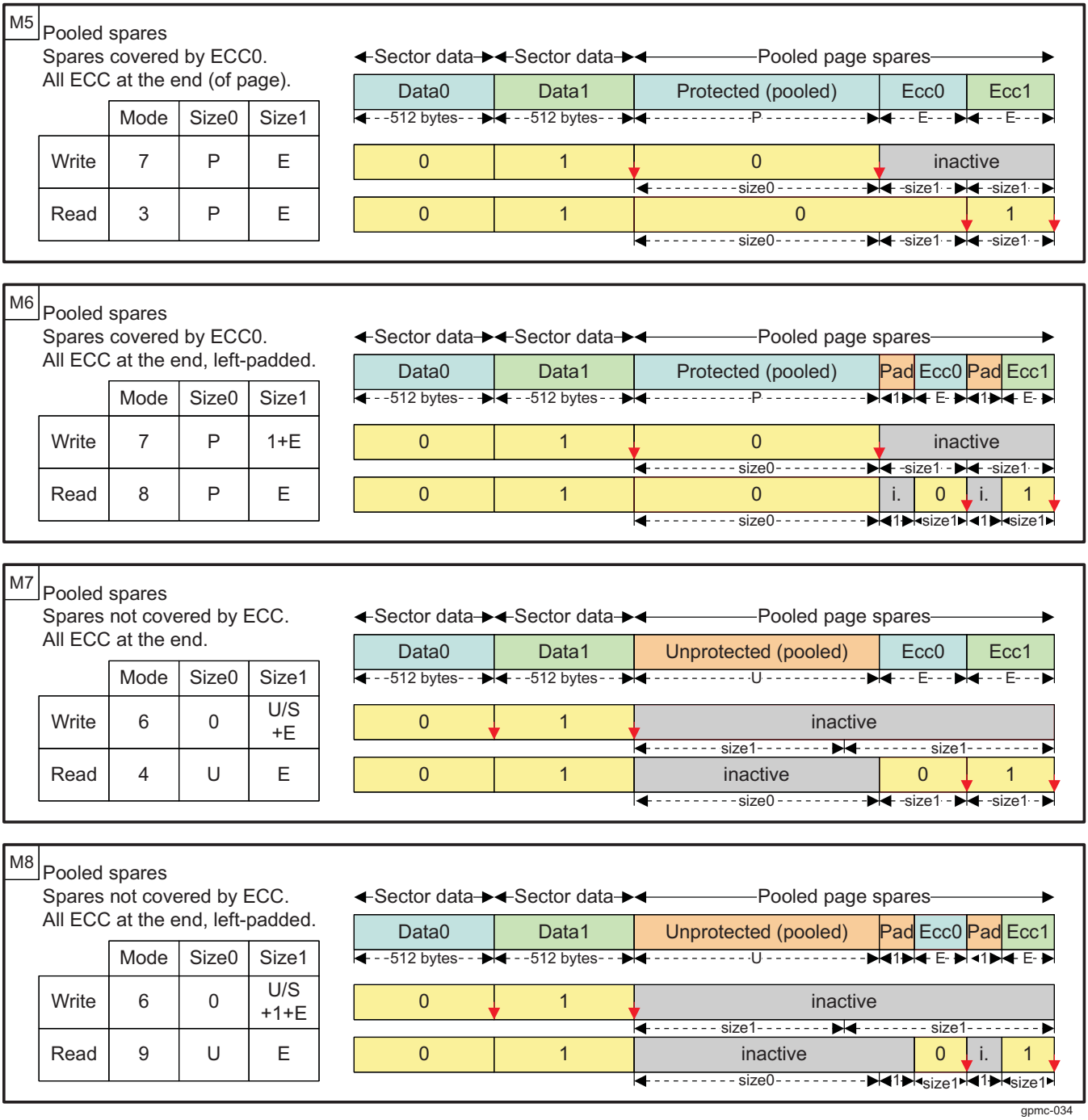


7.3.4.12.3.2.3.2 Pooled Spare Mapping

In the following schemes, the spare area is pooled for the page.

- The ECC of each sector is aligned at the end of the spare area.
- The non-ECC spare data may or may not be covered by the ECC of sector 0.

Figure 7-197. NAND Page Mapping and ECC: Pooled Spare Schemes

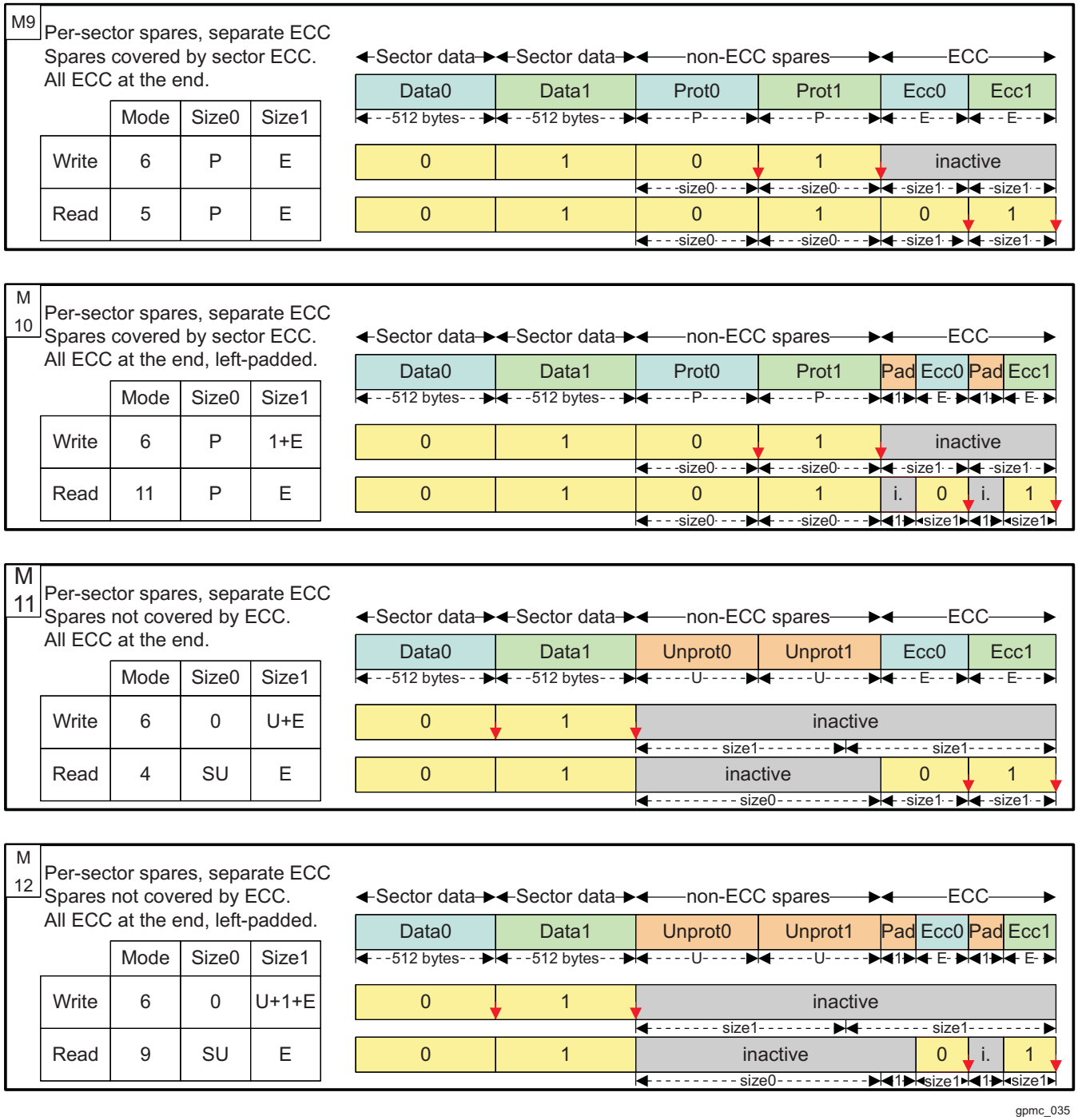


7.3.4.12.3.2.3.3 Per-Sector Spare Mapping, with ECC Separated at the End of the Page

In these schemes, each 512-byte sector of the main area is associated with two sections of the spare area.

- ECC section, all aligned at the end of the page
- Other data section, aligned before the ECCs, each of which may or may not be protected by the ECC for its sector.

Figure 7-198. NAND Page Mapping and ECC: Per-Sector Schemes, With Separate ECC



7.3.4.12.4 Prefetch and Write-Posting Engine

NAND device data access cycles are usually much slower than the CPU system frequency; such NAND read or write accesses issued by the processor affect the overall system performance, especially considering long read or write sequences required for NAND page loading or programming. To minimize this effect on system performance, the GPMC includes a prefetch and write-posting engine, which can be used to read from or write to any chip-select location in a buffered manner.

The prefetch and write-posting engine is a simplified embedded-access requester that presents requests to the access engine on a user-defined chip-select target. The access engine interleaves these requests with any request coming from the interconnect interface; as a default, the prefetch and write-posting engine has the lowest priority.

The prefetch and write-posting engine is dedicated to data-stream access (as opposed to random data access); thus, it is primarily dedicated to NAND support. The engine does not include an address generator; the request is limited to chip-select target identification. It includes a 64-byte FIFO associated with a DMA request synchronization line, for optimal DMA-based use.

The prefetch and write-posting engine uses an embedded 64-byte (32 16-bit word) FIFO to prefetch data from the NAND device in read mode (prefetch mode) or to store host data to be programmed into the NAND device in write mode (write-posting mode). The FIFO draining and filling (read and write) can be controlled by a device host processor through interrupt synchronization (an interrupt is triggered whenever a programmable threshold is reached) or by a device DMA module through DMA request synchronization, with a programmable request byte size in prefetch or posting mode.

The prefetch and write-posting engine includes a single memory pool. Therefore, only one mode, read or write, can be used at any given time. In other words, the prefetch and write-posting engine is a single-context engine that can be allocated to only one chip-select at a time for a read prefetch or a write-posting process.

The engine does not support atomic command and address phase programming and is limited to linear memory read or write access. As a consequence, it is limited to NAND data-stream access. The engine depends on the NAND software driver to control block and page opening with the correct data address pointer initialization, before the engine can read from or write to the NAND memory device.

Once started, engine data read and write sequencing is based solely on FIFO location availability and until the total programmed number of bytes is read or written.

Any host-concurrent accesses to a different chip-select are correctly interleaved with ongoing engine accesses. The engine has the lowest priority access so that host accesses to a different chip-select do not suffer a large latency.

A round-robin arbitration scheme can be enabled to ensure minimum bandwidth to the prefetch and write-posting engine in the case of back-to-back direct memory requests to a different chip-select. If the GPMC_PREFETCH_CONFIG1[23] PFPWENROUNDROBIN bit is enabled, the arbitration grants the prefetch and write posting engine access to the GPMC bus for a number of requests programmed in the GPMC_PREFETCH_CONFIG1[19-16] PFPWWEIGHTEDPRIO bit field.

The prefetch/write-posting engine read or write request is routed to the access engine with the chip-select destination ID. After the required arbitration phase, the access engine processes the request as a single access with the data access size equal to the device size specified in the corresponding chip-select configuration.

NOTE: The destination chip-select configuration must be set to the NAND protocol-compatible configuration for which address lines are not used (the address bus is not changed from its current value). Selecting a different chip-select configuration can produce undefined behavior.

7.3.4.12.4.1 General Facts About the Engine Configuration

The engine can be configured only if the GPMC_PREFETCH_CONTROL[0] STARTENGINE bit is deasserted.

The engine must be correctly configured in prefetch or write-posting mode and must be linked to a NAND chip-select before it can be started. The chip-select is linked using the GPMC_PREFETCH_CONFIG1[26-24] ENGINECSSELECTOR bit field.

In prefetch and write-posting modes, the engine uses byte or 16-bit word access requests, respectively, for an 8- or 16-bit-wide NAND device attached to the linked chip-select. The FIFOTHRESHOLD and TRANSFERCOUNT bit fields must be programmed accordingly as a number of bytes.

When the GPMC_PREFETCH_CONFIG1[7] ENABLEENGINE bit is set, the FIFO entry on the interconnect port side is accessible at any address in the associated chip-select memory region. When the ENABLEENGINE bit is set, any host access to this chip-select is rerouted to the FIFO input. Directly accessing the NAND device linked to this chip-select from the host is still possible through the following registers (where i = 0 to 3):

- GPMC_NAND_COMMAND_i
- GPMC_NAND_ADDRESS_i
- GPMC_NAND_DATA_i

The FIFO entry on the interconnect port can be accessed with byte, 16-bit word, or 32-bit word access size, according to little-endian format, even though the FIFO input is 32 bits wide.

The FIFO control is made easier through the use of interrupts or DMA requests associated with the FIFOTHRESHOLD bit field. The GPMC_PREFETCH_STATUS[30-24] FIFOPOINTER bit field monitors the number of available bytes to be read in prefetch mode or the number of free empty slots that can be written in write-posting mode. The GPMC_PREFETCH_STATUS[13-0] COUNTVALUE bit field monitors the number of remaining bytes to be read or written by the engine according to the value of the TRANSFERCOUNT bit field. The FIFOPOINTER and COUNTVALUE bit fields are always expressed as a number of bytes even if a 16-bit-wide NAND device is attached to the linked chip-select.

In prefetch mode, when the FIFOPOINTER equals 0 (that is, the FIFO is empty), a host read access receives the byte last read from the FIFO as its response. In case of 32-bit word or 16-bit word read accesses, the last byte read from the FIFO is copied the required number of times to fit the requested word size. In write-posting mode, when the FIFOPOINTER equals 0 (that is, the FIFO is full), a host write overwrites the last FIFO byte location. There is no underflow or overflow error reporting in the GPMC.

7.3.4.12.4.2 Prefetch Mode

The prefetch mode is selected when the GPMC_PREFETCH_CONFIG1[0] ACCESSMODE bit is cleared.

The NAND software driver must issue the block and page opening (READ) command with the correct data address pointer initialization before the engine can be started to read from the NAND memory device. The engine is started by asserting the GPMC_PREFETCH_CONTROL[0] STARTENGINE bit. The STARTENGINE bit automatically clears when the prefetch process completes.

If required, the ECC calculator engine must be initialized (that is, reset, configured, and enabled) before the prefetch engine is started so that the ECC is computed correctly on all data read by the prefetch engine.

When the GPMC_PREFETCH_CONFIG1[3] SYNCHROMODE bit is cleared, the prefetch engine starts requesting data as soon as the STARTENGINE bit is set. If using this configuration, the host must monitor the NAND device-ready pin so that it sets the STARTENGINE bit only when the NAND device is in a ready state (that is, data is valid for prefetching).

When the SYNCHROMODE bit is set, the prefetch engine starts requesting data when an active-to-inactive WAIT signal transition is detected. The transition detector must be cleared before any transition detection (see Section 7.3.4.12.2.2, *Ready Pin Monitored by Hardware Interrupt*). The GPMC_PREFETCH_CONFIG1[5-4] WAITPINSELECTOR bit field selects which GPMC_WAIT pin edge detector triggers the prefetch engine in this synchronized mode.

If the STARTENGINE bit is set after the NAND address phase (page opening command), the engine is effectively started only after the actual NAND address phase completion. To prevent GPMC stall during this NAND address phase, set the STARTENGINE bit before NAND address phase completion when in synchronized mode. The prefetch engine starts when an active-to-inactive WAIT signal transition is detected. The STARTENGINE bit is automatically cleared on prefetch process completion.

The prefetch engine issues a read request to fill the FIFO with the amount of data specified by the GPMC_PREFETCH_CONFIG2[13-0] TRANSFERCOUNT bit field.

Table 7-483 describes the prefetch mode configuration.

Table 7-483. Prefetch Mode Configuration

Bit Field	Register	Value	Comments
STARTENGINE	GPMC_PREFETCH_CONTROL[0]	0	Prefetch engine can be configured only if STARTENGINE is set to 0.
ENGINECSSELECTOR	GPMC_PREFETCH_CONFIG1[26-24]	0 to 3	Selects the chip-select associated with a NAND device where the prefetch engine is active.
ACCESSMODE	GPMC_PREFETCH_CONFIG1[0]	0	Selects prefetch mode
FIFOTHRESHOLD	GPMC_PREFETCH_CONFIG1[14-8]		Selects the maximum number of bytes read or written by the host on DMA or interrupt request
TRANSFERCOUNT	GPMC_PREFETCH_CONFIG2[13-0]		Selects the number of bytes to be read or written by the engine to the selected chip-select
SYNCHROMODE	GPMC_PREFETCH_CONFIG1[3]	0/1	Selects when the engine starts the access to the chip-select
WAITPINSELECT	GPMC_PREFETCH_CONFIG1[17-16]	0 to 1	Selects WAIT pin edge detector (if GPMC_PREFETCH_CONFIG1[3] SYNCHROMODE = 1h)
ENABLEOPTIMIZEDACCESS	GPMC_PREFETCH_CONFIG1[27]	0/1	See Section 7.3.4.12.4.6, *Optimizing NAND Access Using the Prefetch and Write-Posting Engine*.
CYCLEOPTIMIZATION	GPMC_PREFETCH_CONFIG1[30-28]		Number of clock cycle removed to timing parameters
ENABLEENGINE	GPMC_PREFETCH_CONFIG1[7]	1	Engine enabled
STARTENGINE	GPMC_PREFETCH_CONTROL[0]	1	Starts the prefetch engine

7.3.4.12.4.3 FIFO Control in Prefetch Mode

The FIFO can be drained directly by a device host processor or a DMA module channel.

In draining mode, the FIFO status can be monitored through the GPMC_PREFETCH_STATUS[30-24] FIFOPOINTER bit field or through the GPMC_PREFETCH_STATUS[16] FIFOTHRESHOLDSTATUS bit. The FIFOPOINTER indicates the current number of available data to be read; FIFOTHRESHOLDSTATUS set to 1 indicates that at least FIFOTHRESHOLD bytes are available from the FIFO.

An interrupt can be triggered by the GPMC if the GPMC_IRQENABLE[0] FIFOEVENTENABLE bit is set. The FIFO interrupt event is logged, and the GPMC_IRQSTATUS[0] FIFOEVENTSTATUS bit is set. To clear the interrupt, all the available bytes must be read, or at least enough bytes to get below the programmed FIFO threshold, and the FIFOEVENTSTATUS bit must be cleared to enable further interrupt events. The FIFOEVENTSTATUS bit must always be reset before asserting the FIFOEVENTENABLE bit to clear any out-of-date logged interrupt event. This interrupt generation must be enabled after enabling the STARTENGINE bit.

Prefetch completion can be monitored through the GPMC_PREFETCH_STATUS[13-0] COUNTVALUE bit field. COUNTVALUE indicates the number of currently remaining data to be requested according to the TRANSFERCOUNT value. An interrupt can be triggered by the GPMC when the prefetch process is complete (that is, COUNTVALUE equals 0) if the GPMC_IRQENABLE[1] TERMINALCOUNTEVENTENABLE bit is set. At prefetch completion, the TERMINALCOUNT interrupt event is also logged, and the GPMC_IRQSTATUS[1] TERMINALCOUNTSTATUS bit is set. To clear the interrupt, the TERMINALCOUNTSTATUS bit must be cleared. The TERMINALCOUNTSTATUS bit must always be cleared prior to asserting the TERMINALCOUNTEVENTENABLE bit to clear any out-of-date logged interrupt event.

NOTE: The COUNTVALUE value is valid only when the prefetch engine is active (started), and an interrupt is only triggered when COUNTVALUE reaches 0, that is, when the prefetch engine automatically goes from an active to an inactive state.

The number of bytes to be prefetched (programmed in TRANSFERCOUNT) must be a multiple of the programmed FIFOTHRESHOLD to trigger the correct number of interrupts allowing a deterministic and transparent FIFO control. If this guideline is respected, the number of ISR accesses is always required and the FIFO is always empty after the last interrupt is triggered. In other cases, the TERMINALCOUNT interrupt must be used to read the remaining bytes in the FIFO (the number of remaining bytes being lower than the FIFOTHRESHOLD value).

In DMA draining mode, the GPMC_PREFETCH_CONFIG1[2] DMAMODE bit must be set so that the GPMC issues a DMA hardware request when at least FIFOTHRESHOLD bytes are ready to be read from the FIFO. The DMA channel that owns this DMA request must be programmed so that the number of bytes programmed in FIFOTHRESHOLD is read from the FIFO during the DMA request process. The DMA request is kept active until this number of bytes has effectively been read from the FIFO, and no other DMA request can be issued until the ongoing active request is complete.

In prefetch mode, the TERMINALCOUNT event is also a source of DMA requests if the number of bytes to be prefetched is not a multiple of FIFOTHRESHOLD, the remaining bytes in the FIFO can be read by the DMA channel using the last DMA request. This assumes that the number of remaining bytes to be read is known and controlled through the DMA channel programming model.

Any potentially active DMA request is cleared when the prefetch engine goes from inactive-to-active prefetch (the STARTENGINE bit is set to 1). The associated DMA channel must always be enabled after setting the STARTENGINE bit so that the out-of-date active DMA request does not trigger spurious DMA transfers.

7.3.4.12.4.4 Write-Posting Mode

The write-posting mode is selected when the GPMC_PREFETCH_CONFIG1[0] ACCESSMODE bit is set.

The NAND software driver must issue the correct address pointer initialization command (page program) before the engine can start writing data into the NAND memory device. The engine starts when the GPMC_PREFETCH_CONTROL[0] STARTENGINE bit is set to 1. The STARTENGINE bit clears automatically when posting completes. When all data have been written to the NAND memory device, the NAND software driver must issue the second cycle program command and monitor the status for programming process completion (adding ECC handling, if required).

If used, the ECC calculator engine must be started (configured, reset, and enabled) before the posting engine is started so that the ECC parities are calculated properly on all data written by the prefetch engine to the associated chip-select.

In write-posting mode, the GPMC_PREFETCH_CONFIG1[3] SYNCHROMODE bit must be cleared so that posting starts as soon as the STARTENGINE bit is set and the FIFO is not empty.

If the STARTENGINE bit is set after the NAND address phase (page program command), the STARTENGINE setting is effective only after the actual NAND command completion. To prevent GPMC stall during this NAND command phase, set the STARTENGINE bit field before the NAND address completion and ensure that the associated DMA channel is enabled after the NAND address phase.

The posting engine issues a write request when valid data are available from the FIFO and until the programmed GPMC_PREFETCH_CONFIG2[13-0] TRANSFERCOUNT accesses are complete.

The STARTENGINE bit clears automatically when posting completes. When all data have been written to the NAND memory device, the NAND software driver must issue the second cycle program command and monitor the status for programming process completion. The closing program command phase must be issued only when the full NAND page has been written into the NAND flash write buffer, including the spare area data and the ECC parities, if used.

Table 7-484 describes the write-posting configuration.

Table 7-484. Write-Posting Mode Configuration

Bit Field	Register	Value	Comments
STARTENGINE	GPMC_PREFETCH_CONTROL[0]	0	Write-posting engine can be configured only if STARTENGINE is set to 0.

Table 7-484. Write-Posting Mode Configuration (continued)

Bit Field	Register	Value	Comments
ENGINECSSELECTOR	GPMC_PREFETCH_CONFIG1[26-24]	0 to 3	Selects the chip-select associated with a NAND device where the prefetch engine is active
ACCESSMODE	GPMC_PREFETCH_CONFIG1[0]	1	Selects write-posting mode
FIFOTHRESHOLD	GPMC_PREFETCH_CONFIG1[14-8]		Selects the maximum number of bytes read or written by the host on DMA or interrupt request
TRANSFERCOUNT	GPMC_PREFETCH_CONFIG2[13-0]		Selects the number of bytes to be read or written by the engine from/to the selected chip-select
SYNCHROMODE	GPMC_PREFETCH_CONFIG1[3]	0	Engine starts the access to chip-select as soon as STARTENGINE is set.
ENABLEOPTIMIZEDACCESS	GPMC_PREFETCH_CONFIG1[27]	0/1	See Section 7.3.4.12.4.6, *Optimizing NAND Access Using the Prefetch and Write-Posting Engine*.
CYCLEOPTIMIZATION	GPMC_PREFETCH_CONFIG1[30-28]		
ENABLEENGINE	GPMC_PREFETCH_CONFIG1[7]	1	Engine enabled
STARTENGINE	GPMC_PREFETCH_CONTROL[0]	1	Starts the prefetch engine

7.3.4.12.4.5 FIFO Control in Write-Posting Mode

The FIFO can be filled directly by a device host processor or a DMA module channel.

In filling mode, the FIFO status can be monitored through the FIFOPOINTER or through the GPMC_PREFETCH_STATUS[16] FIFOTHRESHOLDSTATUS bit. FIFOPOINTER indicates the current number of available free byte places in the FIFO, and the FIFOTHRESHOLDSTATUS bit, when set, indicates that at least FIFOTHRESHOLD free byte places are available in the FIFO.

An interrupt can be issued by the GPMC if the GPMC_IRQENABLE[0] FIFOEVENTENABLE bit is set. When the interrupt is fired, the GPMC_IRQSTATUS[0] FIFOEVENTSTATUS bit is set. To clear the interrupt, enough bytes must be written to fill the FIFO or to get below the programmed threshold, and the FIFOEVENTSTATUS bit must be cleared to get further interrupt events. The FIFOEVENTSTATUS bit must always be cleared before asserting the FIFOEVENTENABLE bit to clear any out-of-date logged interrupt event. This interrupt must be enabled after enabling the STARTENGINE bit.

The posting completion can be monitored through the GPMC_PREFETCH_STATUS[13-0] COUNTVALUE bit field. COUNTVALUE indicates the current number of remaining data to be written based on the value of the TRANSFERCOUNT bit field. An interrupt is issued by the GPMC when the write-posting process completes (that is, COUNTVALUE equal to 0) if the GPMC_IRQENABLE[1] TERMINALCOUNTEVENTENABLE bit is set. When the interrupt is fired, the GPMC_IRQSTATUS[1] TERMINALCOUNTSTATUS bit is set. To clear the interrupt, the TERMINALCOUNTSTATUS bit must be cleared. The TERMINALCOUNTSTATUS bit must always be cleared before asserting the TERMINALCOUNTEVENTENABLE bit to clear any out-of-date logged interrupt event.

NOTE: The value of the COUNTVALUE bit field is valid only if the write-posting engine is active and started, and an interrupt is issued only when COUNTVALUE reaches 0; that is, when the posting engine automatically goes from active to inactive.

In DMA filling mode, the DMAMode bit field in the GPMC_PREFETCH_CONFIG1[2] DMAMODE bit must be set so that the GPMC issues a DMA hardware request when at least FIFOTHRESHOLD bytes-free places are available in the FIFO. The DMA channel that owns this DMA request must be programmed so that a number of bytes equal to the value programmed in the FIFOTHRESHOLD bit field are written into the FIFO during the DMA access. The DMA request remains active until the associated number of bytes has effectively been written into the FIFO, and no other DMA request can be issued until the ongoing active request completes.

Any potentially active DMA request is cleared when the prefetch engine goes from inactive-to-active prefetch (STARTENGINE set to 1). The associated DMA channel must always be enabled after setting the STARTENGINE bit so that an out-of-date active DMA request does not trigger spurious DMA transfers.

In write-posting mode, DMA or CPU fills the FIFO with no consideration for the associated byte enables. Any byte stored in the FIFO is written into the memory device.

7.3.4.12.4.6 Optimizing NAND Access Using the Prefetch and Write-Posting Engine

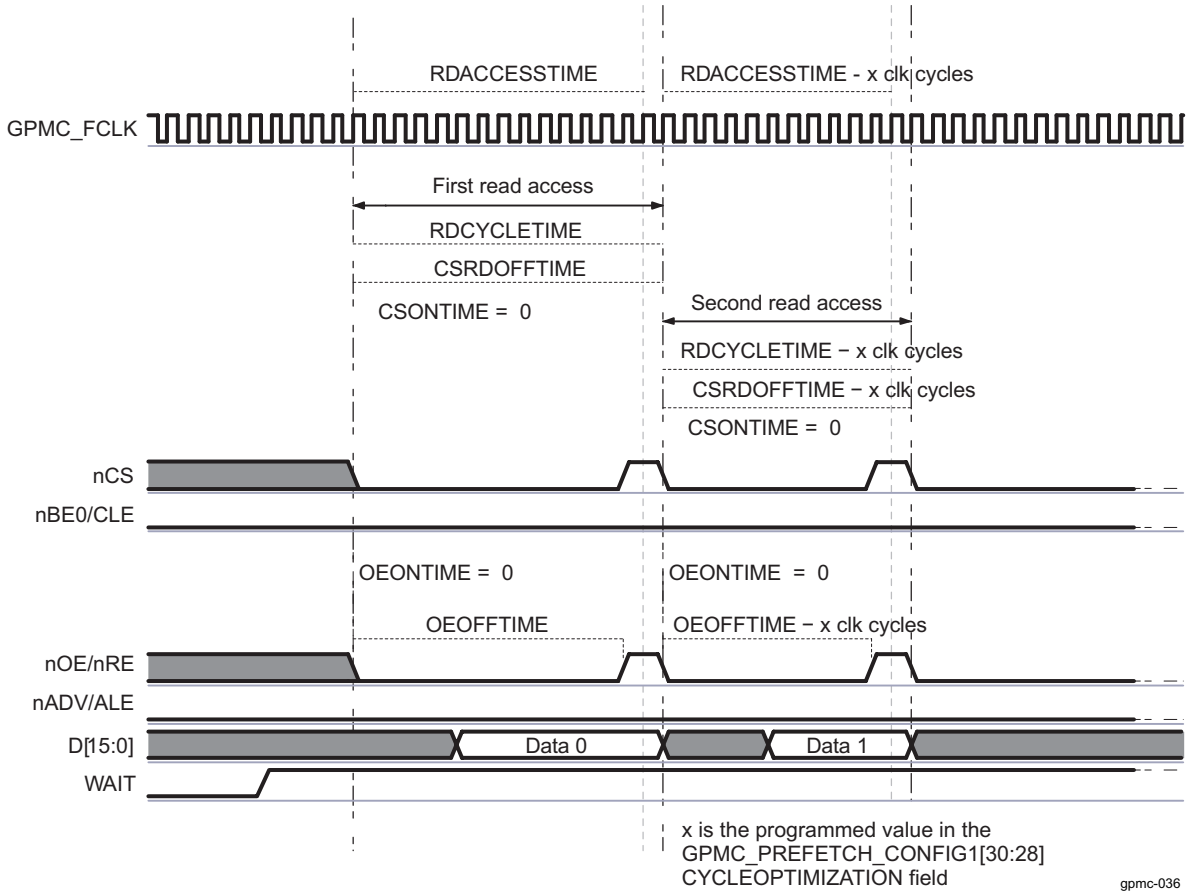
Access time to a NAND memory device can be optimized for back-to-back accesses if the associated nCS signal is not deasserted between accesses. The GPMC access engine can track prefetch engine accesses to optimize the access timing parameter programmed for the allocated chip-select, if no accesses to other chip-selects (that is, interleaved accesses) occur. Similarly, the access engine also eliminates CYCLE2CYCLEDELAY even if CYCLE2CYCLESAMECSEN is set. This capability is limited to the prefetch and write-posting engine accesses, and accesses to a NAND memory device (through the defined chip-select memory region or through the GPMC_NAND_DATA_i location, where i = 0 to 3) are never optimized.

The GPMC_PREFETCH_CONFIG1[27] ENABLEOPTIMIZEDACCESS bit must be set to enable optimized accesses. To optimize access time, the GPMC_PREFETCH_CONFIG1[30-28] CYCLEOPTIMIZATION bit field defines the number of GPMC_FCLK cycles to be suppressed from the following timing parameters:

- RDCYCLETIME
- WRCYCLETIME
- RDACCESSTIME
- WRACCESSTIME
- CSOFFTIME
- ADVOFFTIME
- OEOFFTIME
- WEOFFTIME

Figure 7-199 shows that in the case of back-to-back accesses to the NAND flash through the prefetch engine, CYCLE2CYCLESAMECSEN is forced to 0 when using optimized accesses. The first access uses the regular timing settings for this chip-select. All accesses after this one use settings reduced by x clock cycles, x being defined by the GPMC_PREFETCH_CONFIG1[30-28] CYCLEOPTIMIZATION bit field.

Figure 7-199. NAND Read Cycle Optimization Timing Description



7.3.4.12.4.7 Interleaved Accesses Between Prefetch and Write-Posting Engine and Other Chip-Selects

Any on-going read or write access from the prefetch and write-posting engine is completed before an access to any other chip-select can be initiated. As a default, the arbiter uses a fixed-priority algorithm, and the prefetch and write-posting engine has the lowest priority. The maximum latency added to access starting time in this case equals the RDCYCLETIME or WRCYCLETIME (optimized or not) plus the requested BUSTURNAROUND delay for bus turnaround completion programmed for the chip-select to which the NAND device is connected.

Alternatively, a round-robin arbitration can be used to prioritize accesses to the external bus. This arbitration scheme is enabled by setting the GPMC_PREFETCH_CONFIG1[23] PFPWENROUNDROBIN bit. When a request to another chip-select is received while the prefetch and write-posting engine is active, priority is given to the new request. The request processed thereafter is the prefetch and write-posting engine request, even if another interconnect request is passed in the mean time. The engine keeps control of the bus for an additional number of requests programmed in the GPMC_PREFETCH_CONFIG1[19-16] PFPWWEIGHTEDPRIO bit field. Control is then passed to the direct interconnect request.

As an example, the round-robin arbitration scheme is selected with PFPWWEIGHTEDPRIO set to 2h. Considering that the prefetch and write-posting engine and the interconnect interface are always requesting access to the external interface, the GPMC grants priority to the direct interconnect access for one request. The GPMC then grants priority to the engine for three requests, and finally back to the direct interconnect access, until the arbiter is reset when one of the two initiators stops initiating requests.

7.3.5 GPMC Basic Programming Model

7.3.5.1 GPMC High-Level Programming Model Overview

The goal of the basic high-level programming model is to introduce a top-down approach to users that need to configure the GPMC module.

Figure 7-200 and through Table 7-486 show a programming model top-level diagram for the GPMC, and a description of each step. Each block of the diagram is described in one of the following sections through a set of registers to configure.

Figure 7-200. Programming Model Top-Level Diagram

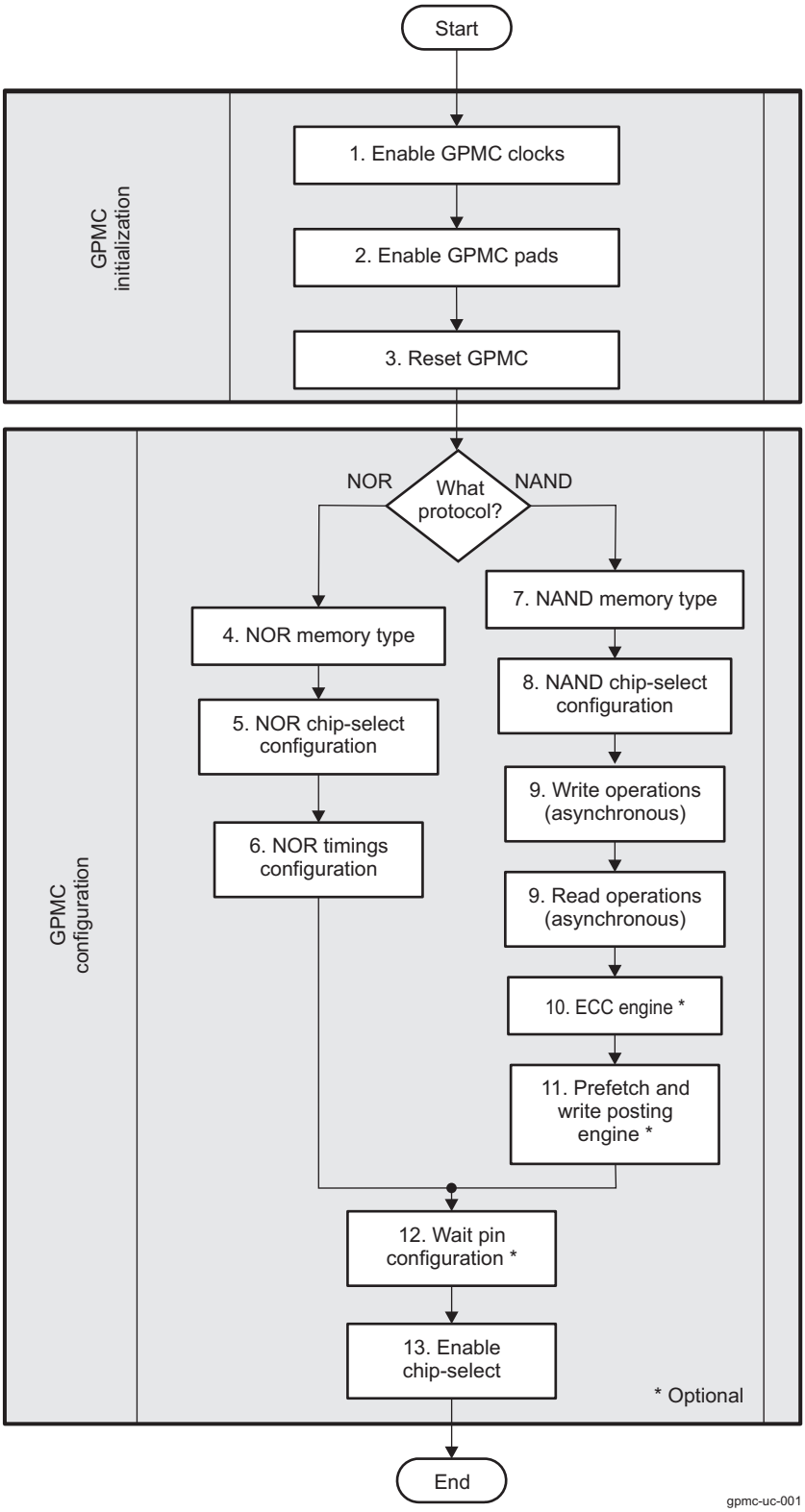


Table 7-485. GPMC Configuration in NOR Mode

Step	Description
NOR Memory Type	See Table 7-488.
NOR Chip-Select Configuration	See Table 7-489.
NOR Timings Configuration	See Table 7-490.
WAIT Pin Configuration	See Table 7-498.
Enable Chip-Select	See Table 7-499.

Table 7-486. GPMC Configuration in NAND Mode

Step	Description
NAND Memory Type	See Table 7-493.
NAND Chip-Select Configuration	See Table 7-494.
Write Operations (Asynchronous)	See Table 7-495.
Read Operations (Asynchronous)	See Table 7-495.
ECC Engine	See Table 7-496.
Prefetch and Write-Posting Engine	See Table 7-497.
WAIT Pin Configuration	See Table 7-498.
Enable Chip-Select	See Table 7-499.

7.3.5.2 GPMC Initialization

Table 7-515 describes the settings required to prepare the GPMC; that is enabling its clock and pads, and proceeding to a GPMC reset.

Table 7-487. Reset GPMC

Subprocess Name	Register/Bit Field	Value
Start a software reset.	GPMC_SYSCONFIG[1] SOFTRESET	1h
Wait until	GPMC_SYSSTATUS[0] RESETDONE =	1h

7.3.5.3 GPMC Configuration in NOR Mode

This section gives a generic configuration for parameters related to the NOR memory connected to the GPMC.

Table 7-488. NOR Memory Type

Subprocess Name	Register / Bit Field	Value
Set the NOR protocol.	GPMC_CONFIG1_i[11-10] DEVICETYPE	0h
Set a device size.	GPMC_CONFIG1_i[13-12] DEVICESIZE	x
Select an address and data multiplexing protocol.	GPMC_CONFIG1_i[9] MUXADDDATA	x
Set the attached device page length.	GPMC_CONFIG1_i[24-23] ATTACHEDDEVICEPAGELENGTH	x
Set the wrapping burst capabilities.	GPMC_CONFIG1_i[31] WRAPBURST	x
Select a timing signals latencies factor.	GPMC_CONFIG1_i[4] TIMEPARAGRANULARITY	x
Select an output clock frequency (1).	GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER	x
Choose an output clock activation time (1).	GPMC_CONFIG1_i[26-25] CLKACTIVATIONTIME	x
Set a single or multiple access for read operations (1).	GPMC_CONFIG1_i[30] READMULTIPLE	x
Set a synchronous or asynchronous mode for read operations.	GPMC_CONFIG1_i[29] READTYPE	x
Set a single or multiple access for write operations.	GPMC_CONFIG1_i[28] WRITEMULTIPLE	x

(1) Applies only to synchronous configurations (or non-multiplexed asynchronous for multiple access one)

Table 7-488. NOR Memory Type (continued)

Subprocess Name	Register / Bit Field	Value
Set a synchronous or asynchronous mode for write operations.	GPMC_CONFIG1_i[27] WRITETYPE	x

Table 7-489. NOR Chip-Select Configuration

Subprocess Name	Register/Bit Field	Value
Select the chip-select base address.	GPMC_CONFIG7_i[5-0] BASEADDRESS	x
Select the chip-select mask address.	GPMC_CONFIG7_i[11-8] MASKADDRESS	x

Table 7-490. NOR Timings Configuration

Subprocess Name	Register/Bit Field	Value
Configure adequate timing parameters in various memory modes.	See Section 7.3.5.6, *GPMC Timing Parameters*	

Table 7-491. WAIT Pin Configuration

Subprocess Name	Register/Bit Field	Value
Enable or disable WAIT pin monitoring for read operations.	GPMC_CONFIG1_i[22] WAITREADMONITORING	x
Enable or disable WAIT pin monitoring for write operations.	GPMC_CONFIG1_i[21] WAITWRITEMONITORING	x
Select a WAIT pin monitoring time.	GPMC_CONFIG1_i[19-18] WAITMONITORINGTIME	x
Choose the input WAIT pin for the chip-select.	GPMC_CONFIG1_i[17-16] WAITPINSELECT	x

Table 7-492. Enable Chip-Select

Subprocess Name	Register/Bit Field	Value
When all parameters are configured, enable the chip-select.	GPMC_CONFIG7_i[6] CSVALID	x

7.3.5.4 GPMC Configuration in NAND Mode

This section gives a generic configuration for parameters related to the NAND memory connected to the GPMC.

Table 7-493. NAND Memory Type

Subprocess Name	Register/Bit Field	Value
Set the NAND protocol.	GPMC_CONFIG1_i[11-10] DEVICETYPE	2h
Set a device size.	GPMC_CONFIG1_i[13-12] DEVICESIZE	x
Set the address and data multiplexing protocol to non-multiplexed attached device.	GPMC_CONFIG1_i[9] MUXADDDATA	0h
Select a timing signals latencies factor.	GPMC_CONFIG1_i[4] TIMEPARAGRANULARITY	x
Set a synchronous or asynchronous mode and a single or multiple access for read and write operations.	See Section 7.3.5.5, *Set Memory Access*.	x

Table 7-494. NAND Chip-Select Configuration

Subprocess Name	Register/Bit Field	Value
Select the chip-select base address.	GPMC_CONFIG7_i[5-0] BASEADDRESS	x
Select the chip-select minimum granularity (16MB).	GPMC_CONFIG7_i[11-8] MASKADDRESS	x

Table 7-495. Asynchronous Read and Write Operations

Subprocess Name	Register/Bit Field	Value
Configure adequate timing parameters in asynchronous modes	See Section 7.3.5.6, *GPMC Timing Parameters*.	

Table 7-496. ECC Engine

Subprocess Name	Register/Bit Field	Value
Select the ECC result register where the first ECC computation is stored (applies only to Hamming).	GPMC_ECC_CONTROL[3-0] ECCPOINTER	x (1)
Clear all ECC result registers.	GPMC_ECC_CONTROL[8] ECCCLEAR	Write 1 to clear.
Define ECCSIZE0 and ECCSIZE1.	GPMC_ECC_SIZE_CONFIG[19-12] ECCSIZE0 and [29-22] ECCSIZE1	x (2)
Select the size of each of the 9 result registers (size specified by ECCSIZE0 or ECCSIZE1).	GPMC_ECC_SIZE_CONFIG[j-1] ECCjRESULTSIZE where j = 1 to 9	x
Select the chip-select where ECC is computed.	GPMC_ECC_CONFIG[3-1] ECCCS	x
Select the Hamming code or BCH code ECC algorithm in use.	GPMC_ECC_CONFIG[16] ECCALGORITHM	x
Select word size for ECC calculation.	GPMC_ECC_CONFIG[7] ECC16B	x
If the BCH code is used, Set an error correction capability and Select a number of sectors to process.	GPMC_ECC_CONFIG[13-12] ECCBCHTSEL and GPMC_ECC_CONFIG[6-4] ECCTOPSECTOR	x
Enable the ECC computation.	GPMC_ECC_CONFIG[0] ECCENABLE	1h

(1) This parameter depends on the numbers of sectors in a page.
(2) Depends on the size of each sector in the NAND page

Table 7-497. Prefetch and Write-Posting Engine

Subprocess Name	Register/Bit Field	Value
Disable the engine before configuration.	GPMC_PREFETCH_CONTROL[0] STARTENGINE	0h
Select the chip-select associated with a NAND device where the prefetch engine is active.	GPMC_PREFETCH_CONFIG1[26-24] ENGINECSSELECTOR	x
Select access direction through prefetch engine, read or write.	GPMC_PREFETCH_CONFIG1[0] ACCESSMODE	x
Select the threshold used to issue a DMA request.	GPMC_PREFETCH_CONFIG1[14-8] FIFOTHRESHOLD	x
Select DMA synchronized mode or software manual mode.	GPMC_PREFETCH_CONFIG1[2] DMAMODE	x
Select if the engine immediately starts accessing the memory upon STARTENGINE assertion or if hardware synchronization based on a WAIT signal is used.	GPMC_PREFETCH_CONFIG1[3] SYNCHROMODE	x
Select which WAIT pin edge detector should start the engine in synchronized mode.	GPMC_PREFETCH_CONFIG1[5-4] WAITPINSELECTOR	x
Enter a number of clock cycles removed to timing parameters (for all back-to-back accesses to the NAND flash except the first one).	GPMC_PREFETCH_CONFIG1[30-28] CYCLEOPTIMIZATION	x
Enable the prefetch postwrite engine.	GPMC_PREFETCH_CONFIG1[7] ENABLEENGINE	1h
Select the number of bytes to be read or written by the engine to the selected chip-select.	GPMC_PREFETCH_CONFIG2[13-0] TRANSFERCOUNT	x
Start the prefetch engine.	GPMC_PREFETCH_CONTROL[0] STARTENGINE	1h

Table 7-498. WAIT Pin Configuration

Subprocess Name	Register/Bit Field	Value
Selects when the engine starts the access to chip-select.	GPMC_PREFETCH_CONFIG1[3] SYNCHROMODE	x
Select which WAIT pin edge detector should start the engine in synchronized mode.	GPMC_PREFETCH_CONFIG1[5-4] WAITPINSELECTOR	x

Table 7-499. Enable Chip-Select

Subprocess Name	Register/Bit Field	Value
When all parameters are configured, enable the chip-select.	GPMC_CONFIG7_i[6] CSVALID	x

7.3.5.5 Set Memory Access

This section describes the bit field to configure to set the GPMC in various memory modes. Table 7-500 and Table 7-501 provide check lists for mode parameters and access type parameters, respectively.

Table 7-500. Mode Parameters Check List

Register	Bit	Name	Asynchronous				Synchronous			
			Single Read Access	Single Write Access	Multiple Read (Page) Access	Multiple Write (Page) Access	Single Read Access	Single Write Access	Multiple Read (Burst) Access	Multiple Write (Burst) Access
GPMC_CONFIG1_i	30	READMULTIPLE	0h	–	1h (1)	N/S	0h	–	1h	–
GPMC_CONFIG1_i	29	READTYPE	0h	–	0h (1)	N/S	1h	–	1h	–
GPMC_CONFIG1_i	28	WRITEMULTIPLE	–	0h	- (1)	N/S	–	0h	–	1h
GPMC_CONFIG1_i	27	WRITETYPE	–	0h	- (1)	N/S	–	1h	–	1h

(1) Multiple read is not supported in address/data-multiplexed and AAD-multiplexed modes. Multiple read is supported in non-multiplexed mode.

Table 7-501. Access Type Parameters Check List

Register	Bit	Name	Access Type		
			non-multiplexed	Address/ Data-Multiplexed	AAD-Multiplexed
GPMC_CONFIG1_i	9-8	MUXADDDATA	0h	2h	1h

7.3.5.6 GPMC Timing Parameters

Figure 7-201 shows a programming model diagram for the NOR interfacing timing parameters.

Figure 7-201. NOR Interfacing Timing Parameters Diagram

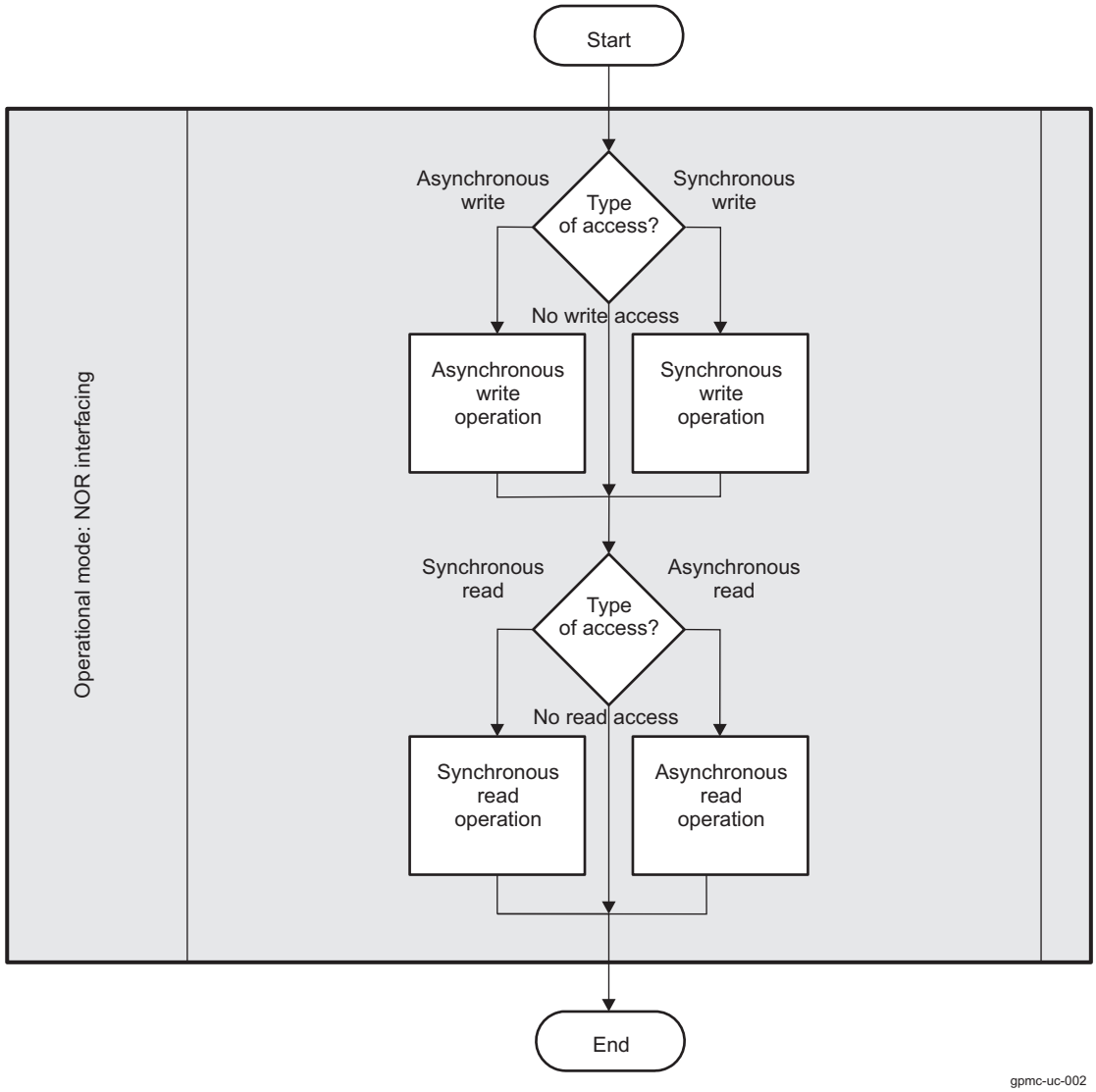


Table 7-502 lists the bit fields to configure adequate timing parameters in various memory modes.

Table 7-502. Timing Parameters

			Asynchronous			Synchronous				Access Type		
Register	Bit	Name	Single Read Access	Single Write Access	Multiple Read (Page) access	Single Read Access	Single Write Access	Multiple Read (Burst) Access	Multiple Write (Burst) Access	Non-multiplexed	Address / Data-Multiplexed	AAD Multiplexed
GPMC_CONFIG1_i	9	MUXADDDATA	y	y	y	y	y	y	y	y	y	y
GPMC_CONFIG1_i	29	READTYPE	y		y	y		y		y	y	y
GPMC_CONFIG1_i	30	READMULTIPLE	y		y	y		y		y	y	y
GPMC_CONFIG1_i	27	WRITETYPE		y			y		y	y	y	y
GPMC_CONFIG1_i	28	WRITEMULTIPLE		y			y		y	y	y	y
GPMC_CONFIG1_i	31	WRAPBURST						y	y	y	y	y
GPMC_CONFIG1_i	26-25	CLKACTIVATIONTIME				y	y	y	y	y	y	y
GPMC_CONFIG1_i	19-18	WAITMONITORINGTIME	y	y	y	y	y	y	y	y	y	y
GPMC_CONFIG1_i	4	TIMEPARAGRANULARITY	y	y	y	y	y	y	y	y	y	y
GPMC_CONFIG2_i	20-16	CSWROFFTIME		y			y		y	y	y	y
GPMC_CONFIG2_i	12-8	CSRDOFFTIME	y		y	y		y		y	y	y
GPMC_CONFIG2_i	7	CSEXTRADELAY	y	y	y	y	y	y	y	y	y	y
GPMC_CONFIG2_i	3-0	CSONTIME	y	y	y	y	y	y	y	y	y	y
GPMC_CONFIG3_i	30-28	ADVAADMUXWROFFTIME		y			y		y			y
GPMC_CONFIG3_i	30-29	ADVAADMUXRDOFFTIME	y		y	y		y				y
GPMC_CONFIG3_i	6-4	ADVAADMUXONTIME	y	y	y	y	y	y	y			y
GPMC_CONFIG3_i	20-16	ADVWROFFTIME		y			y		y	y	y	y
GPMC_CONFIG3_i	12-8	ADVRDOFFTIME	y		y	y		y		y	y	y
GPMC_CONFIG3_i	7	ADVEXTRADELAY	y	y	y	y	y	y	y	y	y	y
GPMC_CONFIG3_i	3-0	ADVONTIME	y	y	y	y	y	y	y	y	y	y
GPMC_CONFIG4_i	15-13	OEAADMUXOFFTIME	y	y	y	y	y	y	y			y
GPMC_CONFIG4_i	6-4	OEAADMUXONTIME	y	y	y	y	y	y	y			y
GPMC_CONFIG4_i	28-24	WEOFFTIME		y			y		y	y	y	y
GPMC_CONFIG4_i	23	WEEXTRADELAY		y			y		y	y	y	y
GPMC_CONFIG4_i	19-16	WEONTIME		y			y		y	y	y	y
GPMC_CONFIG4_i	12-8	OEOFFTIME	y		y	y		y		y	y	y
GPMC_CONFIG4_i	7	OEEXTRADELAY	y		y	y		y		y	y	y
GPMC_CONFIG4_i	3-0	OEONTIME	y		y	y		y		y	y	y
GPMC_CONFIG5_i	27-24	PAGEBURSTACCESSTIME			y			y	y	y	y	y
GPMC_CONFIG5_i	20-16	RDACCESSTIME	y		y	y		y		y	y	y

Table 7-502. Timing Parameters (continued)

			Asynchronous			Synchronous				Access Type		
GPMC_CONFIG5_i	12-8	WRCYCLETIME		y			y		y	y	y	y
GPMC_CONFIG5_i	4-0	RDCYCLETIME	y		y	y		y		y	y	y
GPMC_CONFIG6_i	28-24	WRACCESSTIME		y			y		y	y	y	y
GPMC_CONFIG6_i	19-16	WRDATAONADMUXBUS		y			y		y		y	y
GPMC_CONFIG6_i	11-8	CYCLE2CYCLEDELAY	y	y	y	y	y	y	y	y	y	y
GPMC_CONFIG6_i	7	CYCLE2CYCLESAMECSEN	y	y	y	y	y	y	y	y	y	y
GPMC_CONFIG6_i	6	CYCLE2CYCLEDIFFCSEN	y	y	y	y	y	y	y	y	y	y
GPMC_CONFIG6_i	3-0	BUSTURNAROUND	y	y	y	y	y	y	y	y	y	y
GPMC_CONFIG7_i	6	CSVALID	y	y	y	y	y	y	y	y	y	y

7.3.5.6.1 GPMC Timing Parameters Formulas

This section is intended to help the user calculate the GPMC timing bit field values. Formulas are not listed exhaustively.

The section describes:

- NAND flash interface timing parameters formulas
- Synchronous NOR flash timing parameters formulas
- Asynchronous NOR flash timing parameters formulas

For complete information, such as OPP and board effects on timings, see the device Data Manual.

7.3.5.6.1.1 NAND Flash Interface Timing Parameters Formulas

This section lists formulas to calculate NAND timing parameters. This is the case when GPMC_CONFIG1_i[11-10] DEVICETYPE = 2h. Table 7-503 describes the NAND timing parameters.

Table 7-503. NAND Formulas Description

Configuration Parameter	Unit	Description
A	ns	Pulse duration – GPMC_WEn valid time
B	ns	Delay time – GPMC_CS valid to GPMC_WEn valid
C	ns	Delay time – GPMC_BEn0_CLE/GPMC_ADVn_ALE high to GPMC_WEn valid
D	ns	Delay time – GPMC_AD[15:0] valid to GPMC_WEn valid
E	ns	Delay time – GPMC_WEn invalid to GPMC_AD[15:0] invalid
F	ns	Delay time – GPMC_WEn invalid to GPMC_BEn0_CLE/GPMC_ADVn_ALE invalid
G	ns	Delay time – GPMC_WEn invalid to GPMC_CS invalid
H	ns	Cycle time – Write cycle time
I	ns	Delay time – GPMC_CS valid to GPMC_OEn_REn valid
J	ns	Setup time – GPMC_AD[15:0] valid to GPMC_OEn_REn invalid
K	ns	Pulse duration – GPMC_OEn_REn valid time
L	ns	Cycle time – Read cycle time
M	ns	Delay time – GPMC_OEn_REn invalid to GPMC_CS invalid

The configuration parameters are calculated through the following formulas. For more information, see the device Data Manual.

A = (WEOffTime – WEOnTime) * (TimeParaGranularity + 1) * GPMC_FCLK period

B = ((WEOnTime – CSOnTime) * (TimeParaGranularity + 1) + 0.5 * (WEExtraDelay – CSExtraDelay)) * GPMC_FCLK period

C = ((WEOnTime – ADVOnTime) * (TimeParaGranularity + 1) + 0.5 * (WEExtraDelay - ADVExtraDelay)) * GPMC_FCLK period

D = (WEOnTime * (TimeParaGranularity + 1) + 0.5 * WEExtraDelay) * GPMC_FCLK period

E = (WrCycleTime – WEOffTime * (TimeParaGranularity + 1) – 0.5 * WEExtraDelay) * GPMC_FCLK period

F = (ADVWrOffTime – WEOffTime * (TimeParaGranularity + 1) + 0.5 * (ADVExtraDelay – WEExtraDelay) * GPMC_FCLK period

G = (CSWrOffTime – WEOffTime * (TimeParaGranularity + 1) + 0.5 * (CSExtraDelay – WEExtraDelay) * GPMC_FCLK period

H = WrCycleTime * (1 + TimeParaGranularity) * GPMC_FCLK period

I = ((OEOnTime – CSOnTime) * (TimeParaGranularity + 1) + 0.5 * (OEExtraDelay – CSExtraDelay)) * GPMC_FCLK period

J = ((RdAccessTime – OEOffTime) * (TimeParaGranularity + 1) – 0.5 * OEExtraDelay)) * GPMC_FCLK period

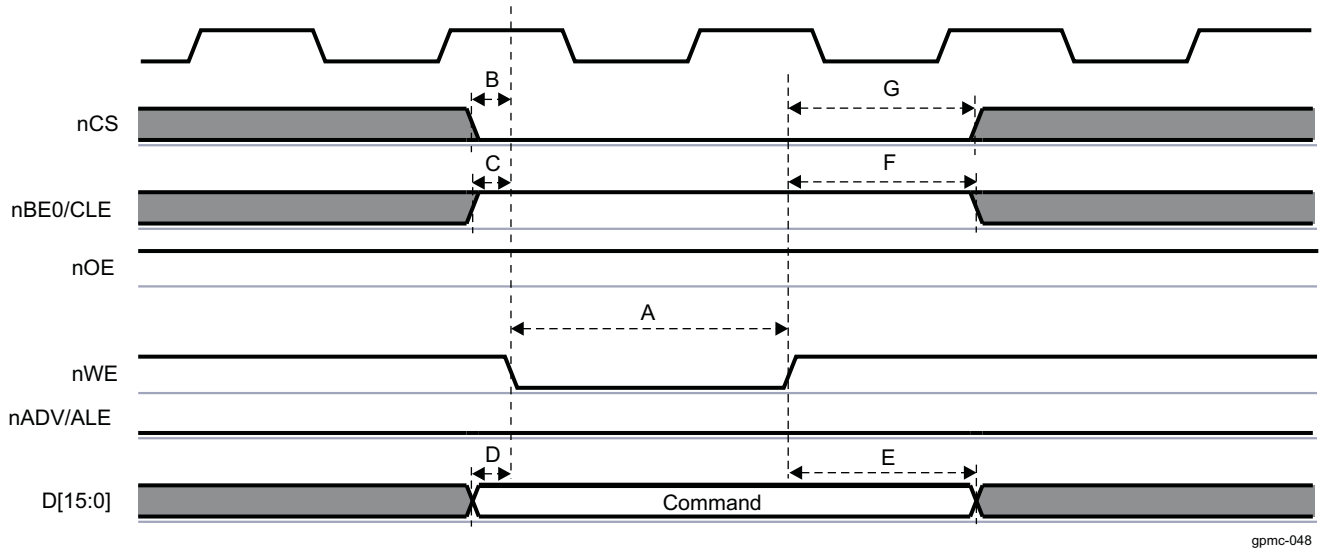
K = (OEOffTime – OEOnTime) * (1 + TimeParaGranularity) * GPMC_FCLK period

L = RdCycleTime * (1 + TimeParaGranularity) * GPMC_FCLK period

M = (CSRdOffTime – OEOffTime * (TimeParaGranularity + 1) + 0.5 * (CSExtraDelay - OEExtraDelay) * GPMC_FCLK period

Figure 7-202 shows a simplified example of command latch cycle timing where formulas are associated with signal waves.

Figure 7-202. NAND Command Latch Cycle Timing Simplified Example



7.3.5.6.1.2 Synchronous NOR Flash Timing Parameters Formulas

This section lists all formulas to calculate synchronous NOR timing parameters. This is the case when GPMC_CONFIG1_i[11-10] DEVICETYPE = 0h and when READTYPE or WRITETYPE are set to synchronous mode. Table 7-504 describes the synchronous NOR formulas.

Table 7-504. Synchronous NOR Formulas Description

Configuration Parameter	Unit	Description
A	ns	Pulse duration – GPMC_CS low
B	ns	Delay time – address bus valid to GPMC_CLK first edge
		Delay time – GPMC_BEn0_CLE/GPMC_BEn1 valid to GPMC_CLK first edge
C	ns	Pulse duration – GPMC_BEn0_CLE/GPMC_BEn1 low
D	ns	Delay time – GPMC_CLK rising edge to GPMC_BEn0_CLE/GPMC_BEn1 invalid
		Delay time – GPMC_CLK rising edge to GPMC_ADVn_ALE invalid
E	ns	Delay time – GPMC_CLK rising edge to GPMC_CS invalid
		Delay time – GPMC_CLK rising edge to GPMC_OEn_REn invalid
F	ns	Delay time – GPMC_CLK rising edge to GPMC_CS transition
G	ns	Delay time – GPMC_CLK rising edge to GPMC_ADVn_ALE transition
H	ns	Delay time – GPMC_CLK rising edge to GPMC_OEn_REn transition
I	ns	Delay time – GPMC_CLK rising edge to GPMC_WEn transition
J	ns	Delay time – GPMC_CLK rising edge to GPMC_AD data bus transition
		Delay time – GPMC_CLK rising edge to GPMC_BEn0_CLE/GPMC_BEn1 transition
K	ns	Pulse duration – GPMC_ADVn_ALE low
L	ns	Delay time – GPMC_WAIT invalid to first data latching GPMC_CLK edge

The configuration parameters are calculated through the following formulas. For more information, see the device Data Manual.

1. For single read accesses:

   A = (CSRDOFFTIME – CSONTIME) * (TIMEPARAGRANULARITY + 1) * GPMC_FCLK period
   C = RDCYCLETIME * (TIMEPARAGRANULARITY + 1) * GPMC_FCLK period
   D = (RDCYCLETIME – RDACCESSTIME) * GPMC_FCLK period
   E = (CSRDOFFTIME – RDACCESSTIME) * GPMC_FCLK period

2. For burst read accesses (where n is the page burst access number):

   A = (CSRDOFFTIME – CSONTIME + (n – 1) * PAGEBURSTACCESSTIME) * (TIMEPARAGRANULARITY + 1) * GPMC_FCLK period
   C = (RDCYCLETIME + (n – 1) * PAGEBURSTACCESSTIME) * (TIMEPARAGRANULARITY + 1) * GPMC_FCLK period
   D = (RDCYCLETIME – (RDACCESSTIME + (n – 1) * PAGEBURSTACCESSTIME)) * GPMC_FCLK period
   E = (CSRDOFFTIME – (RDACCESSTIME + (n – 1) * PAGEBURSTACCESSTIME)) * GPMC_FCLK period

3. For burst write accesses (where n is the page burst access number):

   A = (CSWROFFTIME – CSONTIME + (n – 1) * PAGEBURSTACCESSTIME) * (TIMEPARAGRANULARITY + 1) * GPMC_FCLK period
   C = (WRCYCLETIME + (n – 1) * PAGEBURSTACCESSTIME) * (TIMEPARAGRANULARITY + 1) * GPMC_FCLK period
   D = (WRCYCLETIME – (RDACCESSTIME + (n – 1) * PAGEBURSTACCESSTIME)) * GPMC_FCLK period
   E = (CSWROFFTIME – (RDACCESSTIME + (n – 1) * PAGEBURSTACCESSTIME)) * GPMC_FCLK period

4. For all accesses:

   For nCS falling edge (chip-select activated):

   - Case where GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER = 0h:
     F = 0.5 * CSEXTRADELAY * GPMC_FCLK period
   - Case where GPMCFCLKDIVIDER = 1h:
     F = 0.5 * CSEXTRADELAY * GPMC_FCLK period, when (CLKACTIVATIONTIME and CSONTIME are odd) or (CLKACTIVATIONTIME and CSONTIME are even)
     F = (1 + 0.5 * CSEXTRADELAY) * GPMC_FCLK period otherwise.
   - Case where GPMCFCLKDIVIDER = 2h:
     F = 0.5 * CSEXTRADELAY * GPMC_FCLK period, when (CSONTIME – CLKACTIVATIONTIME) is a multiple of 3
     F = (1 + 0.5 * CSEXTRADELAY) * GPMC_FCLK period, when (CSONTIME – CLKACTIVATIONTIME – 1) is a multiple of 3
     F = (2 + 0.5 * CSEXTRADELAY) * GPMC_FCLK period, when (CSONTIME – CLKACTIVATIONTIME – 2) is a multiple of 3

   For nCS rising edge (chip-select deactivated) in reading mode:

   - Case where GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER = 0h:
     F = 0.5 * CSEXTRADELAY * GPMC_FCLK period
   - Case where GPMCFCLKDIVIDER = 1h:
     F = 0.5 * CSEXTRADELAY * GPMC_FCLK period, when (CLKACTIVATIONTIME and CSRDOFFTIME are odd) or (CLKACTIVATIONTIME and CSRDOFFTIME are even)
     F = (1 + 0.5 * CSEXTRADELAY) * GPMC_FCLK period otherwise.
   - Case where GPMCFCLKDIVIDER = 2h:
     F = 0.5 * CSEXTRADELAY * GPMC_FCLK period, when (CSRDOFFTIME – CLKACTIVATIONTIME) is a multiple of 3
     F = (1 + 0.5 * CSEXTRADELAY) * GPMC_FCLK period, when (CSRDOFFTIME – CLKACTIVATIONTIME – 1) is a multiple of 3
     F = (2 + 0.5 * CSEXTRADELAY) * GPMC_FCLK period, when (CSRDOFFTIME – CLKACTIVATIONTIME – 2) is a multiple of 3

   For nCS rising edge (chip-select deactivated) in writing mode:

- Case where GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER = 0h:
  F = 0.5 * CSEXTRADELAY * GPMC_FCLK period
- Case where GPMCFCLKDIVIDER = 1h:
  F = 0.5 * CSEXTRADELAY * GPMC_FCLK period, when (CLKACTIVATIONTIME and CSWROFFTIME are odd) or (CLKACTIVATIONTIME and CSWROFFTIME are even)
  F = (1 + 0.5 * CSEXTRADELAY) * GPMC_FCLK period otherwise.
- Case where GPMCFCLKDIVIDER = 2h:
  F = 0.5 * CSEXTRADELAY * GPMC_FCLK period, when (CSWROFFTIME – CLKACTIVATIONTIME) is a multiple of 3
  F = (1 + 0.5 * CSEXTRADELAY) * GPMC_FCLK period, when (CSWROFFTIME – CLKACTIVATIONTIME – 1) is a multiple of 3
  F = (2 + 0.5 * CSEXTRADELAY) * GPMC_FCLK period, when (CSWROFFTIME – CLKACTIVATIONTIME – 2) is a multiple of 3

For nADV falling edge (nADV activated):

- Case where GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER = 0h:
  G = 0.5 * ADVEXTRADELAY * GPMC_FCLK period
- Case where GPMCFCLKDIVIDER = 1h:
  G = 0.5 * ADVEXTRADELAY * GPMC_FCLK period, when (CLKACTIVATIONTIME and ADVONTIME are odd) or (CLKACTIVATIONTIME and ADVONTIME are even)
  G = (1 + 0.5 * ADVEXTRADELAY) * GPMC_FCLK period otherwise.
- Case where GPMCFCLKDIVIDER = 2h:
  G = 0.5 * ADVEXTRADELAY * GPMC_FCLK period, when (ADVONTIME – CLKACTIVATIONTIME) is a multiple of 3
  G = (1 + 0.5 * ADVEXTRADELAY) * GPMC_FCLK period, when (ADVONTIME – CLKACTIVATIONTIME – 1) is a multiple of 3
  G = (2 + 0.5 * ADVEXTRADELAY) * GPMC_FCLK period, when (ADVONTIME – CLKACTIVATIONTIME – 2) is a multiple of 3

For nADV rising edge (nADV deactivated) in reading mode:

- Case where GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER = 0h:
  G = 0.5 * ADVEXTRADELAY * GPMC_FCLK period
- Case where GPMCFCLKDIVIDER = 1h:
  G = 0.5 * ADVEXTRADELAY * GPMC_FCLK period, when (CLKACTIVATIONTIME and ADVRDOFFTIME are odd) or (CLKACTIVATIONTIME and ADVRDOFFTIME are even)
  G = (1 + 0.5 * ADVEXTRADELAY) * GPMC_FCLK period otherwise.
- Case where GPMCFCLKDIVIDER = 2h:
  G = 0.5 * ADVEXTRADELAY * GPMC_FCLK period, when (ADVRDOFFTIME - CLKACTIVATIONTIME) is a multiple of 3
  G = (1 + 0.5 * ADVEXTRADELAY) * GPMC_FCLK period, when (ADVRDOFFTIME - CLKACTIVATIONTIME - 1) is a multiple of 3
  G = (2 + 0.5 * ADVEXTRADELAY) * GPMC_FCLK period, when (ADVRDOFFTIME - CLKACTIVATIONTIME - 2) is a multiple of 3

For nADV rising edge (nADV deactivated) in writing mode:

- Case where GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER = 0h:
  G = 0.5 * ADVEXTRADELAY * GPMC_FCLK period
- Case where GPMCFCLKDIVIDER = 1h:
  G = 0.5 * ADVEXTRADELAY * GPMC_FCLK period, when (CLKACTIVATIONTIME and ADVWROFFTIME are odd) or (CLKACTIVATIONTIME and ADVWROFFTIME are even)
  G = (1 + 0.5 * ADVEXTRADELAY) * GPMC_FCLK period otherwise.
- Case where GPMCFCLKDIVIDER = 2h:
  G = 0.5 * ADVEXTRADELAY * GPMC_FCLK period, when (ADVWROFFTIME – CLKACTIVATIONTIME) is a multiple of 3
  G = (1 + 0.5 * ADVEXTRADELAY) * GPMC_FCLK period, when (ADVWROFFTIME – CLKACTIVATIONTIME – 1) is a multiple of 3
  G = (2 + 0.5 * ADVEXTRADELAY) * GPMC_FCLK period, when (ADVWROFFTIME – CLKACTIVATIONTIME – 2) is a multiple of 3

For nOE falling edge (nOE activated):

- Case where GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER = 0h:
  H = 0.5 * OEEXTRADELAY * GPMC_FCLK period
- Case where GPMCFCLKDIVIDER = 1h:
  H = 0.5 * OEEXTRADELAY * GPMC_FCLK period, when (CLKACTIVATIONTIME and OEONTIME are odd) or (CLKACTIVATIONTIME and OEONTIME are even)
  H = (1 + 0.5 * OEEXTRADELAY) * GPMC_FCLK period otherwise.
- Case where GPMCFCLKDIVIDER = 2h:
  H = 0.5 * OEEXTRADELAY * GPMC_FCLK period, when (OEONTIME – CLKACTIVATIONTIME) is a multiple of 3
  H = (1 + 0.5 * OEEXTRADELAY) * GPMC_FCLK period, when (OEONTIME – CLKACTIVATIONTIME – 1) is a multiple of 3
  H = (2 + 0.5 * OEEXTRADELAY) * GPMC_FCLK period, when (OEONTIME – CLKACTIVATIONTIME – 2) is a multiple of 3

For nOE rising edge (nOE deactivated):

- Case where GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER = 0h:
  H = 0.5 * OEEXTRADELAY * GPMC_FCLK period
- Case where GPMCFCLKDIVIDER = 1h:
  H = 0.5 * OEEXTRADELAY * GPMC_FCLK period, when (CLKACTIVATIONTIME and OEOFFTIME are odd) or (CLKACTIVATIONTIME and OEOFFTIME are even)
  H = (1 + 0.5 * OEEXTRADELAY) * GPMC_FCLK period otherwise.
- Case where GPMCFCLKDIVIDER = 2h:
  H = 0.5 * OEEXTRADELAY * GPMC_FCLK period, when (OEOFFTIME – CLKACTIVATIONTIME) is a multiple of 3
  H = (1 + 0.5 * OEEXTRADELAY) * GPMC_FCLK period, when (OEOFFTIME – CLKACTIVATIONTIME – 1) is a multiple of 3
  H = (2 + 0.5 * OEEXTRADELAY) * GPMC_FCLK period, when (OEOFFTIME – CLKACTIVATIONTIME – 2) is a multiple of 3

For nWE falling edge (nWE activated):

- Case where GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER = 0h:
  I = 0.5 * WEEXTRADELAY * GPMC_FCLK period
- Case where GPMCFCLKDIVIDER = 1h:
  I = 0.5 * WEEXTRADELAY * GPMC_FCLK period, when (CLKACTIVATIONTIME and WEONTIME are odd) or (CLKACTIVATIONTIME and WEONTIME are even)
  I = (1 + 0.5 * WEEXTRADELAY) * GPMC_FCLK period otherwise.
- Case where GPMCFCLKDIVIDER = 2h:
  I = 0.5 * WEEXTRADELAY * GPMC_FCLK period, when (WEONTIME – CLKACTIVATIONTIME) is a multiple of 3
  I = (1 + 0.5 * WEEXTRADELAY) * GPMC_FCLK period, when (WEONTIME – CLKACTIVATIONTIME – 1) is a multiple of 3
  I = (2 + 0.5 * WEEXTRADELAY) * GPMC_FCLK period, when (WEONTIME – CLKACTIVATIONTIME – 2) is a multiple of 3

For nWE rising edge (nWE deactivated):

- Case where GPMC_CONFIG1_i[1-0] GPMCFCLKDIVIDER = 0h:
  I = 0.5 * WEEXTRADELAY * GPMC_FCLK period
- Case where GPMCFCLKDIVIDER = 1h:
  I = 0.5 * WEEXTRADELAY * GPMC_FCLK period, when (CLKACTIVATIONTIME and WEOFFTIME are odd) or (CLKACTIVATIONTIME and WEOFFTIME are even)
  I = (1 + 0.5 * WEEXTRADELAY) * GPMC_FCLK period otherwise.
- Case where GPMCFCLKDIVIDER = 2h:
  I = 0.5 * WEEXTRADELAY * GPMC_FCLK period, when (WEOFFTIME – CLKACTIVATIONTIME) is a multiple of 3
  I = (1 + 0.5 * WEEXTRADELAY) * GPMC_FCLK period, when (WEOFFTIME – CLKACTIVATIONTIME – 1) is a multiple of 3
  I = (2 + 0.5 * WEEXTRADELAY) * GPMC_FCLK period, when (WEOFFTIME – CLKACTIVATIONTIME – 2) is a multiple of 3

For gpmc_nadv low pulse duration:

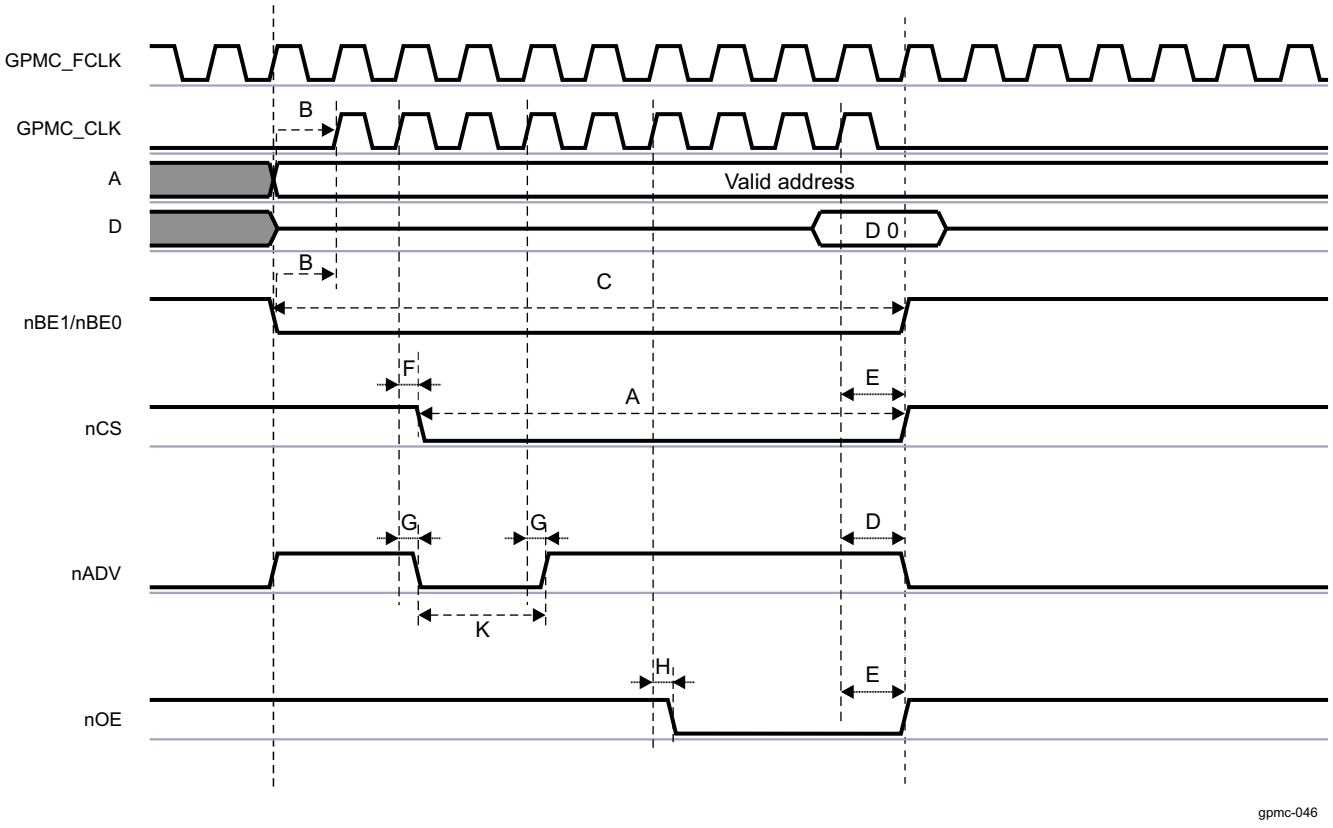
- Read operation:
  K = (ADVRDOFFTIME – ADVONTIME) * (TIMEPARAGRANULARITY + 1) * GPMC_FCLK period
- Write operation:
  K = (ADVWROFFTIME – ADVONTIME) * (TIMEPARAGRANULARITY + 1) * GPMC_FCLK period

For GPMC_WAIT invalid to first data latching GPMC_CLK edge:

- L = WAITMONITORINGTIME * (GPMCFCLKDIVIDER + 1) * GPMC_FCLK period + GPMC_CLK period

Figure 7-203 shows a simplified example of a synchronous NOR single read where formulas are associated with signal waves.

Figure 7-203. Synchronous NOR Single Read Simplified Example



7.3.5.6.1.3 Asynchronous NOR Flash Timing Parameters Formulas

This section lists all the formulas to calculate asynchronous NOR timing parameters. This is the case when GPMC_CONFIG1_i[11-10] DEVICETYPE = 0h and when READTYPE or WRITETYPE are set to asynchronous mode. Table 7-505 describes the asynchronous NOR formulas.

Table 7-505. Asynchronous NOR Formulas Description

Configuration Parameter	Unit	Description
A	ns	Pulse duration – GPMC_CS low
B	ns	Delay time – GPMC_CS valid to GPMC_ADVn_ALE invalid
C	ns	Delay time – GPMC_CS valid to GPMC_OEn_REn invalid (single read)
D	ns	Pulse duration – address bus valid - 2nd, 3rd and 4th accesses
E	ns	Delay time – GPMC_CS valid to GPMC_WEn valid
F	ns	Delay time – GPMC_CS valid to GPMC_WEn invalid
G	ns	Address invalid duration between two successive R/W accesses

Table 7-505. Asynchronous NOR Formulas Description (continued)

Configuration Parameter	Unit	Description
H	ns	Setup time – read data valid before GPMC_OEn_REn high
I	ns	Delay time – GPMC_CS valid to GPMC_OEn_REn invalid (burst read)
J	ns	Delay time – address bus valid to GPMC_CS valid
		Delay time – data bus valid to GPMC_CS valid
		Delay time – GPMC_BEn0_CLE/GPMC_BEn1 valid to GPMC_CS valid
K	ns	Delay time – GPMC_CS valid to GPMC_ADVn_ALE valid
L	ns	Delay time – GPMC_CS valid to GPMC_OEn_REn valid
M	ns	Delay time – GPMC_CS valid to first data latching edge
N	ns	Pulse duration – GPMC_BEn0_CLE/GPMC_BEn1 valid time
O	ns	Delay time – GPMC_CS valid to GPMC_ADVn_ALE valid

The configuration parameters are calculated through the following formulas. These formulas are not exhaustive. For more information, see the device Data Manual.

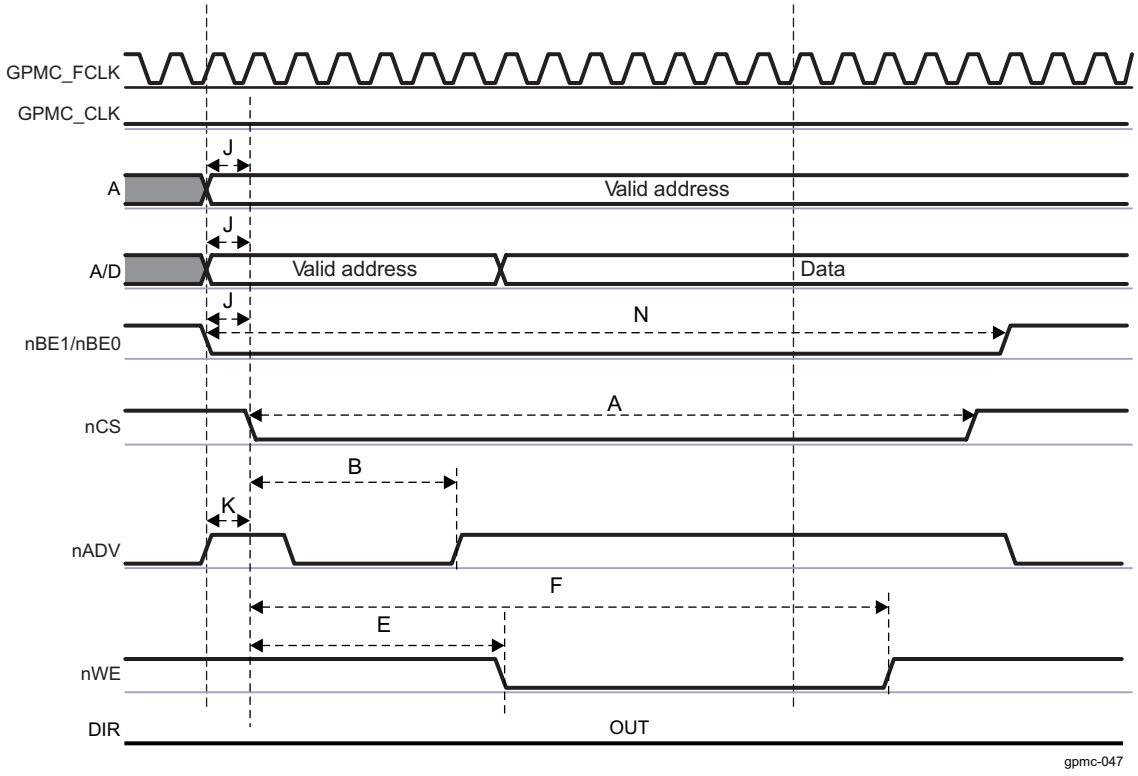
- GPMC_CS low pulse:
  For single read: A = (CSRDOFFTIME – CSONTIME) * (TIMEPARAGRANULARITY + 1) * GPMC_FCLK period
  For burst read: A = (CSRDOFFTIME – CSONTIME + (N – 1) * PAGEBURSTACCESSTIME) * (TIMEPARAGRANULARITY + 1) * GPMC_FCLK period, where N = page burst access number
  For single write: A = (CSWROFFTIME – CSONTIME) * (TIMEPARAGRANULARITY + 1) * GPMC_FCLK period
  For burst write: A = (CSWROFFTIME – CSONTIME + (N – 1) * PAGEBURSTACCESSTIME) * (TIMEPARAGRANULARITY + 1) * GPMC_FCLK period, where N = page burst access number
- GPMC_CS valid to GPMC_ADVn_ALE invalid delay:
  For reading: B = ((ADVRDOFFTIME – CSONTIME) * (TIMEPARAGRANULARITY + 1) + 0.5 * (ADVEXTRADELAY – CSEXTRADELAY)) * GPMC_FCLK period
  For writing: B = ((ADVWROFFTIME – CSONTIME) * (TIMEPARAGRANULARITY + 1) + 0.5 * (ADVEXTRADELAY – CSEXTRADELAY)) * GPMC_FCLK period
- C = ((OEOFFTIME – CSONTIME) * (TIMEPARAGRANULARITY + 1) + 0.5 * (OEEXTRADELAY – CSEXTRADELAY)) * GPMC_FCLK period
- D = PAGEBURSTACCESSTIME * (TIMEPARAGRANULARITY + 1) * GPMC_FCLK period
- E = ((WEONTIME – CSONTIME) * (TIMEPARAGRANULARITY + 1) + 0.5 * (WEEXTRADELAY – CSEXTRADELAY)) * GPMC_FCLK period
- F = ((WEOFFTIME – CSONTIME) * (TIMEPARAGRANULARITY + 1) + 0.5 * (WEEXTRADELAY – CSEXTRADELAY)) * GPMC_FCLK period
- G = CYCLE2CYCLEDELAY * GPMC_FCLK period
- H = ((OEOFFTIME – RDACCESSTIME) * (TIMEPARAGRANULARITY + 1) + 0.5 * OEEXTRADELAY) * GPMC_FCLK period
- I = ((OEOFFTIME + (N – 1) * PAGEBURSTACCESSTIME – CSONTIME) * (TIMEPARAGRANULARITY + 1) + 0.5 * (OEEXTRADELAY – CSEXTRADELAY)) * GPMC_FCLK period, where N = page burst access number
- J = (CSONTIME * (TIMEPARAGRANULARITY + 1) + 0.5 * CSEXTRADELAY) * GPMC_FCLK period
- K = ((ADVONTIME – CSONTIME) * (TIMEPARAGRANULARITY + 1) + 0.5 * (ADVEXTRADELAY – CSEXTRADELAY)) * GPMC_FCLK period
- L = ((OEONTIME – CSONTIME) * (TIMEPARAGRANULARITY + 1) + 0.5 * (OEEXTRADELAY – CSEXTRADELAY)) * GPMC_FCLK period
- M = ((RDACCESSTIME - CSONTIME) * (TIMEPARAGRANULARITY + 1) – 0.5 * CSEXTRADELAY) * GPMC_FCLK period
- GPMC_BEn0_CLE/GPMC_BEn1 pulse:
  For single read: N = RDCYCLETIME * (TIMEPARAGRANULARITY + 1) * GPMC_FCLK period

For burst read: N = (RDCYCLETIME + (N – 1) * PAGEBURSTACCESSTIME) * (TIMEPARAGRANULARITY + 1) * GPMC_FCLK period, where N = page burst access number
For burst write: N = (WRCYCLETIME + (N – 1) * PAGEBURSTACCESSTIME) * (TIMEPARAGRANULARITY + 1) * GPMC_FCLK period, where N = page burst access number

- O = ((WRCYCLETIME + (N – 1) * PAGEBURSTACCESSTIME – CSONTIME) * (TIMEPARAGRANULARITY + 1) + 0.5 * (ADVEXTRADELAY – CSEXTRADELAY)) * GPMC_FCLK period

Figure 7-204 shows a simplified example of an asynchronous NOR single write where formulas are associated with signal waves.

Figure 7-204. Asynchronous NOR Single Write Simplified Example



NOTE: Write multiple access is not supported in asynchronous mode. If WRITEMULTIPLE is enabled with WRITETYPE as asynchronous, the GPMC processes single asynchronous accesses.

7.3.6 GPMC Use Cases and Tips

7.3.6.1 How to Set GPMC Timing Parameters for Typical Accesses

7.3.6.1.1 External Memory Attached to the GPMC Module

As discussed in the introduction to this chapter, the GPMC module supports the following external memory types:

- Asynchronous or synchronous, 8- or 16-bit-wide memory or device
- 16-bit address/data-multiplexed or not multiplexed NOR flash device
- 8- or 16-bit NAND flash device

The following examples describe how to calculate GPMC timing parameters by showing a typical parameter setup for the access to be performed.

The example is based on a 512-Mb multiplexed NOR flash memory with the following characteristics:

- Type: NOR flash (address/data-multiplexed mode)
- Size: 512M bits
- Data Bus: 16 bits wide
- Speed: 104-MHz clock frequency
- Read access time: 80 ns

7.3.6.1.2 Typical GPMC Setup

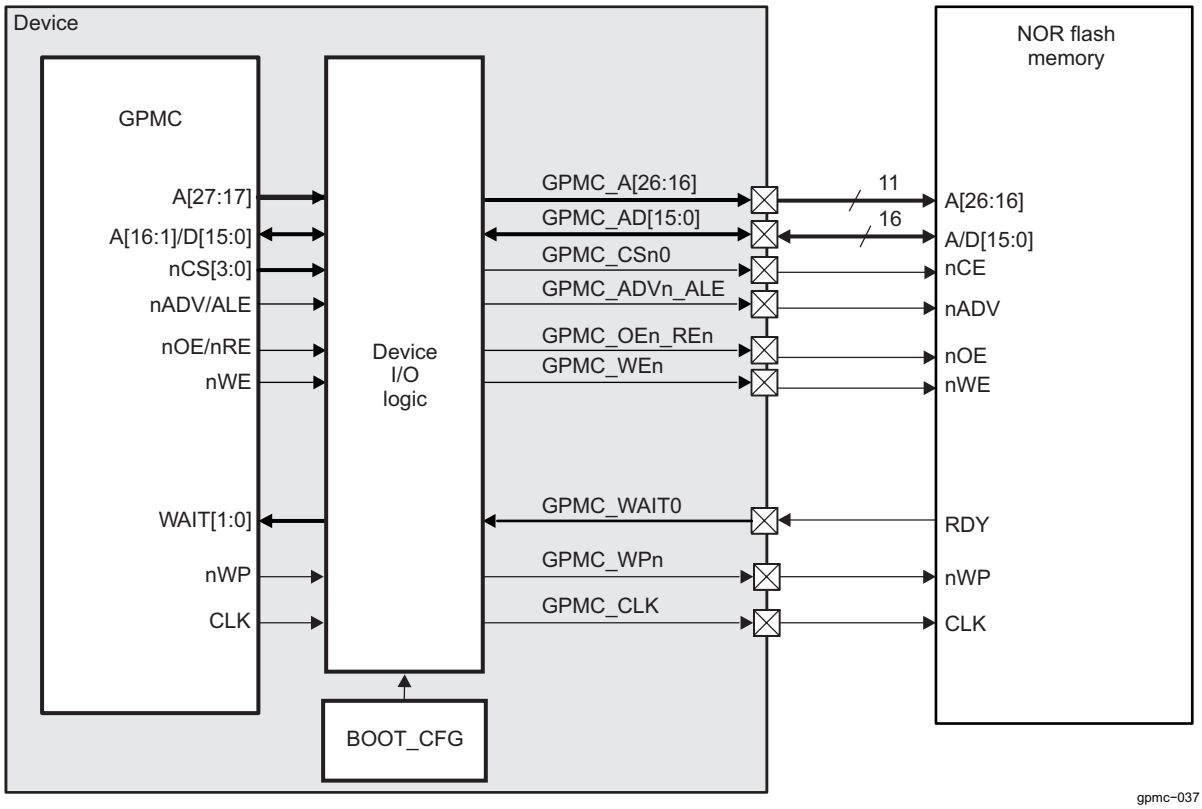
Table 7-506 lists some of the I/Os of the GPMC module.

Table 7-506. GPMC Signals

Signal Name	I/0	Description
GPMC_FCLK	Internal	Functional and interface clock. Acts as the time reference.
GPMC_CLK	O	External clock provided to the external device for synchronous operations
GPMC_A[26:16]	O	Address
GPMC_AD[15:0]	I/O	Data-multiplexed with addresses A[16:1] on memory side
GPMC_CSn[3:0]	O	Chip-selects
GPMC_ADVn_ALE	O	Address valid enable
GPMC_OEn_REn	O	Output enable (read access only)
GPMC_WEn	O	Write enable (write access only)
GPMC_WAIT[1:0]	I	Ready signal from memory device. Indicates when valid burst data is ready to be read

Figure 7-205 shows the typical connection between the GPMC module and an attached NOR Flash memory.

Figure 7-205. GPMC Connection to an External NOR Flash Memory



The following sections demonstrate how to calculate GPMC parameters for three access types:

- Synchronous burst read
- Asynchronous read
- Asynchronous single write

7.3.6.1.2.1 GPMC Configuration for Synchronous Burst Read Access

NOTE: The examples in Section 7.3.6.1.2.1 through Section 7.3.6.1.2.3 are based on a clock rate of 104 MHz. See the device Data Manual for the maximum frequency appropriate for this device and the memory datasheet for the maximum frequency for the particular memory device.

The clock runs at 104 MHz (f = 104 MHz; T = 9, 615 ns).

Table 7-507 shows the timing parameters (on the memory side) that determine the parameters on the GPMC side.

Table 7-508 shows how to calculate timings for the GPMC using the memory parameters.

Figure 7-206 shows the synchronous burst read access.

Table 7-507. Useful Timing Parameters on the Memory Side

AC Read Characteristics on the Memory Side	Description	Duration (ns)
tCES	nCS setup time to clock	0
tACS	Address setup time to clock	3

Table 7-507. Useful Timing Parameters on the Memory Side (continued)

AC Read Characteristics on the Memory Side	Description	Duration (ns)
tIACC	Synchronous access time	80
tBACC	Burst access time valid clock to output delay	5,2
tCEZ	Chip-select to High-Z	7
tOEZ	Output enable to High-Z	7
tAVC	nADV setup time	6
tAVD	nAVD pulse	6
tACH	Address hold time from clock	3

The following terms, which describe the timing interface between the controller and its attached device, are used to calculate the timing parameters on the GPMC side:

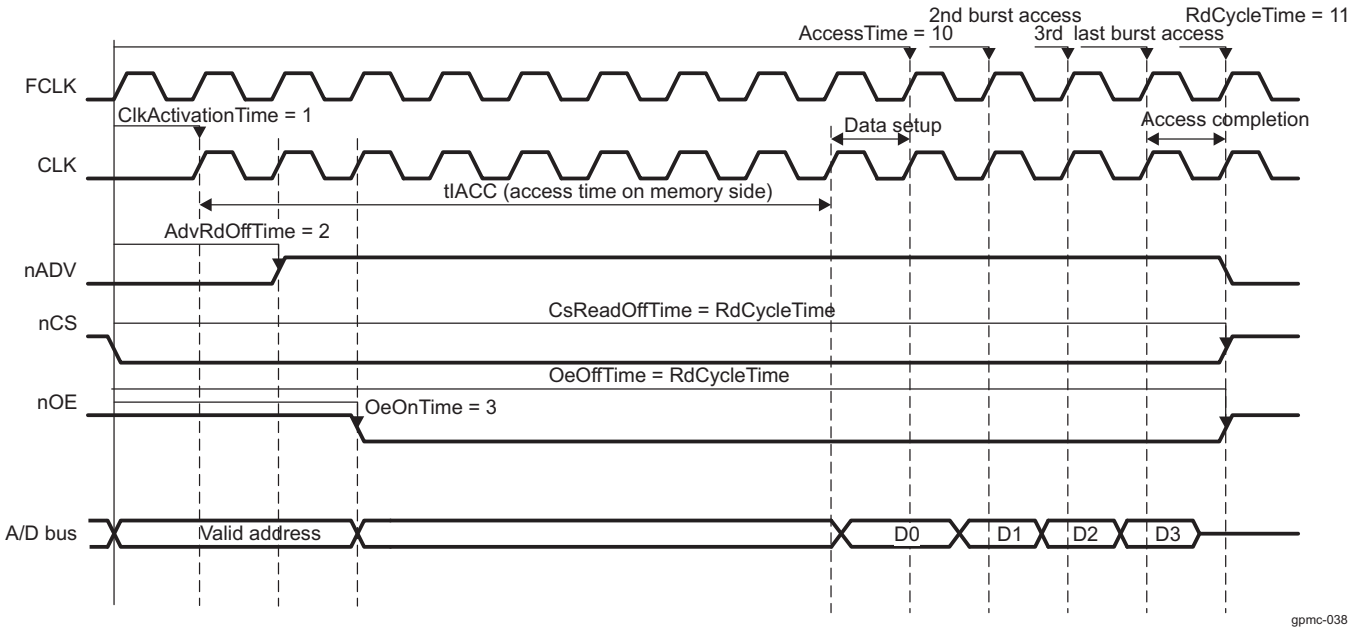
- Read access time (GPMC side): Time required to activate the clock + read access time requested on the memory side + data setup time required for optimal capture of a burst of data
- Data setup time (GPMC side): Ensures a good capture of a burst of data (as opposed to taking a burst of data out). One word of data is processed in one clock cycle (T = 9,615 ns). The read access time between two bursts of data is tBACC = 5.2 ns. Therefore, data setup time is a clock period – tBACC = 4,415 ns of data setup.
- Access completion (GPMC side): (Different from page burst access time) Time required between the last burst access and access completion: nCS/nOE hold time (nCS and nOE must be released at the end of an access. These signals are held to allow the access to complete).
- Read cycle time (GPMC side): Read access time + access completion
- Write cycle time for burst access: Not supported for NOR flash memory

Table 7-508. Calculating GPMC Timing Parameters

Parameter Name on GPMC Side	Formula	Duration (ns)	Number of Clock Cycles (F = 104 MHz)	GPMC Register Configurations
GPMC FCLK Divider	–	–	–	GPMCFCLKDIVIDER = 0h
ClkActivation Time	min (tCES, tACS)	3	1	CLKACTIVATIONTIME = 1h
RdAccessTime	roundmax (ClkActivationTime + tIACC + DataSetupTime)	94.03: (9,615 + 80 + 4,415)	10: roundmax (94.03 / 9,615)	RDACCESSTIME = Ah
PageBurst RdAccessTime	roundmax (tBACC)	roundmax (5.2)	1	PAGEBURSTACCESSTIME = 1h
RdCycleTime	RdAccess time + max (tCEZ, tOEZ)	101.03: (94.03 + 7)	11	RDCYCLETIME = Bh
CsOnTime	tCES	0	0	CSONTIME = 0h
CsReadOffTime	RdCycleTime	-	11	CSRDOFFTIME = Bh
AdvOnTime	tAVC (1)	0	0	ADVONTIME = 0h
AdvRdOffTime	tAVD + tAVC (2)	12	2	ADVRDOFFTIME = 2h
OeOnTime (3)	(ClkActivationTime + tACH) < OeOnTime (ClkActivationTime + tIACC)	–	3, for instance	OEONTIME = 3h
OeOffTime	RdCycleTime	–	11	OEOFFTIME = Bh

(1) The external clock provided to the NOR flash is not yet available.
(2) AdvRdOffTime – AdvOnTime = tAVD; thus, AdvRdOffTime = tAVD + AdvOnTime = tAVD + tAVC.
(3) OeOnTime must ensure that addresses are available. It must not exceed the availability of the first burst of data.

Figure 7-206. Synchronous Burst Read Access (Timing Parameters in Clock Cycles)



7.3.6.1.2.2 GPMC Configuration for Asynchronous Read Access

The clock runs at 104 MHz (f = 104 MHz; T = 9, 615 ns).

Table 7-509 shows the timing parameters (on the memory side) that determine the parameters on the GPMC side.

Table 7-510 shows how to calculate timings for the GPMC using the memory parameters.

Figure 7-207 shows the asynchronous read access.

Table 7-509. AC Characteristics for Asynchronous Read Access

AC Read Characteristics on the Memory Side	Description	Duration (ns)
tCE	Read Access time from nCS low	80
tAAVDS	Address setup time to rising edge of nADV	3
tAVDP	nADV low time	6
tCAS	nCS setup time to nADV	0
tOE	Output enable to output valid	6
tOEZ	Output enable to High-Z	7

Use the following formula to calculate the RdCycleTime parameter for this typical access:

RdCycleTime = RdAccessTime + AccessCompletion = RdAccessTime + 1 clock cycle + tOEZ:

1. On the memory side, the external memory makes the data available to the output bus. This is the memory-side read access time defined in Table 7-509: the number of clock cycles between the address capture (nADV rising edge) and the data valid on the output bus.

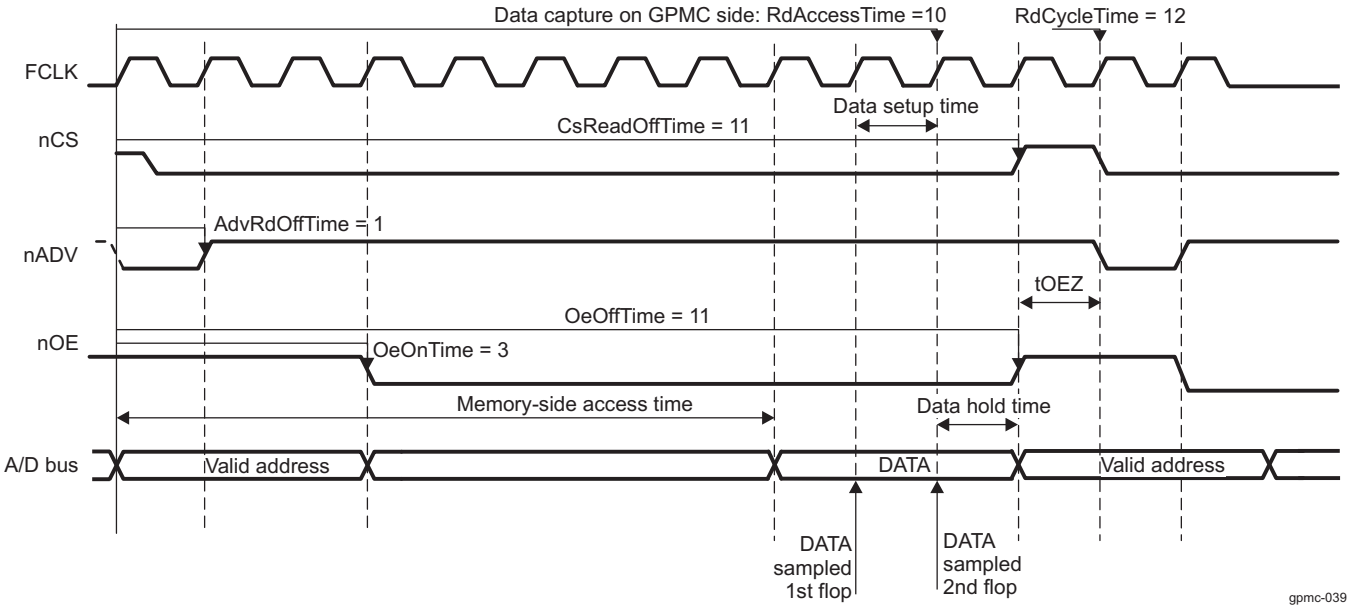
   The GPMC requires some hold time to allow the data to be captured correctly and the access to be finished.
2. To read the data correctly, the GPMC must be configured to meet the data setup time requirement of the memory; the GPMC module captures the data on the next rising edge. This is access time on the GPMC side.
3. There must also be a data hold time for correctly reading the data (checking that there is no nOE/nCS deassertion while reading the data). This data hold time is one clock cycle (that is, RdAccessTime + 1).

4. To complete the access, nOE/nCS signals are driven to High-Z. RdAccessTime + 1 + tOEZ is the read cycle time.
5. Addresses can now be relatched and a new read cycle begun.

Table 7-510. GPMC Timing Parameters for Asynchronous Read Access

Parameter Name on GPMC Side	Formula	Duration (ns)	Number of Clock Cycles (F = 104 MHz)	GPMC Register Configurations
ClkActivationTime	n/a (asynchronous mode)			
RdAccessTime	round max (tCE)	80	10	RDACCESSTIME = Ah
PageBurstAccess Time	N/A (single access)			
RdCycleTime	RdAccessTime + 1cycle + tOEZ	96, 615	12	RDCYCLETIME = Ch
CsOnTime	tCAS	0	0	CSONTIME = 0h
CsReadOffTime	RdAccessTime + 1 cycle	89, 615	11	CSRDOFFTIME = Bh
AdvOnTime	tAAVDS	3	1	ADVONTIME = 1h
AdvRdOffTime	tAAVDS + tAVDP	9	1	ADVRDOFFTIME = 01h
OeOnTime	OeOnTime >= AdvRdOffTime (multiplexed mode)	-	3, for instance	OEONTIME = 3h
OeOffTime	RdAccessTime + 1cycle	89, 615	11	OEOFFTIME = Bh

Figure 7-207. Asynchronous Single Read Access (Timing Parameters in Clock Cycles)



7.3.6.1.2.3 GPMC Configuration for Asynchronous Single Write Access

The clock runs at 104 MHz: (f = 104 MHz; T = 9, 615 ns).

Table 7-512 shows how to calculate timings for the GPMC using the memory parameters.

Table 7-511 shows the timing parameters (on the memory side) that determine the parameters on the GPMC side.

Figure 7-208 shows the synchronous burst write access.

Table 7-511. AC Characteristics for Asynchronous Single Write (Memory Side)

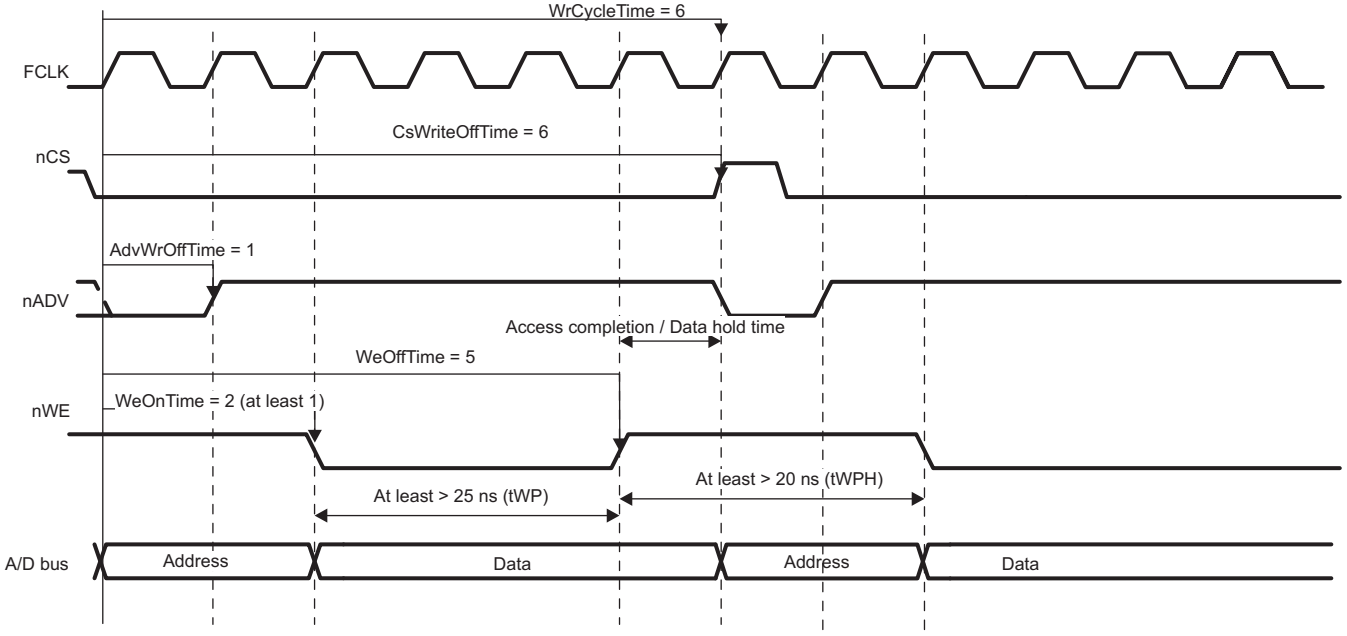
AC Characteristics on the Memory Side	Description	Duration (ns)
tWC	Write cycle time	60
tAVDP	nADV low time	6
tWP	Write pulse width	25
tWPH	Write pulse width high	20
tCS	nCS setup time to nWE	3
tCAS	nCS setup time to nADV	0
tAVSC	nADV setup time	3

For asynchronous single write access, write cycle time is WrCycleTime = WeOffTime + AccessCompletion = WeOffTime + 1. For the AccesCompletion, the GPMC requires one cycle of data hold time (nCS deassertion). For more information, see the device Data Manual.

Table 7-512. GPMC Timing Parameters for Asynchronous Single Write

Parameter Name on GPMC Side	Formula	Duration (ns)	Number of Clock Cycles (F = 104 MHz)	GPMC Registers Configuration
ClkActivationTime	N/A (asynchronous mode)			
WdAccessTime	Applicable only to WAITMONITORING (the value is the same as for read access)			
PageBurstAccessTime	N/A (single access)			
WrCycleTime	WeOffTime + AccessCompletion	57, 615	6	WRCYCLETIME = 6h
CsOnTime	tCAS	0	0	CSONTIME = 0h
CsWrOffTime	WeOffTime + 1	57, 615	6	CSWROFFTIME = 6h
AdvOnTime	tAVSC	3	1	ADVONTIME = 1h
AdvWrOffTime	tAVSC + tAVDP	9	1	ADVWROFFTIME = 1h
WeOnTime	tCS	3	1	WEONTIME = 1h
WeOffTime	tCS + tWP + tWPH	48	5	WEOFFTIME = 5h

Figure 7-208. Asynchronous Single Write Access (Timing Parameters in Clock Cycles)



7.3.6.2 How to Choose a Suitable Memory to Use With the GPMC

This section is intended to help the user select a suitable memory device to interface with the GPMC controller.

7.3.6.2.1 Supported Memories or Devices

NAND flash and NOR flash architectures are the two flash technologies. The GPMC supports various types of external memory or devices, basically any one that supports NAND or NOR protocols:

- 8- and 16-bit-wide asynchronous or synchronous memory or device (only 8-bit: nonburst device)
- 16-bit address and data-multiplexed NOR flash devices (pSRAM, OneNAND™, and so on)
- 8- and 16-bit NAND flash devices

7.3.6.2.1.1 Memory Pin Multiplexing

This section describes the interfacing differences of the GPMC supported memories.

Table 7-513. Supported Memory Interfaces

Function	16-Bit Address/ Data-Multiplexed pSRAM or NOR Flash (1)	OneNAND	16-Bit NAND	8-Bit NAND
GPMC_A[27]	A27			
GPMC_A[26]				
GPMC_A[25]				
GPMC_A[24]				
GPMC_A[23]				
GPMC_A[22]				
GPMC_A[21]				
GPMC_A[20]				
GPMC_A[19]				
GPMC_A[18]				
GPMC_A[17]				
GPMC_A[16]				
GPMC_A[15]				
GPMC_A[14]				
GPMC_A[13]				
GPMC_A[12]				
GPMC_A[11]				
GPMC_A[10]	A26			
GPMC_A[9]	A25			
GPMC_A[8]	A24			
GPMC_A[7]	A23			
GPMC_A[6]	A22			
GPMC_A[5]	A21			
GPMC_A[4]	A20			
GPMC_A[3]	A19			
GPMC_A[2]	A18			
GPMC_A[1]	A17			
GPMC_A[0]	A16			
GPMC_AD[15]	D15 or A16		IO15	
GPMC_AD[14]	D14 or A15		IO14	

(1) Addresses seen from the device side. When interfacing to the external device, A1 is connected to the memory A0, A2 to the memory A1, and so on.

Table 7-513. Supported Memory Interfaces (continued)

Function	16-Bit Address/ Data-Multiplexed pSRAM or NOR Flash [1]	OneNAND	16-Bit NAND	8-Bit NAND
GPMC_AD[13]	D13 or A14		IO13	
GPMC_AD[12]	D12 or A13		IO12	
GPMC_AD[11]	D11 or A12		IO11	
GPMC_AD[10]	D10 or A11		IO10	
GPMC_AD[9]	D9 or A10		IO9	
GPMC_AD[8]	D8 or A9		IO8	
GPMC_AD[7]	D7 or A8		IO7	
GPMC_AD[6]	D6 or A7		IO6	
GPMC_AD[5]	D5 or A6		IO5	
GPMC_AD[4]	D4 or A5		IO4	
GPMC_AD[3]	D3 or A4		IO3	
GPMC_AD[2]	D2 or A3		IO2	
GPMC_AD[1]	D1 or A2		IO1	
GPMC_AD[0]	D0 or A1		IO0	
GPMC_CLK	CLK			
GPMC_CSn0	nCS0 (chip-select)		nCE0 (chip-enable)	
GPMC_CSn1	nCS1		nCE1	
GPMC_CSn2	nCS2		nCE2	
GPMC_CSn3	nCS3		nCE3	
GPMC_ADVn_ALE	nADV (address valid)		ALE (address latch enable)	
GPMC_OEn_REn	nOE (output enable)		nRE (read enable)	
GPMC_WEn	nWE (Write enable)		nWE (write enable)	
GPMC_BEn0_CLE	nBE0 (byte enable)		CLE (command latch enable)	
GPMC_BEn1	nBE1			
GPMC_WAIT0	WAIT0		R/nB0 (ready/busy)	
GPMC_WAIT1	WAIT1		R/nB1	
GPMC_WPn	nWP (Write Protect)		nWP (Write Protect)	

7.3.6.2.1.2 NAND Interface Protocol

NAND flash architecture, introduced in 1989, is a flash technology. NAND is a page-oriented memory device; that is, read and write accesses are done by pages. NAND achieves great density by sharing common areas of the storage transistor, which creates strings of serially connected transistors (in NOR devices, each transistor stands alone). Because of its high density NAND is best suited to devices that require high capacity data storage, such as pictures, music, and data files. NAND nonvolatility makes of it a good storage solution for many applications where mobility, low power, and speed are key factors. Low pin count and simple interface are other advantages of NAND.

Table 7-514 summarizes the NAND interface signals level applied to external device or memories.

Table 7-514. NAND Interface Bus Operations Summary

Bus Operation	CLE	ALE	nCE	nWE [1]	nRE [1]	nWP
Read (cmd input)	H	L	L	RE	H	x
Read (add input)	L	H	L	RE	H	x
Write (cmd input)	H	L	L	RE	H	H
Write (add input)	L	H	L	RE	H	H
Data input	L	L	L	RE	H	H

(1) RE stands for rising edge; FE stands for falling edge

Table 7-514. NAND Interface Bus Operations Summary (continued)

Bus Operation	CLE	ALE	nCE	nWE [1]	nRE [1]	nWP
Data output	L	L	L	H	FE	x
Busy (during read)	x	x	H [2]	H [2]	H [2]	x
Busy (during program)	x	x	x	x	x	H
Busy (during erase)	x	x	x	x	x	H
Write protect	x	x	x	x	x	L
Standby	x	x	H	x	x	H/L

[2] Can be nCE high, or WE and nRE high.

7.3.6.2.1.3 NOR Interface Protocol

NOR flash architecture, introduced in 1988, is a flash technology. Unlike NAND, which is a sequential access device, NOR is directly addressable; that is, it is designed to be a random access device. NOR is best suited to devices used to store and run code or firmware, usually in small capacities. While NOR has fast read capabilities, it also has slow write and erase functions when compared to the NAND architecture.

Table 7-515 summarizes the level of the NOR interface signals applied to external devices or memories.

Table 7-515. NOR Interface Bus Operations Summary

Bus Operation	CLK	nADV	nCS	nOE	nWE	WAIT	DQ[15:0]
Read (asynchronous)	x	L	L	L	H	Asserted	Output
Read (synchronous)	Running	L	L	L	H	Driven	Output
Read (burst suspend)	Halted	x	L	H	H	Active	Output
Write	x	L	L	H	L	Asserted	Input
Output disable	x	x	L	H	H	Asserted	High-Z
Standby	x	x	H	x	x	High-Z	High-Z

7.3.6.2.1.4 Other Technologies

Other supported device types interact with the GPMC through the NOR interface protocol.

OneNAND is a high-density, low-power memory device. OneNAND is based on single- or multilevel-cell NAND core with SRAM and logic. It interfaces as a synchronous NOR flash and has synchronous write capability. It reads faster than conventional NAND and writes faster than conventional NOR flash. Hence, it is appropriate for mass storage and code storage.

pSRAM (pseudo-static random access memory) is a low-power memory device. pSRAM is based on the DRAM cell with internal refresh and address control features, and interfaces as a synchronous NOR flash. It also has synchronous write capability.

7.3.7 GPMC Registers

Table 7-517 lists the memory-mapped registers for the GPMC. All register offset addresses not listed in Table 7-517 should be considered as reserved locations and the register contents should not be modified.

NOTE: All GPMC registers are aligned to 32-bit address boundaries. All register file accesses, except to GPMC_NAND_DATA_i register, are little-endian. If the GPMC_NAND_DATA_i register location is accessed, the endianness is access-dependent.

Table 7-516. GPMC Instances

Instance	Base Address
GPMC	2181 8000h

Table 7-517. GPMC Registers

Offset	Acronym	Register Name	GPMC Physical Address	Section
0h	GPMC_REVISION	This register contains the IP revision code.	2181 8000h	Section 7.3.7.1
10h	GPMC_SYSCONFIG	Register related to module software reset and local power management.	2181 8010h	Section 7.3.7.2
14h	GPMC_SYSSTATUS	This register provides status information about the module, excluding the interrupt status information.	2181 8014h	Section 7.3.7.3
18h	GPMC_IRQSTATUS	This interrupt status register regroups all the status of the module internal events that can generate an interrupt.	2181 8018h	Section 7.3.7.4
1Ch	GPMC_IRQENABLE	The interrupt enable register allows to mask/unmask the module internal sources of interrupt, on a event-by-event basis.	2181 801Ch	Section 7.3.7.5
40h	GPMC_TIMEOUT_CONTROL	The GPMC_TIMEOUT_CONTROL register allows the user to set the start value of the timeout counter.	2181 8040h	Section 7.3.7.6
44h	GPMC_ERR_ADDRESS	The GPMC_ERR_ADDRESS register stores the address of the illegal access when an error occurs.	2181 8044h	Section 7.3.7.7
48h	GPMC_ERR_TYPE	The GPMC_ERR_TYPE register stores the type of error when an error occurs.	2181 8048h	Section 7.3.7.8
50h	GPMC_CONFIG	The configuration register allows global configuration of the GPMC.	2181 8050h	Section 7.3.7.9
54h	GPMC_STATUS	The status register provides global status bits of the GPMC.	2181 8054h	Section 7.3.7.10
60h + (i*30h)	GPMC_CONFIG1_i (i = 0 to 3)	The configuration register 1 sets signal control parameters per chip-select.	2181 8060h + (i*30h)	Section 7.3.7.11
64h + (i*30h)	GPMC_CONFIG2_i (i = 0 to 3)	CS signal timing parameter configuration	2181 8064h + (i*30h)	Section 7.3.7.12
68h + (i*30h)	GPMC_CONFIG3_i (i = 0 to 3)	nADV signal timing parameter configuration	2181 8068h + (i*30h)	Section 7.3.7.13
6Ch + (i*30h)	GPMC_CONFIG4_i (i = 0 to 3)	nWE and nOE signals timing parameter configuration	2181 806Ch + (i*30h)	Section 7.3.7.14
70h + (i*30h)	GPMC_CONFIG5_i (i = 0 to 3)	RdAccessTime and CycleTime timing parameters configuration	2181 8070h + (i*30h)	Section 7.3.7.15
74h + (i*30h)	GPMC_CONFIG6_i (i = 0 to 3)	WrAccessTime, WrDataOnADmuxBus, Cycle2Cycle and BusTurnAround parameters configuration	2181 8074h + (i*30h)	Section 7.3.7.16
78h + (i*30h)	GPMC_CONFIG7_i (i = 0 to 3)	CS address mapping configuration	2181 8078h + (i*30h)	Section 7.3.7.17
7Ch + (i*30h)	GPMC_NAND_COMMAND_i (i = 0 to 3)	This register is not a true register, only an address location.	2181 807Ch + (i*30h)	Section 7.3.7.18

Table 7-517. GPMC Registers (continued)

Offset	Acronym	Register Name	GPMC Physical Address	Section
80h + (i*30h)	GPMC_NAND_ADDRESS_i (i = 0 to 3)	This register is not a true register, only an address location.	2181 8080h + (i*30h)	Section 7.3.7.19
84h + (i*30h)	GPMC_NAND_DATA_i (i = 0 to 3)	This register is not a true register, only an address location.	2181 8084h + (i*30h)	Section 7.3.7.20
1E0h	GPMC_PREFETCH_CONFIG1	Prefetch engine configuration 1	2181 81E0h	Section 7.3.7.21
1E4h	GPMC_PREFETCH_CONFIG2	Prefetch engine configuration 2	2181 81E4h	Section 7.3.7.22
1ECh	GPMC_PREFETCH_CONTROL	Prefetch engine control	2181 81ECh	Section 7.3.7.23
1F0h	GPMC_PREFETCH_STATUS	Prefetch engine status	2181 81F0h	Section 7.3.7.24
1F4h	GPMC_ECC_CONFIG	ECC configuration	2181 81F4h	Section 7.3.7.25
1F8h	GPMC_ECC_CONTROL	ECC control	2181 81F8h	Section 7.3.7.26
1FCh	GPMC_ECC_SIZE_CONFIG	ECC size	2181 81FCh	Section 7.3.7.27
200h + (j*4h)	GPMC_ECCj_RESULT (j = 0 to 8)	ECC result register	2181 8200h + (j*4h)	Section 7.3.7.28
240h + (i*10h)	GPMC_BCH_RESULT0_i (i = 0 to 3)	BCH ECC result (bits 0 to 31)	2181 8240h + (i*10h)	Section 7.3.7.29
244h + (i*10h)	GPMC_BCH_RESULT1_i (i = 0 to 3)	BCH ECC result (bits 32 to 63)	2181 8244h + (i*10h)	Section 7.3.7.30
248h + (i*10h)	GPMC_BCH_RESULT2_i (i = 0 to 3)	BCH ECC result (bits 64 to 95)	2181 8248h + (i*10h)	Section 7.3.7.31
24Ch + (i*10h)	GPMC_BCH_RESULT3_i (i = 0 to 3)	BCH ECC result (bits 96 to 127)	2181 824Ch + (i*10h)	Section 7.3.7.32
2D0h	GPMC_BCH_SWDATA	This register is used to directly pass data to the BCH ECC calculator without accessing the actual NAND flash interface.	2181 82D0h	Section 7.3.7.33
300h + (i*10h)	GPMC_BCH_RESULT4_i (i = 0 to 3)	BCH ECC result (bits 128 to 159)	2181 8300h + (i*10h)	Section 7.3.7.34
304h + (i*10h)	GPMC_BCH_RESULT5_i (i = 0 to 3)	BCH ECC result (bits 160 to 191)	2181 8304h + (i*10h)	Section 7.3.7.35
308h + (i*10h)	GPMC_BCH_RESULT6_i (i = 0 to 3)	BCH ECC result (bits 192 to 207)	2181 8308h + (i*10h)	Section 7.3.7.36

7.3.7.1 GPMC_REVISION Register (Offset = 0h) [reset = 60h]

GPMC_REVISION is shown in Figure 7-209 and described in Table 7-519.

This register contains the IP revision code.

Table 7-518. GPMC_REVISION Instances

Instance	Physical Address
GPMC	2181 8000h

Figure 7-209. GPMC_REVISION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REVISION																															
R-60h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 7-519. GPMC_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REVISION	R	60h	TI internal data. Identifies revision of peripheral.

Table 7-520. Register Call Summary for GPMC_REVISION

GPMC Registers
- GPMC Registers: [0]
- GPMC_REVISION Register (Offset = 0h) [reset = 60h]: [0]

7.3.7.2 GPMC_SYSCONFIG Register (Offset = 10h) [reset = 0h]

GPMC_SYSCONFIG is shown in Figure 7-210 and described in Table 7-522.

Register related to module software reset and local power management.

Table 7-521. GPMC_SYSCONFIG Instances

Instance	Physical Address
GPMC	2181 8010h

Figure 7-210. GPMC_SYSCONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED			IDLEMODE		RESERVED	SOFTRESET	AUTOIDLE
R/W-0h			R/W-0h		R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-522. GPMC_SYSCONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0.
4-3	IDLEMODE	R/W	0h	0h: Force-idle. An idle request is acknowledged unconditionally. 1h (R/W) = No-idle. An idle request is never acknowledged. 2h (R/W) = Smart-idle. Acknowledgment to an idle request is given based on the internal activity of the module. 3h (R/W) = Reserved. Do not use.
2	RESERVED	R/W	0h	Write 0 for future compatibility Read returns 0.
1	SOFTRESET	R/W	0h	Software reset. Setting this bit to 1 triggers a module reset. This bit is automatically reset by hardware. During reads, it always returns 0. 0h (R/W) = Normal mode 1h (R/W) = The module is reset.
0	AUTOIDLE	R/W	0h	Internal interface clock-gating strategy 0h (R/W) = Interface clock is free-running. 1h (R/W) = Automatic Interface clock gating strategy is applied, based on the interconnect activity.

Table 7-523. Register Call Summary for GPMC_SYSCONFIG

GPMC Basic Programming Model • GPMC Initialization: [0]
GPMC Registers • GPMC Registers: [0] • GPMC_SYSCONFIG Register (Offset = 10h) [reset = 0h]: [0]

Table 7-523. Register Call Summary for GPMC_SYSCONFIG (continued)

GPMC Functional Description • GPMC Software Reset: [0] • GPMC Power Management: [0][1]

7.3.7.3 GPMC_SYSSTATUS Register (Offset = 14h) [reset = 0000000-h]

GPMC_SYSSTATUS is shown in Figure 7-211 and described in Table 7-525.

This register provides status information about the module, excluding the interrupt status information.

Table 7-524. GPMC_SYSSTATUS Instances

Instance	Physical Address
GPMC	2181 8014h

Figure 7-211. GPMC_SYSSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RESETDONE
R-0h							R-

LEGEND: R = Read Only; *-n* = value after reset

Table 7-525. GPMC_SYSSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Read returns 0.
7-1	RESERVED	R	0h	Read returns 0 (reserved for interconnect-socket status information).
0	RESETDONE	R	-h	Internal reset monitoring 0h (R) = Internal module reset is ongoing. 1h (R) = Reset is complete.

Table 7-526. Register Call Summary for GPMC_SYSSTATUS

GPMC Basic Programming Model • GPMC Initialization: [0]
GPMC Registers • GPMC Registers: [0] • GPMC_SYSSTATUS Register (Offset = 14h) [reset = 0000000-h]: [0]
GPMC Functional Description • GPMC Software Reset: [0]

7.3.7.4 GPMC_IRQSTATUS Register (Offset = 18h) [reset = 0h]

GPMC_IRQSTATUS is shown in Figure 7-212 and described in Table 7-528.

This interrupt status register regroups all the status of the module internal events that can generate an interrupt.

Table 7-527. GPMC_IRQSTATUS Instances

Instance	Physical Address
GPMC	2181 8018h

Figure 7-212. GPMC_IRQSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED						WAIT1EDGEDETECTIONSTATUS	WAIT0EDGEDETECTIONSTATUS
R/W-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED						TERMINALCOUNTSTATUS	FIFOEVENTSTATUS
R/W-0h						R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-528. GPMC_IRQSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
9	WAIT1EDGEDETECTIONSTATUS	R/W	0h	Status of the Wait1 Edge Detection interrupt Write: 0h = WAIT1EDGEDETECTIONSTATUS bit is unchanged. 1h = WAIT1EDGEDETECTIONSTATUS bit is reset. Read: 0h = A transition on WAIT1 input pin has not been detected. 1h = A transition on WAIT1 input pin has been detected.
8	WAIT0EDGEDETECTIONSTATUS	R/W	0h	Status of the Wait0 Edge Detection interrupt Write: 0h = WAIT0EDGEDETECTIONSTATUS bit is unchanged. 1h = WAIT0EDGEDETECTIONSTATUS bit is reset. Read: 0h = A transition on WAIT0 input pin has not been detected. 1h = A transition on WAIT0 input pin has been detected.
7-2	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
1	TERMINALCOUNTSTATUS	R/W	0h	Status of the TerminalCountEvent interrupt Write: 0h = TERMINALCOUNTSTATUS bit is unchanged. 1h = TERMINALCOUNTSTATUS bit is reset. Read: 0h = Indicates that CountValue is greater than 0. 1h = Indicates that CountValue is equal to 0.

Table 7-528. GPMC_IRQSTATUS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	FIFOEVENTSTATUS	R/W	0h	Status of the FIFOEvent interrupt Write: 0h = FIFOEVENTSTATUS bit is unchanged. 1h = FIFOEVENTSTATUS bit is reset. Read: 0h = Indicates that less than GPMC_PREFETCH_STATUS[16] FIFOTHRESHOLDSTATUS bytes are available in prefetch mode and less than FIFOTHRESHOLD bytes free places are available in write-posting mode. 1h = Indicates that at least GPMC_PREFETCH_STATUS[16] FIFOTHRESHOLDSTATUS bytes are available in prefetch mode and at least FIFOTHRESHOLD bytes free places are available in write-posting mode.

Table 7-529. Register Call Summary for GPMC_IRQSTATUS

GPMC Registers • GPMC Registers: [0] • GPMC_IRQSTATUS Register (Offset = 18h) [reset = 0h]: [0]
GPMC Functional Description • FIFO Control in Prefetch Mode: [0][1] • GPMC Interrupt Requests: [0][1][2][3] • FIFO Control in Write-Posting Mode: [0][1] • Ready Pin Monitored by Hardware Interrupt: [0][1][2]

7.3.7.5 GPMC_IRQENABLE Register (Offset = 1Ch) [reset = 0h]

GPMC_IRQENABLE is shown in Figure 7-213 and described in Table 7-531.

The interrupt enable register allows to mask/unmask the module internal sources of interrupt, on a event-by-event basis.

Table 7-530. GPMC_IRQENABLE Instances

Instance	Physical Address
GPMC	2181 801Ch

Figure 7-213. GPMC_IRQENABLE Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED						WAIT1EDGEDETECTIONENABLE	WAIT0EDGEDETECTIONENABLE
R/W-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED						TERMINALCOUNTEVENTENABLE	FIFOEVENTENABLE
R/W-0h						R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-531. GPMC_IRQENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
9	WAIT1EDGEDETECTIONENABLE	R/W	0h	Enables the Wait1 Edge Detection interrupt 0h (R/W) = Wait1EdgeDetection interrupt is masked. 1h (R/W) = Wait1EdgeDetection event generates an interrupt if occurs.
8	WAIT0EDGEDETECTIONENABLE	R/W	0h	Enables the Wait0 Edge Detection interrupt 0h (R/W) = Wait0EdgeDetection interrupt is masked. 1h (R/W) = Wait0EdgeDetection event generates an interrupt if occurs.
7-2	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
1	TERMINALCOUNTEVENTENABLE	R/W	0h	Enables TerminalCountEvent interrupt issuing in prefetch or write-posting mode 0h (R/W) = TerminalCountEvent interrupt is masked. 1h (R/W) = TerminalCountEvent interrupt is not masked.
0	FIFOEVENTENABLE	R/W	0h	Enables the FIFOEvent interrupt 0h (R/W) = FIFOEvent interrupt is masked. 1h (R/W) = FIFOEvent interrupt is not masked.

Table 7-532. Register Call Summary for GPMC_IRQENABLE

GPMC Registers • GPMC Registers: [0] • GPMC_IRQENABLE Register (Offset = 1Ch) [reset = 0h]: [0]
GPMC Functional Description • FIFO Control in Prefetch Mode: [0][1] • GPMC Interrupt Requests: [0][1][2][3] • FIFO Control in Write-Posting Mode: [0][1] • NAND Device-Ready Pin: [0] • Ready Pin Monitored by Hardware Interrupt: [0]

7.3.7.6 GPMC_TIMEOUT_CONTROL Register (Offset = 40h) [reset = 1FF0h]

GPMC_TIMEOUT_CONTROL is shown in Figure 7-214 and described in Table 7-534.

The GPMC_TIMEOUT_CONTROL register allows the user to set the start value of the timeout counter.

Table 7-533. GPMC_TIMEOUT_CONTROL Instances

Instance	Physical Address
GPMC	2181 8040h

Figure 7-214. GPMC_TIMEOUT_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED			TIMEOUTSTARTVALUE				
R/W-0h			R/W-1FFh				
7	6	5	4	3	2	1	0
TIMEOUTSTARTVALUE				RESERVED			TIMEOUTENABLE
R/W-1FFh				R/W-0h			R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-534. GPMC_TIMEOUT_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
12-4	TIMEOUTSTARTVALUE	R/W	1FFh	Start value of the time-out counter 000h: Zero GPMC_FCLK cycle 001h: One GPMC_FCLK cycle ... 1FFh: 511 GPMC_FCLK cycles
3-1	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
0	TIMEOUTENABLE	R/W	0h	Enable bit of the TimeOut feature 0h (R/W) = TimeOut feature is disabled. 1h (R/W) = TimeOut feature is enabled.

Table 7-535. Register Call Summary for GPMC_TIMEOUT_CONTROL

GPMC Registers • GPMC Registers: [0][1] • GPMC_TIMEOUT_CONTROL Register (Offset = 40h) [reset = 1FF0h]: [0][1]
GPMC Functional Description • Error Handling: [0][1]

7.3.7.7 GPMC_ERR_ADDRESS Register (Offset = 44h) [reset = 0h]

GPMC_ERR_ADDRESS is shown in Figure 7-215 and described in Table 7-537.

The GPMC_ERR_ADDRESS register stores the address of the illegal access when an error occurs.

Table 7-536. GPMC_ERR_ADDRESS Instances

Instance	Physical Address
GPMC	2181 8044h

Figure 7-215. GPMC_ERR_ADDRESS Register

31	30	29	28	27	26	25	24
RESERVED	ILLEGALADD						
R/W-0h	R-0h						
23	22	21	20	19	18	17	16
ILLEGALADD							
R-0h							
15	14	13	12	11	10	9	8
ILLEGALADD							
R-0h							
7	6	5	4	3	2	1	0
ILLEGALADD							
R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 7-537. GPMC_ERR_ADDRESS Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0.
30-0	ILLEGALADD	R	0h	Address of illegal access A30: 0 for memory region, 1 for GPMC register region A29-A0: 1GB maximum

Table 7-538. Register Call Summary for GPMC_ERR_ADDRESS

GPMC Registers • GPMC Registers: [0][1] • GPMC_ERR_ADDRESS Register (Offset = 44h) [reset = 0h]: [0][1]
GPMC Functional Description • Error Handling: [0]

7.3.7.8 GPMC_ERR_TYPE Register (Offset = 48h) [reset = 0h]

GPMC_ERR_TYPE is shown in Figure 7-216 and described in Table 7-540.

The GPMC_ERR_TYPE register stores the type of error when an error occurs.

Table 7-539. GPMC_ERR_TYPE Instances

Instance	Physical Address
GPMC	2181 8048h

Figure 7-216. GPMC_ERR_TYPE Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED					ILLEGALMCMD		
R/W-0h					R-0h		
7	6	5	4	3	2	1	0
RESERVED			ERRORNOTSUPPADD	ERRORNOTSUPPMCMD	ERRORTIMEOUT	RESERVED	ERRORVALID
R/W-0h			R-0h	R-0h	R-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 7-540. GPMC_ERR_TYPE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
10-8	ILLEGALMCMD	R	0h	System command of the transaction that caused the error
7-5	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
4	ERRORNOTSUPPADD	R	0h	Not supported address error 0h (R) = No error occurs. 1h (R) = The error is due to a nonsupported address.
3	ERRORNOTSUPPMCMD	R	0h	Not supported command error 0h (R) = No error occurs. 1h (R) = The error is due to a nonsupported command
2	ERRORTIMEOUT	R	0h	Time-out error 0h (R) = No error occurs. 1h (R) = The error is due to a timeout.
1	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0.
0	ERRORVALID	R/W	0h	Error validity status - Must be explicitly cleared with a write 1 transaction 0h (R/W) = All error fields no longer valid 1h (R/W) = Error detected and logged in the other error fields

Table 7-541. Register Call Summary for GPMC_ERR_TYPE

GPMC Registers
- GPMC Registers: [0][1]
- GPMC_ERR_TYPE Register (Offset = 48h) [reset = 0h]: [0][1]

Table 7-541. Register Call Summary for GPMC_ERR_TYPE (continued)

GPMC Functional Description
• Error Handling: [0][1][2]

7.3.7.9 GPMC_CONFIG Register (Offset = 50h) [reset = 200h]

GPMC_CONFIG is shown in Figure 7-217 and described in Table 7-543.

The configuration register allows global configuration of the GPMC.

Table 7-542. GPMC_CONFIG Instances

Instance	Physical Address
GPMC	2181 8050h

Figure 7-217. GPMC_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED						WAIT1PINPOLARITY	WAIT0PINPOLARITY
R/W-0h						R/W-1h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED			WRITEPROTECT	RESERVED			NANDFORCEPOSTEDWRITE
R/W-0h			R/W-0h	R/W-0h			R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-543. GPMC_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
9	WAIT1PINPOLARITY	R/W	1h	Selects the polarity of input pin WAIT1 0h (R/W) = WAIT1 active low 1h (R/W) = WAIT1 active high
8	WAIT0PINPOLARITY	R/W	0h	Selects the polarity of input pin WAIT0 0h (R/W) = WAIT0 active low 1h (R/W) = WAIT0 active high
7-5	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
4	WRITEPROTECT	R/W	0h	Controls the WP output pin level 0h (R/W) = nWP output pin is low 1h (R/W) = nWP output pin is high
3-1	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
0	NANDFORCEPOSTEDWRITE	R/W	0h	Enables the Force Posted Write feature to NAND Cmd/Add/Data location 0h (R/W) = Disables Force Posted Write 1h (R/W) = Enables Force Posted Write

Table 7-544. Register Call Summary for GPMC_CONFIG

GPMC Registers
• GPMC Registers: [0] • GPMC_CONFIG Register (Offset = 50h) [reset = 200h]: [0]

Table 7-544. Register Call Summary for GPMC_CONFIG (continued)

GPMC Functional Description
• GPMC Interrupt Requests: [0][1] • NAND Device Command and Address Phase Control: [0] • WAIT Pin Monitoring Control: [0] • Ready Pin Monitored by Hardware Interrupt: [0] • Write Protect Signal (nWP): [0]

7.3.7.10 GPMC_STATUS Register (Offset = 54h) [reset = 00000-01h]

GPMC_STATUS is shown in Figure 7-218 and described in Table 7-546.

The status register provides global status bits of the GPMC.

Table 7-545. GPMC_STATUS Instances

Instance	Physical Address
GPMC	2181 8054h

Figure 7-218. GPMC_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED						WAIT1STATUS	WAIT0STATUS
R/W-0h						R-	R-
7	6	5	4	3	2	1	0
RESERVED							EMPTYWRITEBUFFERSTATUS
R/W-0h							R-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 7-546. GPMC_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
9	WAIT1STATUS	R	-h	Is a copy of input pin WAIT1. (Reset value is WAIT1 input pin sampled at device reset.) 0h (R) = WAIT1 asserted (inactive state) 1h (R) = WAIT1 deasserted
8	WAIT0STATUS	R	-h	Is a copy of input pin WAIT0. (Reset value is WAIT0 input pin sampled at device reset.) 0h (R) = WAIT0 asserted (inactive state) 1h (R) = WAIT0 deasserted
7-1	RESERVED	R/W	0h	Write 0s for future compatibility. Reads returns 0
0	EMPTYWRITEBUFFERSTATUS	R	1h	Stores the empty status of the write buffer 0h (R) = Write buffer is not empty. 1h (R) = Write buffer is empty.

Table 7-547. Register Call Summary for GPMC_STATUS

GPMC Registers • GPMC Registers: [0] • GPMC_STATUS Register (Offset = 54h) [reset = 00000-01h]: [0]
GPMC Functional Description • NAND Device Command and Address Phase Control: [0] • NAND Device-Ready Pin: [0][1] • Ready Pin Monitored by Software Polling: [0]

7.3.7.11 GPMC_CONFIG1_i Register (Offset = 60h + i*30h) [reset = 0h]

GPMC_CONFIG1_i (where i = 0 to 3) is shown in Figure 7-219 and described in Table 7-549.

The configuration register 1 sets signal control parameters per chip-select.

Table 7-548. GPMC_CONFIG1_i Instances

Instance	Physical Address
GPMC	2181 8060h + (i*30h)

Figure 7-219. GPMC_CONFIG1_i Register

31	30	29	28	27	26	25	24
WRAPBURST	READMULTIPLE	READTYPE	WRITEMULTIPLE	WRITETYPE	CLKACTIVATIONTIME		ATTACHEDDEVICEPAGELENGTH
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h
23	22	21	20	19	18	17	16
ATTACHEDDEVICEPAGELENGTH	WAITREADMONITORING	WAITWRITEMONITORING	RESERVED	WAITMONITORINGTIME		WAITPINSELECT	
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
RESERVED		DEVICESIZE		DEVICETYPE		MUXADDDATA	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED			TIMEPARAGRANULARITY	RESERVED		GPMCFCLKDIVIDER	
R/W-0h			R/W-0h	R/W-0h		R/W-0h	

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-549. GPMC_CONFIG1_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31	WRAPBURST	R/W	0h	Enables the wrapping burst capability. Must be set if the attached device is configured in wrapping burst 0h (R/W) = Synchronous wrapping burst not supported 1h (R/W) = Synchronous wrapping burst supported
30	READMULTIPLE	R/W	0h	Selects the read single or multiple access 0h (R/W) = Single access 1h (R/W) = Multiple access (burst if synchronous, page if asynchronous)
29	READTYPE	R/W	0h	Selects the read mode operation 0h (R/W) = Read asynchronous 1h (R/W) = Read synchronous
28	WRITEMULTIPLE	R/W	0h	Selects the write single or multiple access 0h (R/W) = Single access 1h (R/W) = Multiple access (burst if synchronous, considered as single if asynchronous)
27	WRITETYPE	R/W	0h	Selects the write mode operation 0h (R/W) = Write asynchronous 1h (R/W) = Write synchronous

Table 7-549. GPMC_CONFIG1_i Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26-25	CLKACTIVATIONTIME	R/W	0h	Output GPMC_CLK activation time 0h (R/W) = First rising edge of GPMC_CLK at start access time 1h (R/W) = First rising edge of GPMC_CLK one GPMC_FCLK cycle after start access time 2h (R/W) = First rising edge of GPMC_CLK two GPMC_FCLK cycles after start access time 3h (R/W) = Reserved
24-23	ATTACHEDDEVICEPAGELENGTH	R/W	0h	Specifies the attached device page (burst) length 0h (R/W) = 4 words 1h (R/W) = 8 words 2h (R/W) = 16 words 3h (R/W) = Reserved (1 word = interface size)
22	WAITREADMONITORING	R/W	0h	Selects the Wait monitoring configuration for Read accesses 0h (R/W) = WAIT pin is not monitored for read accesses. 1h (R/W) = WAIT pin is monitored for read accesses.
21	WAITWRITEMONITORING	R/W	0h	Selects the Wait monitoring configuration for Write accesses 0h (R/W) = WAIT pin is not monitored for write accesses. 1h (R/W) = WAIT pin is monitored for write accesses.
20	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0.
19-18	WAITMONITORINGTIME	R/W	0h	Selects input pin Wait monitoring time 0h (R/W) = WAIT pin is monitored with valid data. 1h (R/W) = WAIT pin is monitored one GPMC_CLK cycle before valid data. 2h (R/W) = WAIT pin is monitored two GPMC_CLK cycle before valid data. 3h (R/W) = Reserved
17-16	WAITPINSELECT	R/W	0h	Selects the input WAIT pin for this chip-select 0h (R/W) = Wait input pin is WAIT0. 1h (R/W) = Wait input pin is WAIT1. 2h, 3h: Reserved .
15-14	RESERVED	R/W	0h	Write 0s for future compatibility. Reads returns 0
13-12	DEVICESIZE	R/W	0h	Selects the device size attached **NOTE: Reset value is 0h for CS0 and 1h for CS1 to CS3** 0h (R/W) = 8 bit 1h (R/W) = 16 bit 2h (R/W) = Reserved 3h (R/W) = Reserved
11-10	DEVICETYPE	R/W	0h	Selects the attached device type 0h (R/W) = NOR flash-like, asynchronous and synchronous devices 1h (R/W) = Reserved 2h (R/W) = NAND flash-like devices, stream mode 3h (R/W) = Reserved
9-8	MUXADDDATA	R/W	0h	Enables the address and data multiplexed protocol 0h (R/W) = Nonmultiplexed attached device 1h (R/W) = AAD-multiplexed protocol device 2h (R/W) = Address and data multiplexed attached device 3h (R/W) = Reserved
7-5	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.

Table 7-549. GPMC_CONFIG1_i Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	TIMEPARAGRANULARITY	R/W	0h	Signals timing latencies scalar factor (RD/WRCYCLETIME, RD/WRACCESSTIME, PAGEBURSTACCESSTIME, CSONTIME, CSRD/WROFFTIME, ADVONTIME, ADVRD/WROFFTIME, OEONTIME, OEOFFTIME, WEONTIME, WEOFFTIME, CYCLE2CYCLEDELAY, BUSTURNAROUND, TIMEOUTSTARTVALUE, WRDATAONADMUXBUS) 0h (R/W) = x1 latencies 1h (R/W) = x2 latencies
3-2	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
1-0	GPMCFCLKDIVIDER	R/W	0h	Divides the GPMC_FCLK clock 0h (R/W) = GPMC_CLK frequency = GPMC_FCLK frequency 1h (R/W) = GPMC_CLK frequency = GPMC_FCLK frequency / 2 2h (R/W) = GPMC_CLK frequency = GPMC_FCLK frequency / 3 3h (R/W) = GPMC_CLK frequency = GPMC_FCLK frequency /4

Table 7-550. Register Call Summary for GPMC_CONFIG1_i

GPMC Basic Programming Model
- GPMC Configuration in NOR Mode: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15]
- GPMC Configuration in NAND Mode: [0][1][2][3]
- Synchronous NOR Flash Timing Parameters Formulas: [0][1][2][3][4][5][6][7][8][9][10]
- GPMC Timing Parameters: [0][1][2][3][4][5][6][7][8]
- NAND Flash Interface Timing Parameters Formulas: [0]
- Set Memory Access: [0][1][2][3][4]
- Asynchronous NOR Flash Timing Parameters Formulas: [0]

GPMC Registers
- GPMC Registers: [0]
- GPMC_CONFIG1_i Register (Offset = 60h + i*30h) [reset = 0h]: [0]

GPMC Functional Description
- Page and Burst Support: [0][1]
- Synchronous Multiple (Burst) Read (4-, 8-, 16-Word16 Burst With Wraparound Capability): [0][1][2]
- WAIT Pin Monitoring Control: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16]
- Synchronous Access Description: [0][1]
- Access on Address/Address/Data-Multiplexed Devices: [0][1]
- Chip-Select Configuration for NAND Interfacing in Byte or Word Stream Mode: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17]
- GPMC_CLK and Control Signals Setup and Hold: [0]
- Access Size Adaptation and Device Width: [0]
- Access on Address/Data Multiplexed Devices: [0][1]
- Supported Devices: [0]
- Access Time (RDACCESSTIME / WRACCESSTIME): [0]
- GPMC I/O Configuration Setting: [0][1][2][3][4][5][6][7]
- GPMC Clock Configuration: [0][1]
- System Burst vs External Device Burst Support: [0][1][2]
- Read Cycle Time and Write Cycle Time (RDCYCLETIME / WRCYCLETIME): [0]
- pSRAM Access Specificities: [0]
- NAND Device-Ready Pin: [0][1]
- NOR Access Description: [0][1]
- Address/Data-Multiplexing Interface: [0]
- GPMC_CLK: [0][1]
- Interconnect Port Interface: [0][1]

GPMC Environment
- GPMC Signals: [0]

7.3.7.12 GPMC_CONFIG2_i Register (Offset = 64h + i*30h) [reset = 00101001h]

GPMC_CONFIG2_i (where i = 0 to 3) is shown in Figure 7-220 and described in Table 7-552.

CS signal timing parameter configuration

Table 7-551. GPMC_CONFIG2_i Instances

Instance	Physical Address
GPMC	2181 8064h + (i*30h)

Figure 7-220. GPMC_CONFIG2_i Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED			CSWROFFTIME				
R/W-0h			R/W-10h				
15	14	13	12	11	10	9	8
RESERVED			CSRDOFFTIME				
R/W-0h			R/W-10h				
7	6	5	4	3	2	1	0
CSEXTRADELAY	RESERVED			CSONTIME			
R/W-0h	R/W-0h			R/W-1h			

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-552. GPMC_CONFIG2_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-21	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
20-16	CSWROFFTIME	R/W	10h	CS i deassertion time from start cycle time for write accesses 00h: 0 GPMC_FCLK cycle 01h: 1 GPMC_FCLK cycle ... 1Fh: 31 GPMC_FCLK cycles
15-13	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
12-8	CSRDOFFTIME	R/W	10h	CS i de-assertion time from start cycle time for read accesses 00h: 0 GPMC_FCLK cycle 01h: 1 GPMC_FCLK cycle ... 1Fh: 31 GPMC_FCLK cycles
7	CSEXTRADELAY	R/W	0h	CS i Add extra half-GPMC_FCLK cycle 0h (R/W) = CS i Timing control signal is not delayed 1h (R/W) = CS i Timing control signal is delayed of half GPMC_FCLK clock cycle
6-4	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
3-0	CSONTIME	R/W	1h	CS i assertion time from start cycle time 0h: 0 GPMC_FCLK cycle 1h: 1 GPMC_FCLK cycle ... Fh: 15 GPMC_FCLK cycles

Table 7-553. Register Call Summary for GPMC_CONFIG2_i

GPMC Basic Programming Model • GPMC Timing Parameters: [0][1][2][3]
GPMC Registers • GPMC_CONFIG2_i Register (Offset = 64h + i*30h) [reset = 00101001h]: [0] • GPMC Registers: [0]
GPMC Functional Description • nCS: Chip-Select Signal Control Assertion/Deassertion Time (CSONTIME / CSRDOFFTIME / CSWROFFTIME / CSEXTRADELAY): [0][1][2][3] • Synchronous Multiple (Burst) Write: [0][1] • Synchronous Single Read: [0][1] • Access on Address/Data Multiplexed Devices: [0][1]

7.3.7.13 GPMC_CONFIG3_i Register (Offset = 68h + i*30h) [reset = 22060514h]

GPMC_CONFIG3_i (where i = 0 to 3) is shown in Figure 7-221 and described in Table 7-555.

nADV signal timing parameter configuration

Table 7-554. GPMC_CONFIG3_i Instances

Instance	Physical Address
GPMC	2181 8068h + (i*30h)

Figure 7-221. GPMC_CONFIG3_i Register

31	30	29	28	27	26	25	24
RESERVED	ADVAADMUXWROFFTIME			RESERVED	ADVAADMUXRDOFFTIME		
R/W-0h	R/W-2h			R/W-0h	R/W-2h		
23	22	21	20	19	18	17	16
RESERVED			ADVWROFFTIME				
R/W-0h			R/W-6h				
15	14	13	12	11	10	9	8
RESERVED			ADVRDOFFTIME				
R/W-0h			R/W-5h				
7	6	5	4	3	2	1	0
ADVEXTRADELAY	ADVAADMUXONTIME			ADVONTIME			
R/W-0h	R/W-1h			R/W-4h			

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-555. GPMC_CONFIG3_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
30-28	ADVAADMUXWROFFTIME	R/W	2h	nADV deassertion for first address phase when using the AAD-multiplexed protocol 0h: 0 GPMC_FCLK cycle 1h: 1 GPMC_FCLK cycle ... 7h: 7 GPMC_FCLK cycles
27	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
26-24	ADVAADMUXRDOFFTIME	R/W	2h	nADV assertion for first address phase when using the AAD-multiplexed protocol 0h: 0 GPMC_FCLK cycle 1h: 1 GPMC_FCLK cycle ... 7h: 7 GPMC_FCLK cycles
23-21	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
20-16	ADVWROFFTIME	R/W	6h	nADV deassertion time from start cycle time for write accesses 00h: 0 GPMC_FCLK cycle 01h: 1 GPMC_FCLK cycle ... 1Fh: 31 GPMC_FCLK cycles
15-13	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.

Table 7-555. GPMC_CONFIG3_i Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12-8	ADVRDOFFTIME	R/W	5h	nADV deassertion time from start cycle time for read accesses 00h: 0 GPMC_FCLK cycle 01h: 1 GPMC_FCLK cycle ... 1Fh: 31 GPMC_FCLK cycles
7	ADVEXTRADELAY	R/W	0h	nADV add extra half-GPMC_FCLK cycle 0h (R/W) = nADV timing control signal is not delayed 1h (R/W) = nADV timing control signal is delayed of half GPMC_FCLK clock cycle
6-4	ADVAADMUXONTIME	R/W	1h	nADV assertion for first address phase when using the AAD-multiplexed protocol 0h: 0 GPMC_FCLK cycle 1h: 1 GPMC_FCLK cycle ... 7h: 7 GPMC_FCLK cycles
3-0	ADVONTIME	R/W	4h	nADV assertion time from start cycle time 0h: 0 GPMC_FCLK cycle 1h: 1 GPMC_FCLK cycle ... Fh: 15 GPMC_FCLK cycles

Table 7-556. Register Call Summary for GPMC_CONFIG3_i

GPMC Basic Programming Model • GPMC Timing Parameters: [0][1][2][3][4][5][6]
GPMC Registers • GPMC Registers: [0] • GPMC_CONFIG3_i Register (Offset = 68h + i*30h) [reset = 22060514h]: [0]
GPMC Functional Description • Access on Address/Data Multiplexed Devices: [0][1] • Synchronous Multiple (Burst) Write: [0][1][2][3][4][5] • Access on Address/Address/Data-Multiplexed Devices: [0][1][2][3][4][5][6][7][8][9] • nADV/ALE: Address Valid/Address Latch Enable Signal Control Assertion/Deassertion Time (ADVONTIME / ADVRDOFFTIME / ADVWROFFTIME / ADVEXTRADELAY/ADVAADMUXONTIME/ADVAADMUXRDOFFTIME/ADVAADMUXWROFFTIME): [0][1][2][3][4][5][6] • Synchronous Single Read: [0][1][2][3][4][5][6]

7.3.7.14 GPMC_CONFIG4_i Register (Offset = 6Ch + i*30h) [reset = 10057016h]

GPMC_CONFIG4_i (where i = 0 to 3) is shown in Figure 7-222 and described in Table 7-558.

nWE and nOE signals timing parameter configuration

Table 7-557. GPMC_CONFIG4_i Instances

Instance	Physical Address
GPMC	2181 806Ch + (i*30h)

Figure 7-222. GPMC_CONFIG4_i Register

31	30	29	28	27	26	25	24
RESERVED				WEOFFTIME			
R/W-0h				R/W-10h			
23	22	21	20	19	18	17	16
WEEXTRADELAY	RESERVED			WEONTIME			
R/W-0h	R/W-0h			R/W-5h			
15	14	13	12	11	10	9	8
OEAADMUX_OFFTIME			OEOFFTIME				
R/W-3h			R/W-10h				
7	6	5	4	3	2	1	0
OEEXTRADELAY	OEAADMUX_ONTIME			OEONTIME			
R/W-0h	R/W-1h			R/W-6h			

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-558. GPMC_CONFIG4_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
28-24	WEOFFTIME	R/W	10h	nWE deassertion time from start cycle time 00h: 0 GPMC_FCLK cycle 01h: 1 GPMC_FCLK cycle ... 1Fh: 31 GPMC_FCLK cycles
23	WEEXTRADELAY	R/W	0h	nWE add extra half-GPMC_FCLK cycle 0h (R/W) = nWE timing control signal is not delayed 1h (R/W) = nWE timing control signal is delayed of half-GPMC_FCLK clock cycle
22-20	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
19-16	WEONTIME	R/W	5h	nWE assertion time from start cycle time 0h: 0 GPMC_FCLK cycle 1h: 1 GPMC_FCLK cycle ... Fh: 15 GPMC_FCLK cycles
15-13	OEAADMUX_OFFTIME	R/W	3h	nOE deassertion time for the first address phase in an AAD-multiplexed access 0h: 0 GPMC_FCLK cycle 1h: 1 GPMC_FCLK cycle ... 7h: 7 GPMC_FCLK cycles

Table 7-558. GPMC_CONFIG4_i Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12-8	OEOFFTIME	R/W	10h	nOE deassertion time from start cycle time 00h: 0 GPMC_FCLK cycle 01h: 1 GPMC_FCLK cycle ... 1Fh: 31 GPMC_FCLK cycles
7	OEEXTRADELAY	R/W	0h	nOE add extra half-GPMC_FCLK cycle 0h (R/W) = nOE timing control signal is not delayed 1h (R/W) = nOE timing control signal is delayed of half-GPMC_FCLK clock cycle
6-4	OEAADMUX_ONTIME	R/W	1h	nOE assertion time for the first address phase in an AAD-mux access 0h: 0 GPMC_FCLK cycle 1h: 1 GPMC_FCLK cycle ... 7h: 7 GPMC_FCLK cycles
3-0	OEONTIME	R/W	6h	nOE assertion time from start cycle time 0h: 0 GPMC_FCLK cycle 1h: 1 GPMC_FCLK cycle ... Fh: 15 GPMC_FCLK cycles

Table 7-559. Register Call Summary for GPMC_CONFIG4_i

GPMC Basic Programming Model • GPMC Timing Parameters: [0][1][2][3][4][5][6][7]
GPMC Registers • GPMC Registers: [0] • GPMC_CONFIG4_i Register (Offset = 6Ch + i*30h) [reset = 10057016h]: [0]
GPMC Functional Description • Access on Address/Data Multiplexed Devices: [0][1][2][3] • Synchronous Single Read: [0][1][2][3][4] • Synchronous Multiple (Burst) Write: [0][1][2][3][4][5] • nWE: Write Enable Signal Control Assertion/Deassertion Time (WEONTIME / WEOFFTIME / WEEXTRADELAY): [0][1][2] • Access on Address/Address/Data-Multiplexed Devices: [0][1][2][3] • Chip-Select Configuration for NAND Interfacing in Byte or Word Stream Mode: [0] • nOE/nRE: Output Enable/Read Enable Signal Control Assertion/Deassertion Time (OEONTIME / OEOFFTIME / OEEXTRADELAY / OEAADMUXONTIME / OEAADMUXOFFTIME): [0][1][2]

7.3.7.15 GPMC_CONFIG5_i Register (Offset = 70h + i*30h) [reset = 010F1111h]

GPMC_CONFIG5_i (where i = 0 to 3) is shown in Figure 7-223 and described in Table 7-561.

RdAccessTime and CycleTime timing parameters configuration

Table 7-560. GPMC_CONFIG5_i Instances

Instance	Physical Address
GPMC	2181 8070h + (i*30h)

Figure 7-223. GPMC_CONFIG5_i Register

31	30	29	28	27	26	25	24
RESERVED				PAGEBURSTACCESSTIME			
R/W-0h				R/W-1h			
23	22	21	20	19	18	17	16
RESERVED			RDACCESSTIME				
R/W-0h			R/W-Fh				
15	14	13	12	11	10	9	8
RESERVED			WRCYCLETIME				
R/W-0h			R/W-11h				
7	6	5	4	3	2	1	0
RESERVED			RDCYCLETIME				
R/W-0h			R/W-11h				

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-561. GPMC_CONFIG5_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
27-24	PAGEBURSTACCESSTIME	R/W	1h	Delay between successive words in a multiple access 0h: 0 GPMC_FCLK cycle 1h: 1 GPMC_FCLK cycle ... Fh: 15 GPMC_FCLK cycles
23-21	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
20-16	RDACCESSTIME	R/W	Fh	Delay between start cycle time and first data valid 00h: 0 GPMC_FCLK cycle 01h: 1 GPMC_FCLK cycle ... 1Fh: 31 GPMC_FCLK cycles
15-13	RESERVED	R/W	0h	Write 0s for future compatibility. Reads returns 0
12-8	WRCYCLETIME	R/W	11h	Total write cycle time 00h: 0 GPMC_FCLK cycle 01h: 1 GPMC_FCLK cycle ... 1Fh: 31 GPMC_FCLK cycles
7-5	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.

Table 7-561. GPMC_CONFIG5_i Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	RDCYCLETIME	R/W	11h	Total read cycle time 00h: 0 GPMC_FCLK cycle 01h: 1 GPMC_FCLK cycle ... 1Fh: 31 GPMC_FCLK cycles

Table 7-562. Register Call Summary for GPMC_CONFIG5_i

GPMC Basic Programming Model • GPMC Timing Parameters: [0][1][2][3]
GPMC Registers • GPMC Registers: [0] • GPMC_CONFIG5_i Register (Offset = 70h + i*30h) [reset = 010F1111h]: [0]
GPMC Functional Description • Access on Address/Data Multiplexed Devices: [0][1][2] • Synchronous Multiple (Burst) Read (4-, 8-, 16-Word16 Burst With Wraparound Capability): [0][1][2] • Access Time (RDACCESSTIME / WRACCESSTIME): [0] • Synchronous Single Read: [0][1] • Synchronous Multiple (Burst) Write: [0][1][2][3] • Read Cycle Time and Write Cycle Time (RDCYCLETIME / WRCYCLETIME): [0][1] • Access Time on Read Access: [0][1] • Page Burst Access Time (PAGEBURSTACCESSTIME): [0] • Asynchronous Multiple (Page Mode) Read Operation on non-multiplexed Device: [0][1][2][3]

7.3.7.16 GPMC_CONFIG6_i Register (Offset = 74h + i*30h) [reset = 8F070000h]

GPMC_CONFIG6_i (where i = 0 to 3) is shown in Figure 7-224 and described in Table 7-564.

WrAccessTime, WrDataOnADmuxBus, Cycle2Cycle and BusTurnAround parameters configuration

Table 7-563. GPMC_CONFIG6_i Instances

Instance	Physical Address
GPMC	2181 8074h + (i*30h)

Figure 7-224. GPMC_CONFIG6_i Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED			WRACCESSTIME			
R/W-1h	R/W-0h			R/W-Fh			
23	22	21	20	19	18	17	16
RESERVED				WRDATAONADMUXBUS			
R/W-0h				R/W-7h			
15	14	13	12	11	10	9	8
RESERVED				CYCLE2CYCLEDELAY			
R/W-0h				R/W-0h			
7	6	5	4	3	2	1	0
CYCLE2CYCLESAMECSEN	CYCLE2CYCLEDIFFCSEN	RESERVED		BUSTURNAROUND			
R/W-0h	R/W-0h	R/W-0h		R/W-0h			

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-564. GPMC_CONFIG6_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	1h	TI Internal use - Do not modify.
30-29	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
28-24	WRACCESSTIME	R/W	Fh	Delay from start access time to the GPMC_FCLK rising edge corresponding the GPMC_CLK rising edge used by the attached memory for the first data capture 00h: 0 GPMC_FCLK cycle 01h: 1 GPMC_FCLK cycle ... 1Fh: 31 GPMC_FCLK cycles
23-20	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
19-16	WRDATAONADMUXBUS	R/W	7h	Specifies on which GPMC_FCLK rising edge the first data of the write is driven in the add/data mux bus
15-12	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
11-8	CYCLE2CYCLEDELAY	R/W	0h	Chip-select high pulse delay between successive accesses 0h: 0 GPMC_FCLK cycle 1h: 1 GPMC_FCLK cycle ... Fh: 15 GPMC_FCLK cycles
7	CYCLE2CYCLESAMECSEN	R/W	0h	Add CYCLE2CYCLEDELAY between successive accesses to the same chip-select (any access type) 0h (R/W) = No delay between the two accesses 1h (R/W) = Add CYCLE2CYCLEDELAY

Table 7-564. GPMC_CONFIG6_i Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	CYCLE2CYCLEDIFFCSEN	R/W	0h	Add CYCLE2CYCLEDELAY between successive accesses to a different chip-select (any access type) 0h (R/W) = No delay between the two accesses 1h (R/W) = Add CYCLE2CYCLEDELAY
5-4	RESERVED	R/W	0h	Write 0s for future compatibility. Reads return 0.
3-0	BUSTURNAROUND	R/W	0h	Bus turnaround latency between successive accesses to the same chip-select (read to write) or to a different chip-select (read to read and read to write) 0h: 0 GPMC_FCLK cycle 1h: 1 GPMC_FCLK cycle ... Fh: 15 GPMC_FCLK cycles

Table 7-565. Register Call Summary for GPMC_CONFIG6_i

GPMC Basic Programming Model • GPMC Timing Parameters: [0][1][2][3][4][5]
GPMC Registers • GPMC Registers: [0] • GPMC_CONFIG6_i Register (Offset = 74h + i*30h) [reset = 8F070000h]: [0]
GPMC Functional Description • Access Time on Write Access: [0] • Access Time (RDACCESSTIME / WRACCESSTIME): [0] • WAIT Pin Monitoring Control: [0][1][2][3][4][5] • Access on Address/Data Multiplexed Devices: [0][1][2] • Read Cycle Time and Write Cycle Time (RDCYCLETIME / WRCYCLETIME): [0][1] • Synchronous Multiple (Burst) Write: [0][1][2] • Access on Address/Address/Data-Multiplexed Devices: [0] • Synchronous Single Write: [0]

7.3.7.17 GPMC_CONFIG7_i Register (Offset = 78h + i*30h) [reset = F40h]

GPMC_CONFIG7_i (where i = 0 to 3) is shown in Figure 7-225 and described in Table 7-567.

CS address mapping configuration

Table 7-566. GPMC_CONFIG7_i Instances

Instance	Physical Address
GPMC	2181 8078h + (i*30h)

Figure 7-225. GPMC_CONFIG7_i Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED				MASKADDRESS			
R/W-0h				R/W-Fh			
7	6	5	4	3	2	1	0
RESERVED	CSVALID	BASEADDRESS					
R/W-0h	R/W-1h	R/W-0h					

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-567. GPMC_CONFIG7_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
11-8	MASKADDRESS	R/W	Fh	CS mask address. 0000h: Chip-select size of 256MB 1000h: Chip-select size of 128MB 1100h: Chip-select size of 64MB 1110h: Chip-select size of 32MB 1111h: Chip-select size of 16MB Other values must be avoided as they create holes in the chip-select address space.
7	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0.
6	CSVALID	R/W	1h	CS enable **NOTE: Reset value is 1h for CS0 and 0h for CS1 to CS3** 0h (R/W) = CS disabled 1h (R/W) = CS enabled
5-0	BASEADDRESS	R/W	0h	CSi base address where i = 0 to 3 (16-MB minimum granularity) bits [5-0] corresponds to A29, A28, A27, A26, A25, and A24. See Figure 7-164.

Table 7-568. Register Call Summary for GPMC_CONFIG7_i

GPMC Basic Programming Model
- GPMC Configuration in NOR Mode: [0][1][2]
- GPMC Timing Parameters: [0]
- GPMC Configuration in NAND Mode: [0][1][2]

Table 7-568. Register Call Summary for GPMC_CONFIG7_i (continued)

GPMC Registers • GPMC Registers: [0] • GPMC_CONFIG7_i Register (Offset = 78h + i*30h) [reset = F40h]: [0]
GPMC Functional Description • Chip-Select Base Address and Region Size: [0][1][2] • Chip-Select Configuration for NAND Interfacing in Byte or Word Stream Mode: [0]

7.3.7.18 GPMC_NAND_COMMAND_i Register (Offset = 7Ch + i*30h) [reset = -h]

GPMC_NAND_COMMAND_i (where i = 0 to 3) is shown in Figure 7-226 and described in Table 7-570.

This register is not a true register, only an address location.

Table 7-569. GPMC_NAND_COMMAND_i Instances

Instance	Physical Address
GPMC	2181 807Ch + (i*30h)

Figure 7-226. GPMC_NAND_COMMAND_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPMC_NAND_COMMAND																															
W-																															

LEGEND: W = Write Only; *-n* = value after reset

Table 7-570. GPMC_NAND_COMMAND_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPMC_NAND_COMMAND	W	-h	This register is not a true register, only an address location. Writing data at the GPMC_NAND_COMMAND_i location places the data as the NAND command value on the bus, using a regular asynchronous write access.

Table 7-571. Register Call Summary for GPMC_NAND_COMMAND_i

GPMC Registers • GPMC Registers: [0] • GPMC_NAND_COMMAND_i Register (Offset = 7Ch + i*30h) [reset = -h]: [0][1]
GPMC Functional Description • Command Latch Cycle: [0] • NAND Device Command and Address Phase Control: [0][1][2][3][4] • General Facts About the Engine Configuration: [0]

7.3.7.19 GPMC_NAND_ADDRESS_i Register (Offset = 80h + i*30h) [reset = -h]

GPMC_NAND_ADDRESS_i (where i = 0 to 3) is shown in Figure 7-227 and described in Table 7-573.

This register is not a true register, only an address location.

Table 7-572. GPMC_NAND_ADDRESS_i Instances

Instance	Physical Address
GPMC	2181 8080h + (i*30h)

Figure 7-227. GPMC_NAND_ADDRESS_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPMC_NAND_ADDRESS																															
W-																															

LEGEND: W = Write Only; *-n* = value after reset

Table 7-573. GPMC_NAND_ADDRESS_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPMC_NAND_ADDRESS	W	-h	This register is not a true register, only an address location. Writing data at the GPMC_NAND_ADDRESS_i location places the data as the NAND partial address value on the bus, using a regular asynchronous write access.

Table 7-574. Register Call Summary for GPMC_NAND_ADDRESS_i

GPMC Registers • GPMC Registers: [0] • GPMC_NAND_ADDRESS_i Register (Offset = 80h + i*30h) [reset = -h]: [0][1]
GPMC Functional Description • Address Latch Cycle: [0] • NAND Device Command and Address Phase Control: [0][1][2][3][4] • General Facts About the Engine Configuration: [0]

7.3.7.20 GPMC_NAND_DATA_i Register (Offset = 84h + i*30h) [reset = -h]

GPMC_NAND_DATA_i (where i = 0 to 3) is shown in Figure 7-228 and described in Table 7-576.

This register is not a true register,only an address location.

Table 7-575. GPMC_NAND_DATA_i Instances

Instance	Physical Address
GPMC	2181 8084h + (i*30h)

Figure 7-228. GPMC_NAND_DATA_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPMC_NAND_DATA																															
R/W-																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-576. GPMC_NAND_DATA_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPMC_NAND_DATA	R/W	-h	This register is not a true register, only an address location. Reading data from the GPMC_NAND_DATA_i location or from any location in the associated chip-select memory region activates an asynchronous read access.

Table 7-577. Register Call Summary for GPMC_NAND_DATA_i

GPMC Registers • GPMC Registers: [0][1][2] • GPMC_NAND_DATA_i Register (Offset = 84h + i*30h) [reset = -h]: [0][1]
GPMC Functional Description • Optimizing NAND Access Using the Prefetch and Write-Posting Engine: [0] • General Facts About the Engine Configuration: [0] • NAND Device Data Read and Write Phase Control in Stream Mode: [0][1][2][3]

7.3.7.21 GPMC_PREFETCH_CONFIG1 Register (Offset = 1E0h) [reset = 4000h]

GPMC_PREFETCH_CONFIG1 is shown in Figure 7-229 and described in Table 7-579.

Prefetch engine configuration 1

Table 7-578. GPMC_PREFETCH_CONFIG1 Instances

Instance	Physical Address
GPMC	2181 81E0h

Figure 7-229. GPMC_PREFETCH_CONFIG1 Register

31	30	29	28	27	26	25	24
RESERVED	CYCLEOPTIMIZATION			ENABLEOPTIMIZEDACCESS	ENGINECSSELECTOR		
R/W-0h	R/W-0h			R/W-0h	R/W-0h		
23	22	21	20	19	18	17	16
PFPWENROUNDROBIN	RESERVED			PFPWWEIGHTEDPRIO			
R/W-0h	R/W-0h			R/W-0h			
15	14	13	12	11	10	9	8
RESERVED	FIFOTHRESHOLD						
R/W-0h	R/W-40h						
7	6	5	4	3	2	1	0
ENABLEENGINE	RESERVED	WAITPINSELECTOR		SYNCHROMODE	DMAMODE	RESERVED	ACCESSMODE
R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-579. GPMC_PREFETCH_CONFIG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0.
30-28	CYCLEOPTIMIZATION	R/W	0h	Define the number of GPMC_FCLK cycles to be subtracted from RDCYCLETIME, WRCYCLETIME, RDACCESSTIME, CSRDOFFTIME, CSWROFFTIME, ADVRDOFFTIME, ADVWROFFTIME, OEOFFTIME, WEOFFTIME 0h: 0 GPMC_FCLK cycle 1h: 1 GPMC_FCLK cycle ... 7h: 7 GPMC_FCLK cycles
27	ENABLEOPTIMIZEDACCESS	R/W	0h	Enables access cycle optimization 0h (R/W) = Access cycle optimization is disabled. 1h (R/W) = Access cycle optimization is enabled.
26-24	ENGINECSSELECTOR	R/W	0h	Selects the chip-select where Prefetch Postwrite engine is active 0h: CS0 1h: CS1 2h: CS2 3h: CS3 4h-7h: Reserved
23	PFPWENROUNDROBIN	R/W	0h	Enables the PFPW RoundRobin arbitration 0h (R/W) = Prefetch Postwrite engine round robin arbitration is disabled. 1h (R/W) = Prefetch Postwrite engine round robin arbitration is enabled.

Table 7-579. GPMC_PREFETCH_CONFIG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22-20	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
19-16	PFPWWEIGHTEDPRIO	R/W	0h	When an arbitration occurs between a DMA and a PFPW engine access, the DMA is always serviced. If the PFPWEnRoundRobin is enabled, 0h: The next access is granted to the PFPW engine. 1h: The next two accesses are granted to the PFPW engine. ..., Fh: The next 16 accesses are granted to the PFPW engine.
15	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0.
14-8	FIFOTHRESHOLD	R/W	40h	Selects the maximum number of bytes read from the FIFO or written to the FIFO by the host on a DMA or interrupt request 00h: 0 byte 01h: 1 byte ... 40h: 64 bytes
7	ENABLEENGINE	R/W	0h	Enables the Prefetch Postwite engine 0h (R/W) = Prefetch Postwrite engine is disabled. 1h (R/W) = Prefetch Postwrite engine is enabled.
6	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0.
5-4	WAITPINSELECTOR	R/W	0h	Select which wait pin edge detector should start the engine in synchronized mode 0h (R/W) = Selects Wait0 EdgeDetection 1h (R/W) = Selects Wait1 EdgeDetection 2h, 3h: Reserved
3	SYNCHROMODE	R/W	0h	Selects when the engine starts the access to chip-select 0h (R/W) = Engine starts the access to chip-select as soon as STARTENGINE is set 1h (R/W) = Engine starts the access to chip-select as soon as STARTENGINE is set AND wait to nonwait edge detection on the selected wait pin
2	DMAMODE	R/W	0h	Selects interrupt synchronization or DMA request synchronization 0h (R/W) = Interrupt synchronization is enabled. Only interrupt line is activated on FIFO threshold crossing. 1h (R/W) = DMA request synchronization is enabled. A DMA request protocol is used.
1	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0.
0	ACCESSMODE	R/W	0h	Selects prefetch read or write-posting accesses 0h (R/W) = Prefetch read mode 1h (R/W) = Write-posting mode

Table 7-580. Register Call Summary for GPMC_PREFETCH_CONFIG1

GPMC Basic Programming Model • GPMC Configuration in NAND Mode: [0][1][2][3][4][5][6][7][8][9]
GPMC Registers • GPMC Registers: [0] • GPMC_PREFETCH_CONFIG1 Register (Offset = 1E0h) [reset = 4000h]: [0]

Table 7-580. Register Call Summary for GPMC_PREFETCH_CONFIG1 (continued)

GPMC Functional Description • Optimizing NAND Access Using the Prefetch and Write-Posting Engine: [0][1][2] • GPMC Interrupt Requests: [0] • FIFO Control in Write-Posting Mode: [0] • General Facts About the Engine Configuration: [0][1] • Prefetch and Write-Posting Engine: [0][1] • Interleaved Accesses Between Prefetch and Write-Posting Engine and Other Chip-Selects: [0][1] • Prefetch Mode: [0][1][2][3][4][5][6][7][8][9][10][11] • Write-Posting Mode: [0][1][2][3][4][5][6][7][8] • FIFO Control in Prefetch Mode: [0]

7.3.7.22 GPMC_PREFETCH_CONFIG2 Register (Offset = 1E4h) [reset = 0h]

GPMC_PREFETCH_CONFIG2 is shown in Figure 7-230 and described in Table 7-582.

Prefetch engine configuration 2

Table 7-581. GPMC_PREFETCH_CONFIG2 Instances

Instance	Physical Address
GPMC	2181 81E4h

Figure 7-230. GPMC_PREFETCH_CONFIG2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																		TRANSFERCOUNT													
R/W-0h																		R/W-0h													

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-582. GPMC_PREFETCH_CONFIG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
13-0	TRANSFERCOUNT	R/W	0h	Selects the number of bytes to be read or written by the engine to the selected chip-select 0000h: 0 byte 0001h: 1 byte ... 2000h: 8K bytes

Table 7-583. Register Call Summary for GPMC_PREFETCH_CONFIG2

GPMC Basic Programming Model • GPMC Configuration in NAND Mode: [0]
GPMC Registers • GPMC Registers: [0] • GPMC_PREFETCH_CONFIG2 Register (Offset = 1E4h) [reset = 0h]: [0]
GPMC Functional Description • Write-Posting Mode: [0][1] • Prefetch Mode: [0][1]

7.3.7.23 GPMC_PREFETCH_CONTROL Register (Offset = 1ECh) [reset = 0h]

GPMC_PREFETCH_CONTROL is shown in Figure 7-231 and described in Table 7-585.

Prefetch engine control

Table 7-584. GPMC_PREFETCH_CONTROL Instances

Instance	Physical Address
GPMC	2181 81ECh

Figure 7-231. GPMC_PREFETCH_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							STARTENGINE
R/W-0h							R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-585. GPMC_PREFETCH_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
0	STARTENGINE	R/W	0h	Resets the FIFO pointer and starts the engine Write: 0h = Stops the engine. 1h = Resets the FIFO pointer to 0h in prefetch mode and 40h in postwrite mode and starts the engine. Read: 0h = Engine is stopped. 1h = Engine is running.

Table 7-586. Register Call Summary for GPMC_PREFETCH_CONTROL

GPMC Basic Programming Model • GPMC Configuration in NAND Mode: [0][1]
GPMC Registers • GPMC Registers: [0] • GPMC_PREFETCH_CONTROL Register (Offset = 1ECh) [reset = 0h]: [0]
GPMC Functional Description • Write-Posting Mode: [0][1][2] • General Facts About the Engine Configuration: [0] • Prefetch Mode: [0][1][2]

7.3.7.24 GPMC_PREFETCH_STATUS Register (Offset = 1F0h) [reset = 0h]

GPMC_PREFETCH_STATUS is shown in Figure 7-232 and described in Table 7-588.

Prefetch engine status

Table 7-587. GPMC_PREFETCH_STATUS Instances

Instance	Physical Address
GPMC	2181 81F0h

Figure 7-232. GPMC_PREFETCH_STATUS Register

31	30	29	28	27	26	25	24
RESERVED	FIFOPOINTER						
R/W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							FIFOTHRESHOLDSTATUS
R/W-0h							R-0h
15	14	13	12	11	10	9	8
RESERVED		COUNTVALUE					
R/W-0h		R-0h					
7	6	5	4	3	2	1	0
COUNTVALUE							
R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 7-588. GPMC_PREFETCH_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0.
30-24	FIFOPOINTER	R	0h	Number of available bytes to be read or number of free empty byte places to be written 00h: 0 byte available to be read or 0 free empty place to be written ... 40h: 64 bytes available to be read or 64 empty places to be written
23-17	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
16	FIFOTHRESHOLDSTATUS	R	0h	Set when FIFOPointer exceeds FIFOThreshold value 0h (R) = FIFOPointer smaller or equal to FIFOThreshold. Writing to this bit has no effect. 1h (R) = FIFOPointer greater than FIFOThreshold. Writing to this bit has no effect.
15-14	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
13-0	COUNTVALUE	R	0h	Number of remaining bytes to be read or to be written by the engine according to the TransferCount value 0000h: 0 byte remaining to be read or to be written 0001h: 1 byte remaining to be read or to be written ... 2000h: 8KB remaining to be read or to be written

Table 7-589. Register Call Summary for GPMC_PREFETCH_STATUS

GPMC Registers • GPMC Registers: [0] • GPMC_IRQSTATUS Register (Offset = 18h) [reset = 0h]: [0][1] • GPMC_PREFETCH_STATUS Register (Offset = 1F0h) [reset = 0h]: [0]
GPMC Functional Description • FIFO Control in Prefetch Mode: [0][1][2] • FIFO Control in Write-Posting Mode: [0][1] • General Facts About the Engine Configuration: [0][1]

7.3.7.25 GPMC_ECC_CONFIG Register (Offset = 1F4h) [reset = 1030h]

GPMC_ECC_CONFIG is shown in Figure 7-233 and described in Table 7-591.

ECC configuration

Table 7-590. GPMC_ECC_CONFIG Instances

Instance	Physical Address
GPMC	2181 81F4h

Figure 7-233. GPMC_ECC_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							ECCALGORITHM
R/W-0h							R/W-0h
15	14	13	12	11	10	9	8
RESERVED		ECCBCHTSEL		ECCWRAPMODE			
R/W-0h		R/W-1h		R/W-0h			
7	6	5	4	3	2	1	0
ECC16B	ECCTOPSECTOR			ECCCS			ECCENABLE
R/W-0h	R/W-3h			R/W-0h			R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-591. GPMC_ECC_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
16	ECCALGORITHM	R/W	0h	ECC algorithm used 0h (R/W) = Hamming code 1h (R/W) = BCH code
15-14	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
13-12	ECCBCHTSEL	R/W	1h	Error correction capability used for BCH 0h (R/W) = Up to 4 bits error correction (t = 4) 1h (R/W) = Up to 8 bits error correction (t = 8) 2h (R/W) = Up to 16 bits error correction (t = 16) 3h (R/W) = Reserved
11-8	ECCWRAPMODE	R/W	0h	Spare area organization definition for the BCH algorithm. See the BCH syndrome/parity calculator module functional specification for more details
7	ECC16B	R/W	0h	Selects an ECC calculated on 16 columns 0h (R/W) = ECC calculated on 8 columns 1h (R/W) = ECC calculated on 16 columns
6-4	ECCTOPSECTOR	R/W	3h	Number of sectors to process with the BCH algorithm 0h: 1 sector (512-KB page) 1h: 2 sectors ... 3h: 4 sectors (2-KB page) ... 7h: 8 sectors (4-KB page)

Table 7-591. GPMC_ECC_CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-1	ECCCS	R/W	0h	Selects the CS where ECC is computed 0h (R/W) = CS0 1h (R/W) = CS1 2h (R/W) = CS2 3h (R/W) = CS3 Other: Reserved
0	ECCENABLE	R/W	0h	Enables the ECC feature 0h (R/W) = ECC disabled 1h (R/W) = ECC enabled

Table 7-592. Register Call Summary for GPMC_ECC_CONFIG

GPMC Basic Programming Model • GPMC Configuration in NAND Mode: [0][1][2][3][4][5]
GPMC Registers • GPMC Registers: [0] • GPMC_ECC_CONTROL Register (Offset = 1F8h) [reset = 0h]: [0][1] • GPMC_ECC_CONFIG Register (Offset = 1F4h) [reset = 1030h]: [0] • GPMC_BCH_SWDATA Register (Offset = 2D0h) [reset = 0h]: [0]
GPMC Functional Description • ECC Calculator: [0][1] • Hamming Code: [0][1][2][3][4][5]

7.3.7.26 GPMC_ECC_CONTROL Register (Offset = 1F8h) [reset = 0h]

GPMC_ECC_CONTROL is shown in Figure 7-234 and described in Table 7-594.

ECC control

Table 7-593. GPMC_ECC_CONTROL Instances

Instance	Physical Address
GPMC	2181 81F8h

Figure 7-234. GPMC_ECC_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							ECCCLEAR
R/W-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED				ECCPOINTER			
R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-594. GPMC_ECC_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
8	ECCCLEAR	R/W	0h	Clear all ECC result registers Reads return 0. Write 1h to this field clears all ECC result registers. Write 0h is ignored.
7-4	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
3-0	ECCPOINTER	R/W	0h	Selects ECC result register (Reads to this field give the dynamic position of the ECC pointer - Writes to this field select the ECC result register where the first ECC computation will be stored.); Writing other values disables the ECC engine (ECCENABLE bit of GPMC_ECC_CONFIG set to 0) 0h (R/W) = Writing 0h disables the ECC engine (ECCENABLE bit of GPMC_ECC_CONFIG set to 0) 1h (R/W) = ECC result register 1 selected 2h (R/W) = ECC result register 2 selected 3h (R/W) = ECC result register 3 selected 4h (R/W) = ECC result register 4 selected 5h (R/W) = ECC result register 5 selected 6h (R/W) = ECC result register 6 selected 7h (R/W) = ECC result register 7 selected 8h (R/W) = ECC result register 8 selected 9h (R/W) = ECC result register 9 selected

Table 7-595. Register Call Summary for GPMC_ECC_CONTROL

GPMC Basic Programming Model • GPMC Configuration in NAND Mode: [0][1]
GPMC Registers • GPMC Registers: [0] • GPMC_ECC_CONTROL Register (Offset = 1F8h) [reset = 0h]: [0]
GPMC Functional Description • Hamming Code: [0][1][2][3]

7.3.7.27 GPMC_ECC_SIZE_CONFIG Register (Offset = 1FCh) [reset = FFFFF000h]

GPMC_ECC_SIZE_CONFIG is shown in Figure 7-235 and described in Table 7-597.

ECC size

Table 7-596. GPMC_ECC_SIZE_CONFIG Instances

Instance	Physical Address
GPMC	2181 81FCh

Figure 7-235. GPMC_ECC_SIZE_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED		ECCSIZE1					
R/W-3h		R/W-FFh					
23	22	21	20	19	18	17	16
ECCSIZE1		RESERVED		ECCSIZE0			
R/W-FFh		R/W-3h		R/W-FFh			
15	14	13	12	11	10	9	8
ECCSIZE0				RESERVED			ECC9RESULTSIZE
R/W-FFh				R/W-0h			R/W-0h
7	6	5	4	3	2	1	0
ECC8RESULTSIZE	ECC7RESULTSIZE	ECC6RESULTSIZE	ECC5RESULTSIZE	ECC4RESULTSIZE	ECC3RESULTSIZE	ECC2RESULTSIZE	ECC1RESULTSIZE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-597. GPMC_ECC_SIZE_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R/W	3h	Write 0s for future compatibility. Read returns 3.
29-22	ECCSIZE1	R/W	FFh	Defines Hamming code ECC size 1 in bytes 00h: 2 bytes 01h: 4 bytes 02h: 6 bytes 03h: 8 bytes ... FFh: 512 bytes For BCH code ECC, the size 1 is programmed directly with the number of nibbles. For details, see Section 7.3.4.12.3.2.2.3, *Wrapping Modes*.
21-20	RESERVED	R/W	3h	Write 0s for future compatibility. Read returns 3.
19-12	ECCSIZE0	R/W	FFh	Defines Hamming code ECC size 0 in bytes 00h: 2 bytes 01h: 4 bytes 02h: 6 bytes 03h: 8 bytes ... FFh: 512 bytes For BCH code ECC, the size 0 is programmed directly with the number of nibbles. For details, see Section 7.3.4.12.3.2.2.3, *Wrapping Modes*.
11-9	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.

Table 7-597. GPMC_ECC_SIZE_CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	ECC9RESULTSIZE	R/W	0h	Selects ECC size for ECC 9 result register 0h (R/W) = ECCSIZE0 selected 1h (R/W) = ECCSIZE1 selected
7	ECC8RESULTSIZE	R/W	0h	Selects ECC size for ECC 8 result register 0h (R/W) = ECCSIZE0 selected 1h (R/W) = ECCSIZE1 selected
6	ECC7RESULTSIZE	R/W	0h	Selects ECC size for ECC 7 result register 0h (R/W) = ECCSIZE0 selected 1h (R/W) = ECCSIZE1 selected
5	ECC6RESULTSIZE	R/W	0h	Selects ECC size for ECC 6 result register 0h (R/W) = ECCSIZE0 selected 1h (R/W) = ECCSIZE1 selected
4	ECC5RESULTSIZE	R/W	0h	Selects ECC size for ECC 5 result register 0h (R/W) = ECCSIZE0 selected 1h (R/W) = ECCSIZE1 selected
3	ECC4RESULTSIZE	R/W	0h	Selects ECC size for ECC 4 result register 0h (R/W) = ECCSIZE0 selected 1h (R/W) = ECCSIZE1 selected
2	ECC3RESULTSIZE	R/W	0h	Selects ECC size for ECC 3 result register 0h (R/W) = ECCSIZE0 selected 1h (R/W) = ECCSIZE1 selected
1	ECC2RESULTSIZE	R/W	0h	Selects ECC size for ECC 2 result register 0h (R/W) = ECCSIZE0 selected 1h (R/W) = ECCSIZE1 selected
0	ECC1RESULTSIZE	R/W	0h	Selects ECC size for ECC 1 result register 0h (R/W) = ECCSIZE0 selected 1h (R/W) = ECCSIZE1 selected

Table 7-598. Register Call Summary for GPMC_ECC_SIZE_CONFIG

GPMC Basic Programming Model • GPMC Configuration in NAND Mode: [0][1]
GPMC Registers • GPMC Registers: [0] • GPMC_ECC_SIZE_CONFIG Register (Offset = 1FCh) [reset = FFFFF000h]: [0]
GPMC Functional Description • Hamming Code: [0][1][2][3] • BCH Code: [0][1][2]

7.3.7.28 GPMC_ECCj_RESULT Register (Offset = 200h + j*4h) [reset = 0h]

GPMC_ECCj_RESULT (where j = 0 to 8) is shown in Figure 7-236 and described in Table 7-600.

ECC result register

Table 7-599. GPMC_ECCj_RESULT Instances

Instance	Physical Address
GPMC	2181 8200h + (j*4h)

Figure 7-236. GPMC_ECCj_RESULT Register

31	30	29	28	27	26	25	24
RESERVED				P2048O	P1024O	P512O	P256O
R/W-0h				R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
P128O	P64O	P32O	P16O	P8O	P4O	P2O	P1O
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED				P2048E	P1024E	P512E	P256E
R/W-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
P128E	P64E	P32E	P16E	P8E	P4E	P2E	P1E
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 7-600. GPMC_ECCj_RESULT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
27	P2048O	R	0h	Odd row parity bit 2048, only used for ECC computed on 512 bytes
26	P1024O	R	0h	Odd row parity bit 1024
25	P512O	R	0h	Odd row parity bit 512
24	P256O	R	0h	Odd row parity bit 256
23	P128O	R	0h	Odd row parity bit 128
22	P64O	R	0h	Odd row parity bit 64
21	P32O	R	0h	Odd row parity bit 32
20	P16O	R	0h	Odd row parity bit 16
19	P8O	R	0h	Odd row parity bit 8
18	P4O	R	0h	Odd Column Parity bit 4
17	P2O	R	0h	Odd Column Parity bit 2
16	P1O	R	0h	Odd Column Parity bit 1
15-12	RESERVED	R/W	0h	Write 0s for future compatibility. Read returns 0s.
11	P2048E	R	0h	Even row parity bit 2048, only used for ECC computed on 512 bytes
10	P1024E	R	0h	Even row parity bit 1024
9	P512E	R	0h	Even row parity bit 512
8	P256E	R	0h	Even row parity bit 256
7	P128E	R	0h	Even row parity bit 128
6	P64E	R	0h	Even row parity bit 64

Table 7-600. GPMC_ECCj_RESULT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	P32E	R	0h	Even row parity bit 32
4	P16E	R	0h	Even row parity bit 16
3	P8E	R	0h	Even row parity bit 8
2	P4E	R	0h	Even column parity bit 4
1	P2E	R	0h	Even column parity bit 2
0	P1E	R	0h	Even column parity bit 1

Table 7-601. Register Call Summary for GPMC_ECCj_RESULT

GPMC Registers • GPMC Registers: [0] • GPMC_ECCj_RESULT Register (Offset = 200h + j*4h) [reset = 0h]: [0]
GPMC Functional Description • Hamming Code: [0][1][2][3][4][5][6][7]

7.3.7.29 GPMC_BCH_RESULT0_i Register (Offset = 240h + i*10h) [reset = 0h]

GPMC_BCH_RESULT0_i (where i = 0 to 3) is shown in Figure 7-237 and described in Table 7-603.

BCH ECC result (bits 0 to 31)

Table 7-602. GPMC_BCH_RESULT0_i Instances

Instance	Physical Address
GPMC	2181 8240h + (i*10h)

Figure 7-237. GPMC_BCH_RESULT0_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCH_RESULT_0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-603. GPMC_BCH_RESULT0_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BCH_RESULT_0	R/W	0h	BCH ECC result (bits 0 to 31)

Table 7-604. Register Call Summary for GPMC_BCH_RESULT0_i

GPMC Registers • GPMC Registers: [0] • GPMC_BCH_RESULT0_i Register (Offset = 240h + i*10h) [reset = 0h]: [0]
GPMC Functional Description • BCH Code: [0]

7.3.7.30 GPMC_BCH_RESULT1_i Register (Offset = 244h + i*10h) [reset = 0h]

GPMC_BCH_RESULT1_i (where i = 0 to 3) is shown in Figure 7-238 and described in Table 7-606.

BCH ECC result (bits 32 to 63)

Table 7-605. GPMC_BCH_RESULT1_i Instances

Instance	Physical Address
GPMC	2181 8244h + (i*10h)

Figure 7-238. GPMC_BCH_RESULT1_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCH_RESULT_1																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-606. GPMC_BCH_RESULT1_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BCH_RESULT_1	R/W	0h	BCH ECC result (bits 32 to 63)

Table 7-607. Register Call Summary for GPMC_BCH_RESULT1_i

GPMC Registers • GPMC Registers: [0] • GPMC_BCH_RESULT1_i Register (Offset = 244h + i*10h) [reset = 0h]: [0]
GPMC Functional Description • BCH Code: [0]

7.3.7.31 GPMC_BCH_RESULT2_i Register (Offset = 248h + i*10h) [reset = 0h]

GPMC_BCH_RESULT2_i (where i = 0 to 3) is shown in Figure 7-239 and described in Table 7-609.

BCH ECC result (bits 64 to 95)

Table 7-608. GPMC_BCH_RESULT2_i Instances

Instance	Physical Address
GPMC	2181 8248h + (i*10h)

Figure 7-239. GPMC_BCH_RESULT2_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCH_RESULT_2																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-609. GPMC_BCH_RESULT2_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BCH_RESULT_2	R/W	0h	BCH ECC result (bits 64 to 95)

Table 7-610. Register Call Summary for GPMC_BCH_RESULT2_i

GPMC Registers • GPMC Registers: [0] • GPMC_BCH_RESULT2_i Register (Offset = 248h + i*10h) [reset = 0h]: [0]
GPMC Functional Description • BCH Code: [0]

7.3.7.32 GPMC_BCH_RESULT3_i Register (Offset = 24Ch + i*10h) [reset = 0h]

GPMC_BCH_RESULT3_i (where i = 0 to 3) is shown in Figure 7-240 and described in Table 7-612.

BCH ECC result (bits 96 to 127)

Table 7-611. GPMC_BCH_RESULT3_i Instances

Instance	Physical Address
GPMC	2181 824Ch + (i*10h)

Figure 7-240. GPMC_BCH_RESULT3_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCH_RESULT_3																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-612. GPMC_BCH_RESULT3_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BCH_RESULT_3	R/W	0h	BCH ECC result (bits 96 to 127)

Table 7-613. Register Call Summary for GPMC_BCH_RESULT3_i

GPMC Registers • GPMC Registers: [0] • GPMC_BCH_RESULT3_i Register (Offset = 24Ch + i*10h) [reset = 0h]: [0]
GPMC Functional Description • BCH Code: [0]

7.3.7.33 GPMC_BCH_SWDATA Register (Offset = 2D0h) [reset = 0h]

GPMC_BCH_SWDATA is shown in Figure 7-241 and described in Table 7-615.

This register is used to directly pass data to the BCH ECC calculator without accessing the actual NAND flash interface.

Table 7-614. GPMC_BCH_SWDATA Instances

Instance	Physical Address
GPMC	2181 82D0h

Figure 7-241. GPMC_BCH_SWDATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																BCH_DATA															
R-0h																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 7-615. GPMC_BCH_SWDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Write 0s for future compatibility. Read returns 0s.
15-0	BCH_DATA	R/W	0h	Data to be included in the BCH calculation Only bits 0 to 7 are considered if the calculator is configured to use 8-bit data (GPMC_ECC_CONFIG[7] ECC16B = 0)

Table 7-616. Register Call Summary for GPMC_BCH_SWDATA

GPMC Registers
• GPMC Registers: [0] • GPMC_BCH_SWDATA Register (Offset = 2D0h) [reset = 0h]: [0]

7.3.7.34 GPMC_BCH_RESULT4_i Register (Offset = 300h + i*10h) [reset = 0h]

GPMC_BCH_RESULT4_i (where i = 0 to 3) is shown in Figure 7-242 and described in Table 7-618.

BCH ECC result (bits 128 to 159)

Table 7-617. GPMC_BCH_RESULT4_i Instances

Instance	Physical Address
GPMC	2181 8300h + (i*10h)

Figure 7-242. GPMC_BCH_RESULT4_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCH_RESULT_4																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-618. GPMC_BCH_RESULT4_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BCH_RESULT_4	R/W	0h	BCH ECC result (bits 128 to 159)

Table 7-619. Register Call Summary for GPMC_BCH_RESULT4_i

GPMC Registers
- GPMC Registers: [0]
- GPMC_BCH_RESULT4_i Register (Offset = 300h + i*10h) [reset = 0h]: [0]

7.3.7.35 GPMC_BCH_RESULT5_i Register (Offset = 304h + i*10h) [reset = 0h]

GPMC_BCH_RESULT5_i (where i = 0 to 3) is shown in Figure 7-243 and described in Table 7-621.

BCH ECC result (bits 160 to 191)

Table 7-620. GPMC_BCH_RESULT5_i Instances

Instance	Physical Address
GPMC	2181 8304h + (i*10h)

Figure 7-243. GPMC_BCH_RESULT5_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCH_RESULT_5																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-621. GPMC_BCH_RESULT5_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BCH_RESULT_5	R/W	0h	BCH ECC result (bits 160 to 191)

Table 7-622. Register Call Summary for GPMC_BCH_RESULT5_i

GPMC Registers
- GPMC Registers: [0]
- GPMC_BCH_RESULT5_i Register (Offset = 304h + i*10h) [reset = 0h]: [0]

7.3.7.36 GPMC_BCH_RESULT6_i Register (Offset = 308h + i*10h) [reset = 0h]

GPMC_BCH_RESULT6_i (where i = 0 to 3) is shown in Figure 7-244 and described in Table 7-624.

BCH ECC result (bits 192 to 207)

Table 7-623. GPMC_BCH_RESULT6_i Instances

Instance	Physical Address
GPMC	2181 8308h + (i*10h)

Figure 7-244. GPMC_BCH_RESULT6_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																BCH_RESULT_6															
R-0h																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 7-624. GPMC_BCH_RESULT6_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Write 0s for future compatibility. Read returns 0s.
15-0	BCH_RESULT_6	R/W	0h	BCH ECC result (bits 192 to 207)

Table 7-625. Register Call Summary for GPMC_BCH_RESULT6_i

GPMC Registers

- GPMC Registers: [0]
- GPMC_BCH_RESULT6_i Register (Offset = 308h + i*10h) [reset = 0h]: [0]

7.4 Error Location Module (ELM)

This section describes the Error Location Module (ELM) for the device.

NOTE: ELM is not supported on this family of devices.

7.4.1 ELM Overview

When reading from NAND flash memories, some level of error-correction is required. In the case of NAND modules with no internal correction capability, sometimes referred to as *bare NANDs*, the correction process is delegated to the memory controller.

The general-purpose memory controller (GPMC) probes data read from an external NAND flash and uses this to compute checksum-like information, called syndrome polynomials, on a per-block basis. Each syndrome polynomial gives a status of the read operations for a full block, including 512 bytes of data, parity bits, and an optional spare-area data field, with a maximum block size of 1023 bytes. Computation is based on a Bose-Chaudhuri-Hocquenghem (BCH) algorithm. The error-location module (ELM) extracts error addresses from these syndrome polynomials.

Based on the syndrome polynomial value, the ELM can detect errors, compute the number of errors, and give the location of each error bit. The actual data is not required to complete the error-correction algorithm. Errors can be reported anywhere in the NAND flash block, including in the parity bits.

The maximum acceptable number of errors that can be corrected depends on a programmable configuration parameter. 4-, 8-, and 16-bit error-correction levels are supported. The ELM depends on a static and fixed definition of the generator polynomial for each error-correction level that corresponds to the generator polynomials defined in the GPMC (there are three fixed polynomial for the three correction error levels). A larger number of errors than the programmed error-correction level may be detected, but the ELM cannot correct them all. The offending block is then tagged as *uncorrectable* in the associated computation exit status register. If the computation is successful, that is, if the number of errors detected does not exceed the maximum value authorized for the chosen correction capability, the exit status register contains the information on the number of detected errors.

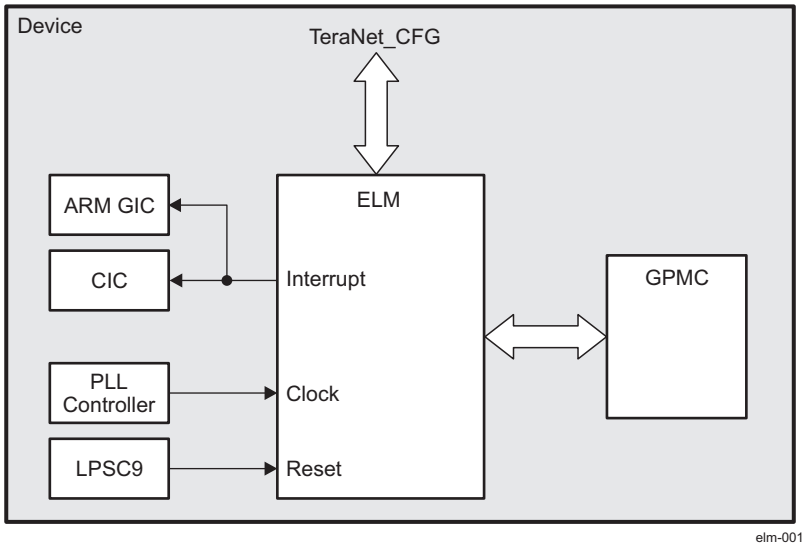
When the error-location process completes, an interrupt is triggered to inform the software that its status can be checked. The number of detected errors and their locations in the NAND block can be retrieved from the module through register accesses.

The ELM has the following features:

- 4, 8, and 16 bits per 512-byte block error-location, based on BCH algorithms
- Eight simultaneous processing contexts
- Page-based and continuous modes
- Interrupt generation on error-location process completion:
  - When the full page has been processed in page mode
  - For each syndrome polynomial in continuous mode.

Figure 7-245 shows the ELM overview.

Figure 7-245. ELM Overview



7.4.2 ELM Integration

The ELM extracts error addresses from generated syndrome polynomials.

The ELM is used with the GPMC. Syndrome polynomials generated on-the-fly when reading a NAND flash page and stored in GPMC registers are passed to the ELM. A host processor can then correct the data block by flipping the bits to which the ELM error-location outputs point.

Figure 7-246 shows the integration of the ELM module in the device.

Figure 7-246. ELM Integration

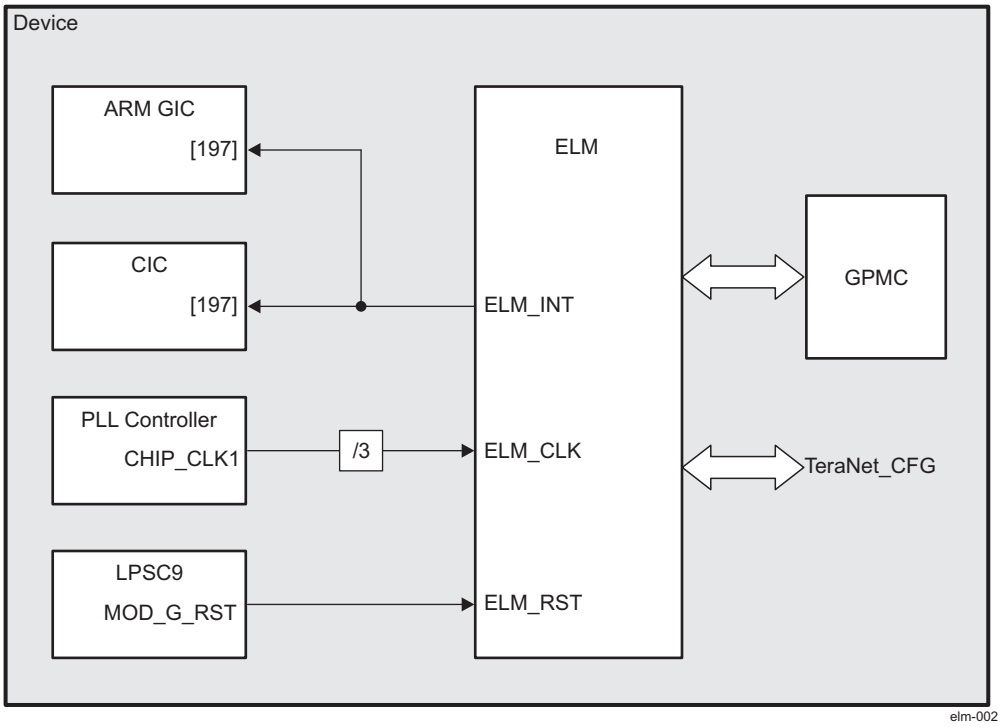


Table 7-626 through Table 7-628 summarize the integration of the ELM module in the device.

Table 7-626. ELM Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
ELM	PD5	LPSC9	TeraNet_CFG

Table 7-627. ELM Clocks and Resets

Clocks				
Module Instance	**Destination Signal Name**	**Source Signal Name**	**Source**	**Description**
ELM	ELM_CLK	CHIP_CLK1/3	PLL Controller	Functional and interface clock
Resets				
Module Instance	**Destination Signal Name**	**Source Signal Name**	**Source**	**Description**
ELM	ELM_RST	MOD_G_RST	LPSC9	Module hardware reset

Table 7-628. ELM Hardware Requests

Interrupt Requests				
Module Instance	**Event Name**	**Mapped To Input Event [Number]**		**Description**
		ARM GIC	**CIC**	
ELM	ELM_INT	[197]	[197]	Error location process complete interrupt

NOTE: For more information about interrupts, see Section 7.4.3.3.

7.4.3 ELM Functional Description

The ELM is designed around the error-location engine, which handles the computation based on the input syndrome polynomials.

The ELM maps the error-location engine to a standard interconnect interface by using a set of registers to control inputs and outputs.

7.4.3.1 ELM Software Reset

To perform a software reset, set the ELM_SYSCONFIG[1] SOFTRESET bit to 1. The ELM_SYSSTATUS[0] RESETDONE bit indicates that the software reset is complete when its value is 1. When the software reset completes, the ELM_SYSCONFIG[1] SOFTRESET bit is automatically reset.

7.4.3.2 ELM Power Management

Table 7-629 describes the power-management features available to the ELM.

Table 7-629. Local Power-Management Features

Feature	Registers	Description
Clock autogating	ELM_SYSCONFIG[0] AUTOGATING	This bit allows a local power optimization inside the module by gating the ELM_CLK clock upon the interface activity.
Idle modes	ELM_SYSCONFIG[4-3] SIDLEMODE	Force-idle, no-idle, and smart-idle modes are available.
Clock activity	ELM_SYSCONFIG[8] CLOCKACTIVITY	The clock can be switched off or maintained.

7.4.3.3 ELM Interrupt Requests

Table 7-630 lists the event flags, and their masks, that can cause module interrupts asserting the ELM_INT signal.

Table 7-630. ELM Events

Event Flag	Event Mask	Description
ELM_IRQSTATUS[8] PAGE_VALID	ELM_IRQENABLE[8] PAGE_MASK	Page interrupt
ELM_IRQSTATUS[7] LOC_VALID_7	ELM_IRQENABLE[7] LOCATION_MASK_7	Error-location interrupt for syndrome polynomial 7
ELM_IRQSTATUS[6] LOC_VALID_6	ELM_IRQENABLE[6] LOCATION_MASK_6	Error-location interrupt for syndrome polynomial 6
ELM_IRQSTATUS[5] LOC_VALID_5	ELM_IRQENABLE[5] LOCATION_MASK_5	Error-location interrupt for syndrome polynomial 5
ELM_IRQSTATUS[4] LOC_VALID_4	ELM_IRQENABLE[4] LOCATION_MASK_4	Error-location interrupt for syndrome polynomial 4
ELM_IRQSTATUS[3] LOC_VALID_3	ELM_IRQENABLE[3] LOCATION_MASK_3	Error-location interrupt for syndrome polynomial 3
ELM_IRQSTATUS[2] LOC_VALID_2	ELM_IRQENABLE[2] LOCATION_MASK_2	Error-location interrupt for syndrome polynomial 2
ELM_IRQSTATUS[1] LOC_VALID_1	ELM_IRQENABLE[1] LOCATION_MASK_1	Error-location interrupt for syndrome polynomial 1
ELM_IRQSTATUS[0] LOC_VALID_0	ELM_IRQENABLE[0] LOCATION_MASK_0	Error-location interrupt for syndrome polynomial 0

7.4.3.4 Processing Initialization

ELM_LOCATION_CONFIG global setting parameters must be set before using the error-location engine. The ELM_LOCATION_CONFIG[1-0] ECC_BCH_LEVEL bit field defines the error-correction level used (4-, 8-, or 16-bit error correction). The ELM_LOCATION_CONFIG[26-16] ECC_SIZE bit field defines the maximum buffer length beyond which the engine processing no longer looks for errors.

Software can choose to use the ELM in continuous mode or page mode. If all ELM_PAGE_CTRL[i] SECTOR_i bits (i is the syndrome polynomial number, where i = 0 to 7) are reset, continuous mode is used. In any other case, page mode is implicitly selected.

- Continuous mode: Each syndrome polynomial is processed independently. Results for a syndrome can be retrieved and acknowledged at any time, regardless of the status of the other seven processing contexts.
- Page mode: Syndrome polynomials are grouped into atomic entities: only one page can be processed at any given time, even if all eight contexts are not used for this page. Unused contexts are lost and cannot be affected to any other processing. The full page must be acknowledged and cleared before moving to the next page.

For completion interrupts to be generated correctly, all ELM_IRQENABLE[i] LOCATION_MASK_i bits (where i = 0 to 7) must be forced to 0 when in page mode, and set to 1 in continuous mode. Additionally, the ELM_IRQENABLE[8] PAGE_MASK bit must be set to 1 when in page mode.

Software initiates error-location processing by writing a syndrome polynomial into one of the eight possible register sets. Each of these register sets includes seven registers: ELM_SYNDROME_FRAGMENT_0_i to ELM_SYNDROME_FRAGMENT_6_i. The first six registers can be written in any order, but ELM_SYNDROME_FRAGMENT_6_i must be written last because it includes the validity bit, which instructs the ELM that this syndrome polynomial must be processed (the ELM_SYNDROME_FRAGMENT_6_i[16] SYNDROME_VALID bit).

As soon as one validity bit is asserted (ELM_SYNDROME_FRAGMENT_6_i[16] SYNDROME_VALID = 1h, where i = 0 to 7), error-location processing can start for the corresponding syndrome polynomial. The associated ELM_LOCATION_STATUS_i and ELM_ERROR_LOCATION_0_i to ELM_ERROR_LOCATION_15_i registers (where i = 0 to 7) are not reset. Software must not consider them until the corresponding ELM_IRQSTATUS[i] LOC_VALID_i bit is set.

7.4.3.5 Processing Sequence

While the error-location engine is busy processing one syndrome polynomial, further syndrome polynomials can be written. They are processed when the current processing completes.

The engine completes early when:

- No error is detected; that is, when the ELM_LOCATION_STATUS_i[8] ECC_CORRECTABLE bit is set to 1 and the ELM_LOCATION_STATUS_i[4-0] ECC_NB_ERRORS bit field is set to 0h.
- Too many errors are detected; that is, when the ELM_LOCATION_STATUS_i[8] ECC_CORRECTABLE bit is set to 0 while the ELM_LOCATION_STATUS_i[4-0] ECC_NB_ERRORS bit field is set with the value output by the error-location engine. The reported number of errors is not ensured if ECC_CORRECTABLE is 0.

If the engine completes early, the associated error-location registers ELM_ERROR_LOCATION_0_i to ELM_ERROR_LOCATION_15_i (where i = 0 to 7) are not updated.

In all other cases, the engine goes through the entire error-location process. Each time an error location is found, it is logged in the associated ECC_ERROR_LOCATION bit field. The first error detected is logged in the ELM_ERROR_LOCATION_0_i[12-0] ECC_ERROR_LOCATION bit field; the second is logged in the ELM_ERROR_LOCATION_1_i[12-0] ECC_ERROR_LOCATION bit field, and so on.

Table 7-631 describes the ELM_LOCATION_STATUS_i value decoding.

Table 7-631. ELM_LOCATION_STATUS_i Value Decoding

ECC_CORRECTABLE Value	ECC_NB_ERRORS Value	Status	Number of Errors Detected	Action Required
1	0	OK	0	None
1	≠ 0	OK	ECC_NB_ERRORS	Correct the data buffer read based on the ELM_ERROR_LOCATION_0_i to ELM_ERROR_LOCATION_15_i results.
0	Any	Failed	Unknown	Software-dependent

7.4.3.6 Processing Completion

When the processing for a given syndrome polynomial completes, its ELM_SYNDROME_FRAGMENT_6_i[16] SYNDROME_VALID bit is reset. It must not be set again until the exit status registers, ELM_LOCATION_STATUS_i (where i = 0 to 7) for this processing are checked. Failure to comply with this rule leads to potential loss of the first polynomial process data output.

The error-location engine signals the process completion to the ELM. When this event is detected, the corresponding ELM_IRQSTATUS[i] LOC_VALID_i bit (where i = 0 to 7) is set. The processing exit status is available from the associated ELM_LOCATION_STATUS_i register, and error locations are stored in order in the ECC_ERROR_LOCATION bit fields. Software must read only valid error-location registers based on the number of errors detected and located.

Immediately after the error-location engine completes, a new syndrome polynomial can be processed, if any is available, as reported by the ELM_SYNDROME_FRAGMENT_6_i[16] SYNDROME_VALID bit, depending on the configured error-correction level. If several syndrome polynomials are available, a round-robin arbitration is used to select one for processing.

In continuous mode (that is, all bits in ELM_PAGE_CTRL are reset), an interrupt is triggered whenever a ELM_IRQSTATUS[i] LOC_VALID_i bit is asserted. Software must read the ELM_IRQSTATUS register to determine which polynomial is processed and retrieve the exit status and error locations (ELM_LOCATION_STATUS_i and ELM_ERROR_LOCATION_0_i to ELM_ERROR_LOCATION_15_i). When done, software must clear the corresponding ELM_IRQSTATUS[i] LOC_VALID_i bit by setting it to 1. Other status bits must be set to 0 so that other interrupts are not unintentionally cleared. When using this mode, the ELM_IRQSTATUS[8] PAGE_VALID interrupt is never triggered.

In page mode, the module does not trigger interrupts for the processing completion of each polynomial because the ELM_IRQENABLE[i] LOCATION_MASK_i bits are cleared. A page is defined using the ELM_PAGE_CTRL register. Each SECTOR_i bit set means the corresponding polynomial i is part of the page processing. A page is fully processed when all tagged polynomials have been processed, as logged in the ELM_IRQSTATUS[i] LOC_VALID_i bits. The module triggers an ELM_IRQSTATUS[8] PAGE_VALID interrupt whenever it detects that the full page has been processed. To make sure the next page can be correctly processed, all status bits in the ELM_IRQSTATUS register must be cleared by using a single atomic-write access.

NOTE: Do not modify page setting parameters in the ELM_PAGE_CTRL register unless the engine is idle, no polynomial input is valid, and all interrupts have been cleared.

Because no polynomial-level interrupt is triggered in page mode, polynomials cleared in the ELM_PAGE_CTRL[i] SECTOR_i bits (where i = 0 to 7) are processed as usual, but are essentially ignored. Software must manually poll the ELM_IRQSTATUS bits to check for their status.

7.4.4 ELM Basic Programming Model

7.4.4.1 ELM Low-Level Programming Model

7.4.4.1.1 Processing Initialization

Table 7-632. ELM Processing Initialization

Step	Register/Bit Field/Programming Model	Value
Resets the module.	ELM_SYSCONFIG[1] SOFTRESET	1h
Wait until reset is done.	ELM_SYSSTATUS[0] RESETDONE	1h
Configure the slave interface power management.	ELM_SYSCONFIG[4-3] SIDLEMODE	Set value
Defines the error-correction level used.	ELM_LOCATION_CONFIG[1-0] ECC_BCH_LEVEL	Set value
Defines the maximum buffer length.	ELM_LOCATION_CONFIG[26-16] ECC_SIZE	Set value
Sets the ELM in continuous mode or page mode.	ELM_PAGE_CTRL	Set value
IF continuous mode is used:	All ELM_PAGE_CTRL[i] SECTOR_i (where i = 0 to 7)	0h
Enable interrupt for syndrome polynomial i.	ELM_IRQENABLE[i] LOCATION_MASK_i	1h
ELSE (page mode is used):	One syndrome polynomial i is set ELM_PAGE_CTRL[i] SECTOR_i (where i = 0 to 7)	1h
Disable all interrupts for syndrome polynomial and enable PAGE_MASK interrupt.	All ELM_IRQENABLE[i] LOCATION_MASK_i = 0h and ELM_IRQENABLE[8] PAGE_MASK = 1h	Set value
ENDIF		Set value
Set the input syndrome polynomial i.	ELM_SYNDROME_FRAGMENT_0_i	Set value
	ELM_SYNDROME_FRAGMENT_1_i	Set value
	ELM_SYNDROME_FRAGMENT_5_i	Set value
	ELM_SYNDROME_FRAGMENT_6_i	Set value
Initiates the computation process.	ELM_SYNDROME_FRAGMENT_6_i[16] SYNDROME_VALID	1h

7.4.4.1.2 Read Results

The engine goes through the entire error-location process and results can be read. Table 7-633 and Table 7-634 describe the processing completion for continuous and page modes, respectively.

Table 7-633. ELM Processing Completion for Continuous Mode

Step	Register/Bit Field/Programming Model	Value
Wait until process is complete for syndrome polynomial i: Wait until the ELM_INT interrupt is generated, or poll the status register.		
Read for which i the error-location process is complete.	ELM_IRQSTATUS[i] LOC_VALID_i	1h
IF the process fails (too many errors):	ELM_LOCATION_STATUS_i[8] ECC_CORRECTABLE	0h
It is software dependant.		
ELSE (process successful, the engine completes):	ELM_LOCATION_STATUS_i[8] ECC_CORRECTABLE	1h
Read the number of errors.	ELM_LOCATION_STATUS_i[4-0] ECC_NB_ERRORS	
Read the error-location bit addresses for syndrome polynomial i of the ECC_NB_ERRORS first registers. Software must correct errors in the data buffer.	ELM_ERROR_LOCATION_0_i[12-0] ECC_ERROR_LOCATION	
	ELM_ERROR_LOCATION_1_i[12-0] ECC_ERROR_LOCATION	
	...	
	ELM_ERROR_LOCATION_15_i[12-0] ECC_ERROR_LOCATION	
ENDIF		

Table 7-633. ELM Processing Completion for Continuous Mode (continued)

Step	Register/Bit Field/Programming Model	Value
Clear the corresponding i interrupt.	ELM_IRQSTATUS[i] LOC_VALID_i	1h

A new syndrome polynomial can be processed after the end of processing (ELM_SYNDROME_FRAGMENT_6_i[16] SYNDROME_VALID = 0h) and after the exit status register check (ELM_LOCATION_STATUS_i).

Table 7-634. ELM Processing Completion for Page Mode

Step	Register/Bit Field/Programming Model	Value
Wait until process is complete for syndrome polynomial i: Wait until the ELM_INT interrupt is generated, or poll the status register.		
Wait for page completed interrupt: All error locations are valid.	ELM_IRQSTATUS[8] PAGE_VALID	1h
Repeat the following actions the necessary number of times. That is, once for each valid defined block in the page.		
Read the process exit status.	ELM_LOCATION_STATUS_i[8] ECC_CORRECTABLE	
IF the process fails (too many errors):	ELM_LOCATION_STATUS_i[8] ECC_CORRECTABLE	0h
It is software dependent.		
ELSE (process successful, the engine completes):	ELM_LOCATION_STATUS_i[8] ECC_CORRECTABLE	1h
Read the number of errors.	ELM_LOCATION_STATUS_i[4-0] ECC_NB_ERRORS	
Read the error-location bit addresses for syndrome polynomial i of the ECC_NB_ERRORS first registers.	ELM_ERROR_LOCATION_0_i[12-0] ECC_ERROR_LOCATION	
	ELM_ERROR_LOCATION_1_i[12-0] ECC_ERROR_LOCATION	
	...	
	ELM_ERROR_LOCATION_15_i[12-0] ECC_ERROR_LOCATION	
ENDIF		
End Repeat		
Clear the ELM_IRQSTATUS register.	ELM_IRQSTATUS	1FFh

Next page can be correctly processed after a page is fully processed, when all tagged polynomials have been processed (ELM_IRQSTATUS[i] LOC_VALID_i = 1h for all syndrome polynomials i used in the page).

7.4.4.2 Use Case: ELM Used in Continuous Mode

In this example, the ELM is programmed for an 8-bit error-correction capability in continuous mode (see Table 7-635). After reading a 528-byte NAND flash sector (512B data plus 16B spare area) with a 16-bit interface, a nonzero polynomial syndrome is reported from the GPMC (polynomial syndrome 0 is used in the ELM):

- P = 0A16ABE115E44F767BFB0D0980h.

Table 7-635. Use Case: Continuous Mode

Step	Register/Bit Field/Programming Model	Value
Reset the module.	ELM_SYSCONFIG[1] SOFTRESET	1h
Wait until reset is done.	ELM_SYSSTATUS[0] RESETDONE	1h
Configure the slave interface power management: Smart idle is used.	ELM_SYSCONFIG[4-3] SIDLEMODE	2h
Define the error-correction level used: 8 bits.	ELM_LOCATION_CONFIG[1-0] ECC_BCH_LEVEL	1h
Define the maximum buffer length: 528 bytes (2 × 528 = 1056).	ELM_LOCATION_CONFIG[26-16] ECC_SIZE	420h

Table 7-635. Use Case: Continuous Mode (continued)

Step	Register/Bit Field/Programming Model	Value
Set the ELM in continuous mode.	ELM_PAGE_CTRL	0
Enable interrupt for syndrome polynomial 0.	ELM_IRQENABLE[0] LOCATION_MASK_0	1h
Set the input syndrome polynomial 0.	ELM_SYNDROME_FRAGMENT_0_i (where i = 0)	FB0D0980h
	ELM_SYNDROME_FRAGMENT_1_i (where i = 0)	E44F767Bh
	ELM_SYNDROME_FRAGMENT_2_i (where i = 0)	16ABE115h
	ELM_SYNDROME_FRAGMENT_3_i (where i = 0)	0000000Ah
Initiate the computation process.	ELM_SYNDROME_FRAGMENT_6_i[16] SYNDROME_VALID (where i = 0)	1h
Wait until process is complete for syndrome polynomial 0: ELM_INT is generated or poll the status register.		
Read that error-location process is complete for syndrome polynomial 0.	ELM_IRQSTATUS[0] LOC_VALID_0	1h
Read the process exit status: All errors were successfully located.	ELM_LOCATION_STATUS_i[8] ECC_CORRECTABLE (where i = 0)	1h
Read the number of errors: Four errors detected.	ELM_LOCATION_STATUS_i[4-0] ECC_NB_ERRORS (where i = 0)	4h
Read the error-location bit addresses for syndrome polynomial 0 of the first four registers: Errors are located in the data buffer at decimal addresses 431, 1062, 1909, 3452.	ELM_ERROR_LOCATION_0_i (where i = 0)	1AFh
	ELM_ERROR_LOCATION_1_i (where i = 0)	426h
	ELM_ERROR_LOCATION_2_i (where i = 0)	775h
	ELM_ERROR_LOCATION_3_i (where i = 0)	D7Ch
Clear the corresponding interrupt for polynomial 0.	ELM_IRQSTATUS[0] LOC_VALID_0	1h

The NAND flash data in the sector are seen as a polynomial of degree 4223 (number of bits in a 528 byte buffer minus 1), with each data bit being a coefficient in the polynomial. When reading from a NAND flash using the GPMC module, computation of the polynomial syndrome assumes that the first NAND word read at address 0h contains the highest-order coefficient in the message. Furthermore, in the 16-bit NAND word, bits are ordered from bit 7 to bit 0, and then from bit 15 to bit 8. Based on this convention, an address table of the data buffer can be built. NAND memory addresses in Table 7-636 are given in decimal format.

Table 7-636. 16-Bit NAND Sector Buffer Address Map

NAND Memory Address	Message Bit Addresses in Memory Word															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	4215	4214	4213	4212	4211	4210	4209	4208	4223	4222	4221	4220	4219	4218	4217	4216
1	4175	4174	4173	4172	4171	4170	4169	4168	4183	4182	4181	4180	4179	4178	4177	4176
...																
47	3463	3462	3461	3460	3459	3458	3457	3456	3471	3470	3469	3468	3467	3466	3465	3464
48	3447	3446	3445	3444	3443	3442	3441	3440	3455	3454	3453	3452	3451	3450	3449	3448
49	3431	3430	3429	3428	3427	3426	3425	3424	3439	3438	3437	3436	3435	3434	3433	3432
50	3415	3414	3413	3412	3411	3410	3409	3408	3423	3422	3421	3420	3419	3418	3417	3416
...																
255	135	134	133	132	131	130	129	128	143	142	141	140	139	138	137	136
256	119	118	117	116	115	114	113	112	127	126	125	124	123	122	121	120
257	103	102	101	100	99	98	97	96	111	110	109	108	107	106	105	104
258	87	86	85	84	83	82	81	80	95	94	93	92	91	90	89	88
259	71	70	69	68	67	66	65	64	79	78	77	76	75	74	73	72
260	55	54	53	52	51	50	49	48	63	62	61	60	59	58	57	56
261	39	38	37	36	35	34	33	32	47	46	45	44	43	42	41	40
262	23	22	21	20	19	18	17	16	31	30	29	28	27	26	25	24

Table 7-636. 16-Bit NAND Sector Buffer Address Map (continued)

263	7	6	5	4	3	2	1	0	15	14	13	12	11	10	9	8

The table can now be used to determine which bits in the buffer were incorrect and must be flipped. In this example, the first bit to be flipped is bit 4 from the 49th byte read from memory. It is up to the processor to correctly map this word to the copied buffer and flip this bit. The same process must be repeated for all detected errors.

7.4.4.3 Use Case: ELM Used in Page Mode

In this example, the ELM module is programmed for a 16-bit error-correction capability in page mode (see Table 7-637). After reading a 528-byte NAND flash sector (512B data plus 16B spare area) with a 16-bit interface, four non-zero polynomial syndromes are reported from the GPMC (polynomial syndrome 0, 1, 2, and 3 are used in the ELM):

- P0 = E8B0 12ADDB5A318E05BE B0693DB28330B5CC A329AA05E0B718EFh
- P1 = BAD0 49A0D932C22E6669 0948DF08BE093336 79C6BA10E5F935EBh
- P2 = 69D9 B86ABCD5EC3697FA A6498FEE54556EA0 1579EF7D60BA3189h
- P3 = 0h

Table 7-637. Use Case: Page Mode

Step	Register/Bit Field/Programming Model	Value
Reset the module	ELM_SYSCONFIG[1] SOFTRESET	1h
Wait until reset is done.	ELM_SYSSTATUS[0] RESETDONE	1h
Configure the slave interface power management: Smart idle is used.	ELM_SYSCONFIG[4-3] SIDLEMODE	2h
Define the error-correction level used: 16 bits	ELM_LOCATION_CONFIG[1-0] ECC_BCH_LEVEL	2h
Define the maximum buffer length: 528 bytes	ELM_LOCATION_CONFIG[26-16] ECC_SIZE	420h
Set the ELM in page mode (four blocks in a page)	ELM_PAGE_CTRL[0] SECTOR_0	1h
	ELM_PAGE_CTRL[1] SECTOR_1	1h
	ELM_PAGE_CTRL[2] SECTOR_2	1h
	ELM_PAGE_CTRL[3] SECTOR_3	1h
Disable all interrupts for syndrome polynomial and enable PAGE_MASK interrupt.	ELM_IRQENABLE	100h
Set the input syndrome polynomial 0.	ELM_SYNDROME_FRAGMENT_0_i (where i = 0)	E0B718EFh
	ELM_SYNDROME_FRAGMENT_1_i (where i = 0)	A329AA05h
	ELM_SYNDROME_FRAGMENT_2_i (where i = 0)	8330B5CCh
	ELM_SYNDROME_FRAGMENT_3_i (where i = 0)	B0693DB2h
	ELM_SYNDROME_FRAGMENT_4_i (where i = 0)	318E05BEh
	ELM_SYNDROME_FRAGMENT_5_i (where i = 0)	12ADDB5Ah
	ELM_SYNDROME_FRAGMENT_6_i (where i = 0)	E8B0h
Set the input syndrome polynomial 1.	ELM_SYNDROME_FRAGMENT_0_i (where i = 1)	E5F935EBh
	ELM_SYNDROME_FRAGMENT_1_i (where i = 1)	79C6BA10h
	ELM_SYNDROME_FRAGMENT_2_i ((where i = 1)	BE093336h
	ELM_SYNDROME_FRAGMENT_3_i (where i = 1)	0948DF08h
	ELM_SYNDROME_FRAGMENT_4_i (where i = 1)	C22E6669h
	ELM_SYNDROME_FRAGMENT_5_i (where i = 1)	49A0D932h
	ELM_SYNDROME_FRAGMENT_6_i (where i = 1)	BAD0h

Table 7-637. Use Case: Page Mode (continued)

Step	Register/Bit Field/Programming Model	Value
Set the input syndrome polynomial 2.	ELM_SYNDROME_FRAGMENT_0_i (where i = 2)	60BA3189h
	ELM_SYNDROME_FRAGMENT_1_i (where i = 2)	1579EF7Dh
	ELM_SYNDROME_FRAGMENT_2_i (where i = 2)	54556EA0h
	ELM_SYNDROME_FRAGMENT_3_i (where i = 2)	A6498FEEh
	ELM_SYNDROME_FRAGMENT_4_i (where i = 2)	EC3697FAh
	ELM_SYNDROME_FRAGMENT_5_i (where i = 2)	B86ABCD5h
	ELM_SYNDROME_FRAGMENT_6_i (where i = 2)	69D9h
Set the input syndrome polynomial 3.	ELM_SYNDROME_FRAGMENT_0_i (where i = 3)	0h
	ELM_SYNDROME_FRAGMENT_1_i (where i = 3)	0h
	ELM_SYNDROME_FRAGMENT_2_i (where i = 3)	0h
	ELM_SYNDROME_FRAGMENT_3_i (where i = 3)	0h
	ELM_SYNDROME_FRAGMENT_4_i (where i = 3)	0h
	ELM_SYNDROME_FRAGMENT_5_i (where i = 3)	0h
	ELM_SYNDROME_FRAGMENT_6_i (where i = 3)	0h
Initiate the computation process for syndrome polynomial 0	ELM_SYNDROME_FRAGMENT_6_i[16] SYNDROME_VALID (where i = 0)	1h
Initiate the computation process for syndrome polynomial 1	ELM_SYNDROME_FRAGMENT_6_i[16] SYNDROME_VALID (where i = 1)	1h
Initiate the computation process for syndrome polynomial 2	ELM_SYNDROME_FRAGMENT_6_i[16] SYNDROME_VALID (where i = 2)	1h
Initiate the computation process for syndrome polynomial 3	ELM_SYNDROME_FRAGMENT_6_i[16] SYNDROME_VALID (where i = 3)	1h
Wait until process is complete for syndrome polynomial 0, 1, 2, and 3: Wait until the ELM_INT interrupt is generated or poll the status register.		
Wait for page completed interrupt: All error locations are valid.	ELM_IRQSTATUS[8] PAGE_VALID	1h
Read the process exit status for syndrome polynomial 0: All errors were successfully located.	ELM_LOCATION_STATUS_i[8] ECC_CORRECTABLE (where i = 0)	1h
Read the process exit status for syndrome polynomial 1: All errors were successfully located.	ELM_LOCATION_STATUS_i[8] ECC_CORRECTABLE (where i = 1)	1h
Read the process exit status for syndrome polynomial 2: All errors were successfully located.	ELM_LOCATION_STATUS_i[8] ECC_CORRECTABLE (where i = 2)	1h
Read the process exit status for syndrome polynomial 3: All errors were successfully located.	ELM_LOCATION_STATUS_i[8] ECC_CORRECTABLE (where i = 3)	1h
Read the number of errors for syndrome polynomial 0: four errors detected.	ELM_LOCATION_STATUS_i[4-0] ECC_NB_ERRORS (where i = 0)	4h
Read the number of errors for syndrome polynomial 1: two errors detected.	ELM_LOCATION_STATUS_i[4-0] ECC_NB_ERRORS (where i = 1)	2h
Read the number of errors for syndrome polynomial 2: one error detected.	ELM_LOCATION_STATUS_i[4-0] ECC_NB_ERRORS (where i = 2)	1h
Read the number of errors for syndrome polynomial 3: no errors detected.	ELM_LOCATION_STATUS_i[4-0] ECC_NB_ERRORS (where i = 3)	0h

Table 7-637. Use Case: Page Mode (continued)

Step	Register/Bit Field/Programming Model	Value
Read the error-location bit addresses for syndrome polynomial 0 of the first four registers:	ELM_ERROR_LOCATION_0_i (where i = 0)	1FEh
	ELM_ERROR_LOCATION_1_i (where i = 0)	617h
	ELM_ERROR_LOCATION_2_i (where i = 0)	650h
	ELM_ERROR_LOCATION_3_i (where i = 0)	A83h
Read the error-location bit addresses for syndrome polynomial 1 of the first two registers:	ELM_ERROR_LOCATION_0_i (where i = 1)	4h
	ELM_ERROR_LOCATION_1_i (where i = 1)	1036h
Read the errors location bit addresses for syndrome polynomial 2 of the first registers:	ELM_ERROR_LOCATION_0_i (where i = 1)	3E8h
Clear the ELM_IRQSTATUS register.	ELM_IRQSTATUS	1FFh

7.4.5 ELM Registers

Table 7-639 lists the memory-mapped registers for the ELM. All register offset addresses not listed in Table 7-639 should be considered as reserved locations and the register contents should not be modified.

Table 7-638. ELM Instances

Instance	Base Address
ELM	021C 8000h

Table 7-639. ELM Registers

Offset	Acronym	Register Name	ELM Physical Address	Section
0h	ELM_REVISION	This register contains the IP revision code. (A write to or reset of this register has no effect.)	021C 8000h	Section 7.4.5.1
10h	ELM_SYSCONFIG	This register controls ELM local power management and software reset.	021C 8010h	Section 7.4.5.2
14h	ELM_SYSSTATUS	Internal reset monitoring	021C 8014h	Section 7.4.5.3
18h	ELM_IRQSTATUS	Interrupt status. This register doubles as a status register for the error-location processes.	021C 8018h	Section 7.4.5.4
1Ch	ELM_IRQENABLE	Interrupt enable	021C 801Ch	Section 7.4.5.5
20h	ELM_LOCATION_CONFIG	ECC algorithm parameters	021C 8020h	Section 7.4.5.6
80h	ELM_PAGE_CTRL	Page definition	021C 8080h	Section 7.4.5.7
400h + (i*40h)	ELM_SYNDROME_FRAGMENT_0_i (i = 0 to 7)	Input syndrome polynomial bits 0 to 31	021C 8400h + (i*40h)	Section 7.4.5.8
404h + (i*40h)	ELM_SYNDROME_FRAGMENT_1_i (i = 0 to 7)	Input syndrome polynomial bits 32 to 63	021C 8404h + (i*40h)	Section 7.4.5.9
408h + (i*40h)	ELM_SYNDROME_FRAGMENT_2_i (i = 0 to 7)	Input syndrome polynomial bits 64 to 95	021C 8408h + (i*40h)	Section 7.4.5.10
40Ch + (i*40h)	ELM_SYNDROME_FRAGMENT_3_i (i = 0 to 7)	Input syndrome polynomial bits 96 to 127	021C 840Ch + (i*40h)	Section 7.4.5.11
410h + (i*40h)	ELM_SYNDROME_FRAGMENT_4_i (i = 0 to 7)	Input syndrome polynomial bits 128 to 159	021C 8410h + (i*40h)	Section 7.4.5.12
414h + (i*40h)	ELM_SYNDROME_FRAGMENT_5_i (i = 0 to 7)	Input syndrome polynomial bits 160 to 191	021C 8414h + (i*40h)	Section 7.4.5.13
418h + (i*40h)	ELM_SYNDROME_FRAGMENT_6_i (i = 0 to 7)	Input syndrome polynomial bits 192 to 207	021C 8418h + (i*40h)	Section 7.4.5.14
800h + (i*100h)	ELM_LOCATION_STATUS_i (i = 0 to 7)	Exit status for the syndrome polynomial processing	021C 8800h + (i*100h)	Section 7.4.5.15
880h + (i*100h)	ELM_ERROR_LOCATION_0_i (i = 0 to 7)	Error-location register 0	021C 8880h + (i*100h)	Section 7.4.5.16
884h + (i*100h)	ELM_ERROR_LOCATION_1_i (i = 0 to 7)	Error-location register 1	021C 8884h + (i*100h)	Section 7.4.5.17
888h + (i*100h)	ELM_ERROR_LOCATION_2_i (i = 0 to 7)	Error-location register 2	021C 8888h + (i*100h)	Section 7.4.5.18
88Ch + (i*100h)	ELM_ERROR_LOCATION_3_i (i = 0 to 7)	Error-location register 3	021C 888Ch + (i*100h)	Section 7.4.5.19
890h + (i*100h)	ELM_ERROR_LOCATION_4_i (i = 0 to 7)	Error-location register 4	021C 8890h + (i*100h)	Section 7.4.5.20
894h + (i*100h)	ELM_ERROR_LOCATION_5_i (i = 0 to 7)	Error-location register 5	021C 8894h + (i*100h)	Section 7.4.5.21

Table 7-639. ELM Registers (continued)

Offset	Acronym	Register Name	ELM Physical Address	Section
898h + (i*100h)	ELM_ERROR_LOCATION_6_i (i = 0 to 7)	Error-location register 6	021C 8898h + (i*100h)	Section 7.4.5.22
89Ch + (i*100h)	ELM_ERROR_LOCATION_7_i (i = 0 to 7)	Error-location register 7	021C 889Ch + (i*100h)	Section 7.4.5.23
8A0h + (i*100h)	ELM_ERROR_LOCATION_8_i (i = 0 to 7)	Error-location register 8	021C 88A0h + (i*100h)	Section 7.4.5.24
8A4h + (i*100h)	ELM_ERROR_LOCATION_9_i (i = 0 to 7)	Error-location register 9	021C 88A4h + (i*100h)	Section 7.4.5.25
8A8h + (i*100h)	ELM_ERROR_LOCATION_10_i (i = 0 to 7)	Error-location register 10	021C 88A8h + (i*100h)	Section 7.4.5.26
8ACh + (i*100h)	ELM_ERROR_LOCATION_11_i (i = 0 to 7)	Error-location register 11	021C 88ACh + (i*100h)	Section 7.4.5.27
8B0h + (i*100h)	ELM_ERROR_LOCATION_12_i (i = 0 to 7)	Error-location register 12	021C 88B0h + (i*100h)	Section 7.4.5.28
8B4h + (i*100h)	ELM_ERROR_LOCATION_13_i (i = 0 to 7)	Error-location register 13	021C 88B4h + (i*100h)	Section 7.4.5.29
8B8h + (i*100h)	ELM_ERROR_LOCATION_14_i (i = 0 to 7)	Error-location register 14	021C 88B8h + (i*100h)	Section 7.4.5.30
8BCh + (i*100h)	ELM_ERROR_LOCATION_15_i (i = 0 to 7)	Error-location register 15	021C 88BCh + (i*100h)	Section 7.4.5.31

7.4.5.1 ELM_REVISION Register (Offset = 0h) [reset = 20h]

ELM_REVISION is shown in Figure 7-247 and described in Table 7-641.

This register contains the IP revision code.
(A write to or reset of this register has no effect.)

Table 7-640. ELM_REVISION Instances

Instance	Physical Address
ELM	021C 8000h

Figure 7-247. ELM_REVISION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REVISION																															
R-20h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 7-641. ELM_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REVISION	R	20h	TI internal data. Identifies revision of peripheral.

Table 7-642. Register Call Summary for ELM_REVISION

ELM Registers
- ELM_REVISION Register (Offset = 0h) [reset = 20h]: [0]
- ELM Registers: [0]

7.4.5.2 ELM_SYSCONFIG Register (Offset = 10h) [reset = 11h]

ELM_SYSCONFIG is shown in Figure 7-248 and described in Table 7-644.

This register controls ELM local power management and software reset.

Table 7-643. ELM_SYSCONFIG Instances

Instance	Physical Address
ELM	021C 8010h

Figure 7-248. ELM_SYSCONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							CLOCKACTIVITYOCP
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED			SIDLEMODE		RESERVED	SOFTRESET	AUTOGATING
R-0h			R/W-2h		R-0h	R/W-0h	R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 7-644. ELM_SYSCONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	CLOCKACTIVITYOCP	R/W	0h	ELM_CLK activity when module is in IDLE mode 0h (R/W) = ELM_CLK can be switched off. 1h (R/W) = ELM_CLK is maintained.
7-5	RESERVED	R	0h	Reserved
4-3	SIDLEMODE	R/W	2h	Slave interface power management (IDLE req/ack control) 0h (R/W) = Force-idle. IDLE request is acknowledged unconditionally and immediately 1h (R/W) = No-idle. IDLE request is never acknowledged. 2h (R/W) = Smart-idle. The acknowledgment to an IDLE request is given based on the internal activity. 3h (R/W) = Reserved — do not use
2	RESERVED	R	0h	Reserved
1	SOFTRESET	R/W	0h	Module software reset This bit is automatically reset by hardware (during reads, it always returns 0). It has same effect as ELM_RST. 0h (R/W) = Normal mode 1h (R/W) = Start soft reset sequence.
0	AUTOGATING	R/W	1h	Internal ELM_CLK gating strategy (no module visible effect other than saving power) 0h (R/W) = ELM_CLK is free-running. 1h (R/W) = Automatic internal ELM_CLK gating strategy is applied based on the Interconnect interface activity.

Table 7-645. Register Call Summary for ELM_SYSCONFIG

ELM Basic Programming Model • Use Case: ELM Used in Page Mode: [0][1] • Processing Initialization: [0][1] • Use Case: ELM Used in Continuous Mode: [0][1]
ELM Registers • ELM_SYSCONFIG Register (Offset = 10h) [reset = 11h]: [0] • ELM Registers: [0]
ELM Functional Description • ELM Software Reset: [0][1] • ELM Power Management: [0][1][2]

7.4.5.3 ELM_SYSSTATUS Register (Offset = 14h) [reset = 1h]

ELM_SYSSTATUS is shown in Figure 7-249 and described in Table 7-647.

Internal reset monitoring
Undefined since:
From hardware perspective, the reset state is 0.
From software user perspective, when the accessible module is 1.

Table 7-646. ELM_SYSSTATUS Instances

Instance	Physical Address
ELM	021C 8014h

Figure 7-249. ELM_SYSSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RESETDONE
R-0h							R-1h

LEGEND: R = Read Only; *-n* = value after reset

Table 7-647. ELM_SYSSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	RESETDONE	R	1h	Internal reset monitoring Undefined since: From hardware perspective, the reset state is 0. From software user perspective, when the accessible module is 1. 0h (R) = Reset is ongoing. 1h (R) = Reset is done (completed).

Table 7-648. Register Call Summary for ELM_SYSSTATUS

ELM Basic Programming Model • Use Case: ELM Used in Page Mode: [0] • Processing Initialization: [0] • Use Case: ELM Used in Continuous Mode: [0]
ELM Registers • ELM_SYSSTATUS Register (Offset = 14h) [reset = 1h]: [0] • ELM Registers: [0]
ELM Functional Description • ELM Software Reset: [0]

7.4.5.4 ELM_IRQSTATUS Register (Offset = 18h) [reset = 0h]

ELM_IRQSTATUS is shown in Figure 7-250 and described in Table 7-650.

Interrupt status. This register doubles as a status register for the error-location processes.

Table 7-649. ELM_IRQSTATUS Instances

Instance	Physical Address
ELM	021C 8018h

Figure 7-250. ELM_IRQSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							PAGE_VALID
R-0h							R/W-0h
7	6	5	4	3	2	1	0
LOC_VALID_7	LOC_VALID_6	LOC_VALID_5	LOC_VALID_4	LOC_VALID_3	LOC_VALID_2	LOC_VALID_1	LOC_VALID_0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 7-650. ELM_IRQSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	PAGE_VALID	R/W	0h	Error-location status for a full page, based on the mask definition Read 0h: Error locations invalid for all polynomials enabled in the ECC_INTERRUPT_MASK register Read 1h: All error locations valid Write 0h: No effect Write 1h: Clear interrupt
7	LOC_VALID_7	R/W	0h	Error-location status for syndrome polynomial 7 Read 0h: No syndrome processed or process in progress Read 1h: Error-location process completed Write 0h: No effect Write 1h: Clear interrupt
6	LOC_VALID_6	R/W	0h	Error-location status for syndrome polynomial 6
5	LOC_VALID_5	R/W	0h	Error-location status for syndrome polynomial 5
4	LOC_VALID_4	R/W	0h	Error-location status for syndrome polynomial 4
3	LOC_VALID_3	R/W	0h	Error-location status for syndrome polynomial 3
2	LOC_VALID_2	R/W	0h	Error-location status for syndrome polynomial 2
1	LOC_VALID_1	R/W	0h	Error-location status for syndrome polynomial 1
0	LOC_VALID_0	R/W	0h	Error-location status for syndrome polynomial 0

Table 7-651. Register Call Summary for ELM_IRQSTATUS

ELM Basic Programming Model • ELM Low-Level Programming Model: [0] • Read Results: [0][1][2][3][4] • Use Case: ELM Used in Page Mode: [0][1][2] • Use Case: ELM Used in Continuous Mode: [0][1]
ELM Registers • ELM_IRQSTATUS Register (Offset = 18h) [reset = 0h]: [0] • ELM Registers: [0]
ELM Functional Description • Processing Completion: [0][1][2][3][4][5][6][7][8] • ELM Interrupt Requests: [0][1][2][3][4][5][6][7][8] • Processing Initialization: [0]

7.4.5.5 ELM_IRQENABLE Register (Offset = 1Ch) [reset = 0h]

ELM_IRQENABLE is shown in Figure 7-251 and described in Table 7-653.

Interrupt enable.

Table 7-652. ELM_IRQENABLE Instances

Instance	Physical Address
ELM	021C 801Ch

Figure 7-251. ELM_IRQENABLE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							PAGE_MASK
R-0h							R/W-0h
7	6	5	4	3	2	1	0
LOCATION_MASK_7	LOCATION_MASK_6	LOCATION_MASK_5	LOCATION_MASK_4	LOCATION_MASK_3	LOCATION_MASK_2	LOCATION_MASK_1	LOCATION_MASK_0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 7-653. ELM_IRQENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	PAGE_MASK	R/W	0h	Page interrupt mask bit 0: Disable interrupt 1: Enable interrupt
7	LOCATION_MASK_7	R/W	0h	Error-location interrupt mask bit for syndrome polynomial 7
6	LOCATION_MASK_6	R/W	0h	Error-location interrupt mask bit for syndrome polynomial 6
5	LOCATION_MASK_5	R/W	0h	Error-location interrupt mask bit for syndrome polynomial 5
4	LOCATION_MASK_4	R/W	0h	Error-location interrupt mask bit for syndrome polynomial 4
3	LOCATION_MASK_3	R/W	0h	Error-location interrupt mask bit for syndrome polynomial 3
2	LOCATION_MASK_2	R/W	0h	Error-location interrupt mask bit for syndrome polynomial 2
1	LOCATION_MASK_1	R/W	0h	Error-location interrupt mask bit for syndrome polynomial 1
0	LOCATION_MASK_0	R/W	0h	Error-location interrupt mask bit for syndrome polynomial 0 0: Disable interrupt 1: Enable interrupt

Table 7-654. Register Call Summary for ELM_IRQENABLE

ELM Basic Programming Model

- Use Case: ELM Used in Page Mode: [0]
- Processing Initialization: [0][1][2]
- Use Case: ELM Used in Continuous Mode: [0]

Table 7-654. Register Call Summary for ELM_IRQENABLE (continued)

ELM Registers • ELM_IRQENABLE Register (Offset = 1Ch) [reset = 0h]: [0] • ELM Registers: [0]
ELM Functional Description • Processing Completion: [0] • ELM Interrupt Requests: [0][1][2][3][4][5][6][7][8] • Processing Initialization: [0][1]

7.4.5.6 ELM_LOCATION_CONFIG Register (Offset = 20h) [reset = 0h]

ELM_LOCATION_CONFIG is shown in Figure 7-252 and described in Table 7-656.

ECC algorithm parameters.

Table 7-655. ELM_LOCATION_CONFIG Instances

Instance	Physical Address
ELM	021C 8020h

Figure 7-252. ELM_LOCATION_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED						ECC_SIZE	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
ECC_SIZE							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ECC_BCH_LEVEL	
R-0h						R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 7-656. ELM_LOCATION_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	Reserved
26-16	ECC_SIZE	R/W	0h	Maximum size of the buffers for which the error-location engine is used, in number of nibbles (4-bit entities)
15-2	RESERVED	R	0h	Reserved
1-0	ECC_BCH_LEVEL	R/W	0h	Error correction level 0h: 4 bits 1h: 8 bits 2h: 16 bits 3h: Reserved

Table 7-657. Register Call Summary for ELM_LOCATION_CONFIG

ELM Basic Programming Model • Use Case: ELM Used in Page Mode: [0][1] • Processing Initialization: [0][1] • Use Case: ELM Used in Continuous Mode: [0][1]
ELM Registers • ELM_LOCATION_CONFIG Register (Offset = 20h) [reset = 0h]: [0] • ELM Registers: [0]
ELM Functional Description • Processing Initialization: [0][1][2]

7.4.5.7 ELM_PAGE_CTRL Register (Offset = 80h) [reset = 0h]

ELM_PAGE_CTRL is shown in Figure 7-253 and described in Table 7-659.

Page definition.

Table 7-658. ELM_PAGE_CTRL Instances

Instance	Physical Address
ELM	021C 8080h

Figure 7-253. ELM_PAGE_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
SECTOR_7	SECTOR_6	SECTOR_5	SECTOR_4	SECTOR_3	SECTOR_2	SECTOR_1	SECTOR_0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 7-659. ELM_PAGE_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	SECTOR_7	R/W	0h	Set to 1 if syndrome polynomial 7 is part of the page in page mode. Must be 0 in continuous mode.
6	SECTOR_6	R/W	0h	Set to 1 if syndrome polynomial 6 is part of the page in page mode. Must be 0 in continuous mode.
5	SECTOR_5	R/W	0h	Set to 1 if syndrome polynomial 5 is part of the page in page mode. Must be 0 in continuous mode.
4	SECTOR_4	R/W	0h	Set to 1 if syndrome polynomial 4 is part of the page in page mode. Must be 0 in continuous mode.
3	SECTOR_3	R/W	0h	Set to 1 if syndrome polynomial 3 is part of the page in page mode. Must be 0 in continuous mode.
2	SECTOR_2	R/W	0h	Set to 1 if syndrome polynomial 2 is part of the page in page mode. Must be 0 in continuous mode.
1	SECTOR_1	R/W	0h	Set to 1 if syndrome polynomial 1 is part of the page in page mode. Must be 0 in continuous mode.
0	SECTOR_0	R/W	0h	Set to 1 if syndrome polynomial 0 is part of the page in page mode. Must be 0 in continuous mode.

Table 7-660. Register Call Summary for ELM_PAGE_CTRL

ELM Basic Programming Model

- Use Case: ELM Used in Page Mode: [0][1][2][3]
- Processing Initialization: [0][1][2]
- Use Case: ELM Used in Continuous Mode: [0]

Table 7-660. Register Call Summary for ELM_PAGE_CTRL (continued)

ELM Registers • ELM_PAGE_CTRL Register (Offset = 80h) [reset = 0h]: [0] • ELM Registers: [0]
ELM Functional Description • Processing Completion: [0][1][2][3] • Processing Initialization: [0]

7.4.5.8 ELM_SYNDROME_FRAGMENT_0_i Register (Offset = 400h + i*40h) [reset = 0h]

ELM_SYNDROME_FRAGMENT_0_i (where i = 0 to 7) is shown in Figure 7-254 and described in Table 7-662.

Input syndrome polynomial bits 0 to 31.

Table 7-661. ELM_SYNDROME_FRAGMENT_0_i Instances

Instance	Physical Address
ELM	021C 8400h + (i*40h)

Figure 7-254. ELM_SYNDROME_FRAGMENT_0_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYNDROME_0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-662. ELM_SYNDROME_FRAGMENT_0_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SYNDROME_0	R/W	0h	Syndrome bits 0 to 31

Table 7-663. Register Call Summary for ELM_SYNDROME_FRAGMENT_0_i

ELM Basic Programming Model • Use Case: ELM Used in Page Mode: [0][1][2][3] • Processing Initialization: [0] • Use Case: ELM Used in Continuous Mode: [0]
ELM Registers • ELM_SYNDROME_FRAGMENT_0_i Register (Offset = 400h + i*40h) [reset = 0h]: [0] • ELM Registers: [0]
ELM Functional Description • Processing Initialization: [0]

7.4.5.9 ELM_SYNDROME_FRAGMENT_1_i Register (Offset = 404h + i*40h) [reset = 0h]

ELM_SYNDROME_FRAGMENT_1_i (where i = 0 to 7) is shown in Figure 7-255 and described in Table 7-665.

Input syndrome polynomial bits 32 to 63.

Table 7-664. ELM_SYNDROME_FRAGMENT_1_i Instances

Instance	Physical Address
ELM	021C 8404h + (i*40h)

Figure 7-255. ELM_SYNDROME_FRAGMENT_1_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYNDROME_1																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-665. ELM_SYNDROME_FRAGMENT_1_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SYNDROME_1	R/W	0h	Syndrome bits 32 to 63

Table 7-666. Register Call Summary for ELM_SYNDROME_FRAGMENT_1_i

ELM Basic Programming Model • Use Case: ELM Used in Page Mode: [0][1][2][3] • Processing Initialization: [0] • Use Case: ELM Used in Continuous Mode: [0]
ELM Registers • ELM_SYNDROME_FRAGMENT_1_i Register (Offset = 404h + i*40h) [reset = 0h]: [0] • ELM Registers: [0]

7.4.5.10 ELM_SYNDROME_FRAGMENT_2_i Register (Offset = 408h + i*40h) [reset = 0h]

ELM_SYNDROME_FRAGMENT_2_i (where i = 0 to 7) is shown in Figure 7-256 and described in Table 7-668.

Input syndrome polynomial bits 64 to 95.

Table 7-667. ELM_SYNDROME_FRAGMENT_2_i Instances

Instance	Physical Address
ELM	021C 8408h + (i*40h)

Figure 7-256. ELM_SYNDROME_FRAGMENT_2_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYNDROME_2																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-668. ELM_SYNDROME_FRAGMENT_2_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SYNDROME_2	R/W	0h	Syndrome bits 64 to 95

Table 7-669. Register Call Summary for ELM_SYNDROME_FRAGMENT_2_i

ELM Basic Programming Model • Use Case: ELM Used in Page Mode: [0][1][2][3] • Use Case: ELM Used in Continuous Mode: [0]
ELM Registers • ELM_SYNDROME_FRAGMENT_2_i Register (Offset = 408h + i*40h) [reset = 0h]: [0] • ELM Registers: [0]

7.4.5.11 ELM_SYNDROME_FRAGMENT_3_i Register (Offset = 40Ch + i*40h) [reset = 0h]

ELM_SYNDROME_FRAGMENT_3_i (where i = 0 to 7) is shown in Figure 7-257 and described in Table 7-671.

Input syndrome polynomial bits 96 to 127.

Table 7-670. ELM_SYNDROME_FRAGMENT_3_i Instances

Instance	Physical Address
ELM	021C 840Ch + (i*40h)

Figure 7-257. ELM_SYNDROME_FRAGMENT_3_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYNDROME_3																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-671. ELM_SYNDROME_FRAGMENT_3_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SYNDROME_3	R/W	0h	Syndrome bits 96 to 127

Table 7-672. Register Call Summary for ELM_SYNDROME_FRAGMENT_3_i

ELM Basic Programming Model • Use Case: ELM Used in Page Mode: [0][1][2][3] • Use Case: ELM Used in Continuous Mode: [0]
ELM Registers • ELM_SYNDROME_FRAGMENT_3_i Register (Offset = 40Ch + i*40h) [reset = 0h]: [0] • ELM Registers: [0]

7.4.5.12 ELM_SYNDROME_FRAGMENT_4_i Register (Offset = 410h + i*40h) [reset = 0h]

ELM_SYNDROME_FRAGMENT_4_i (where i = 0 to 7) is shown in Figure 7-258 and described in Table 7-674.

Input syndrome polynomial bits 128 to 159.

Table 7-673. ELM_SYNDROME_FRAGMENT_4_i Instances

Instance	Physical Address
ELM	021C 8410h + (i*40h)

Figure 7-258. ELM_SYNDROME_FRAGMENT_4_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYNDROME_4																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-674. ELM_SYNDROME_FRAGMENT_4_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SYNDROME_4	R/W	0h	Syndrome bits 128 to 159

Table 7-675. Register Call Summary for ELM_SYNDROME_FRAGMENT_4_i

ELM Basic Programming Model • Use Case: ELM Used in Page Mode: [0][1][2][3]
ELM Registers • ELM_SYNDROME_FRAGMENT_4_i Register (Offset = 410h + i*40h) [reset = 0h]: [0] • ELM Registers: [0]

7.4.5.13 ELM_SYNDROME_FRAGMENT_5_i Register (Offset = 414h + i*40h) [reset = 0h]

ELM_SYNDROME_FRAGMENT_5_i (where i = 0 to 7) is shown in Figure 7-259 and described in Table 7-677.

Input syndrome polynomial bits 160 to 191.

Table 7-676. ELM_SYNDROME_FRAGMENT_5_i Instances

Instance	Physical Address
ELM	021C 8414h + (i*40h)

Figure 7-259. ELM_SYNDROME_FRAGMENT_5_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYNDROME_5																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 7-677. ELM_SYNDROME_FRAGMENT_5_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SYNDROME_5	R/W	0h	Syndrome bits 160 to 191

Table 7-678. Register Call Summary for ELM_SYNDROME_FRAGMENT_5_i

ELM Basic Programming Model • Use Case: ELM Used in Page Mode: [0][1][2][3] • Processing Initialization: [0]
ELM Registers • ELM_SYNDROME_FRAGMENT_5_i Register (Offset = 414h + i*40h) [reset = 0h]: [0] • ELM Registers: [0]

7.4.5.14 ELM_SYNDROME_FRAGMENT_6_i Register (Offset = 418h + i*40h) [reset = 0h]

ELM_SYNDROME_FRAGMENT_6_i (where i = 0 to 7) is shown in Figure 7-260 and described in Table 7-680.

Input syndrome polynomial bits 192 to 207.

Table 7-679. ELM_SYNDROME_FRAGMENT_6_i Instances

Instance	Physical Address
ELM	021C 8418h + (i*40h)

Figure 7-260. ELM_SYNDROME_FRAGMENT_6_i Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							SYNDROME_VALID
R-0h							R/W-0h
15	14	13	12	11	10	9	8
SYNDROME_6							
R/W-0h							
7	6	5	4	3	2	1	0
SYNDROME_6							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 7-680. ELM_SYNDROME_FRAGMENT_6_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	SYNDROME_VALID	R/W	0h	Syndrome valid bit 0h: This syndrome polynomial must not be processed. 1h: This syndrome polynomial must be processed.
15-0	SYNDROME_6	R/W	0h	Syndrome bits 192 to 207

Table 7-681. Register Call Summary for ELM_SYNDROME_FRAGMENT_6_i

ELM Basic Programming Model • Use Case: ELM Used in Page Mode: [0][1][2][3][4][5][6][7] • Read Results: [0] • Processing Initialization: [0][1] • Use Case: ELM Used in Continuous Mode: [0]
ELM Registers • ELM_SYNDROME_FRAGMENT_6_i Register (Offset = 418h + i*40h) [reset = 0h]: [0] • ELM Registers: [0]
ELM Functional Description • Processing Completion: [0][1] • Processing Initialization: [0][1][2][3]

7.4.5.15 ELM_LOCATION_STATUS_i Register (Offset = 800h + i*100h) [reset = 0h]

ELM_LOCATION_STATUS_i (where i = 0 to 7) is shown in Figure 7-261 and described in Table 7-683.

Exit status for the syndrome polynomial processing.

Table 7-682. ELM_LOCATION_STATUS_i Instances

Instance	Physical Address
ELM	021C 8800h + (i*100h)

Figure 7-261. ELM_LOCATION_STATUS_i Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							ECC_CORRECTABLE
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED			ECC_NB_ERRORS				
R-0h			R-0h				

LEGEND: R = Read Only; *-n* = value after reset

Table 7-683. ELM_LOCATION_STATUS_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	ECC_CORRECTABLE	R	0h	Error-location process exit status 0h: ECC error-location process failed. Number of errors and error locations are invalid. 1h: All errors were successfully located. Number of errors and error locations are valid.
7-5	RESERVED	R	0h	Reserved
4-0	ECC_NB_ERRORS	R	0h	Number of errors detected and located

Table 7-684. Register Call Summary for ELM_LOCATION_STATUS_i

ELM Basic Programming Model • Use Case: ELM Used in Page Mode: [0][1][2][3][4][5][6][7] • Read Results: [0][1][2][3][4][5][6][7] • Use Case: ELM Used in Continuous Mode: [0][1]
ELM Registers • ELM_LOCATION_STATUS_i Register (Offset = 800h + i*100h) [reset = 0h]: [0] • ELM Registers: [0]
ELM Functional Description • Processing Completion: [0][1][2] • Processing Sequence: [0][1][2][3][4] • Processing Initialization: [0]

7.4.5.16 ELM_ERROR_LOCATION_0_i Register (Offset = 880h + i*100h) [reset = 0h]

ELM_ERROR_LOCATION_0_i (where i = 0 to 7) is shown in Figure 7-262 and described in Table 7-686.

Error-location register 0.

Table 7-685. ELM_ERROR_LOCATION_0_i Instances

Instance	Physical Address
ELM	021C 8880h + (i*100h)

Figure 7-262. ELM_ERROR_LOCATION_0_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECC_ERROR_LOCATION												
R-0h			R-0h												

LEGEND: R = Read Only; *-n* = value after reset

Table 7-686. ELM_ERROR_LOCATION_0_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	ECC_ERROR_LOCATION	R	0h	Error-location bit address

Table 7-687. Register Call Summary for ELM_ERROR_LOCATION_0_i

ELM Basic Programming Model • Use Case: ELM Used in Page Mode: [0][1][2] • Read Results: [0][1] • Use Case: ELM Used in Continuous Mode: [0]
ELM Registers • ELM_ERROR_LOCATION_0_i Register (Offset = 880h + i*100h) [reset = 0h]: [0] • ELM Registers: [0]
ELM Functional Description • Processing Completion: [0] • Processing Sequence: [0][1][2] • Processing Initialization: [0]

7.4.5.17 ELM_ERROR_LOCATION_1_i Register (Offset = 884h + i*100h) [reset = 0h]

ELM_ERROR_LOCATION_1_i (where i = 0 to 7) is shown in Figure 7-263 and described in Table 7-689.

Error-location register 1.

Table 7-688. ELM_ERROR_LOCATION_1_i Instances

Instance	Physical Address
ELM	021C 8884h + (i*100h)

Figure 7-263. ELM_ERROR_LOCATION_1_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECC_ERROR_LOCATION												
R-0h			R-0h												

LEGEND: R = Read Only; *-n* = value after reset

Table 7-689. ELM_ERROR_LOCATION_1_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	ECC_ERROR_LOCATION	R	0h	Error-location bit address

Table 7-690. Register Call Summary for ELM_ERROR_LOCATION_1_i

ELM Basic Programming Model • Use Case: ELM Used in Page Mode: [0][1] • Read Results: [0][1] • Use Case: ELM Used in Continuous Mode: [0]
ELM Registers • ELM_ERROR_LOCATION_1_i Register (Offset = 884h + i*100h) [reset = 0h]: [0] • ELM Registers: [0]
ELM Functional Description • Processing Sequence: [0]

7.4.5.18 ELM_ERROR_LOCATION_2_i Register (Offset = 888h + i*100h) [reset = 0h]

ELM_ERROR_LOCATION_2_i (where i = 0 to 7) is shown in Figure 7-264 and described in Table 7-692.

Error-location register 2.

Table 7-691. ELM_ERROR_LOCATION_2_i Instances

Instance	Physical Address
ELM	021C 8888h + (i*100h)

Figure 7-264. ELM_ERROR_LOCATION_2_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECC_ERROR_LOCATION												
R-0h			R-0h												

LEGEND: R = Read Only; *-n* = value after reset

Table 7-692. ELM_ERROR_LOCATION_2_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	ECC_ERROR_LOCATION	R	0h	Error-location bit address

Table 7-693. Register Call Summary for ELM_ERROR_LOCATION_2_i

ELM Basic Programming Model • Use Case: ELM Used in Page Mode: [0] • Use Case: ELM Used in Continuous Mode: [0]
ELM Registers • ELM_ERROR_LOCATION_2_i Register (Offset = 888h + i*100h) [reset = 0h]: [0] • ELM Registers: [0]

7.4.5.19 ELM_ERROR_LOCATION_3_i Register (Offset = 88Ch + i*100h) [reset = 0h]

ELM_ERROR_LOCATION_3_i (where i = 0 to 7) is shown in Figure 7-265 and described in Table 7-695.

Error-location register 3.

Table 7-694. ELM_ERROR_LOCATION_3_i Instances

Instance	Physical Address
ELM	021C 888Ch + (i*100h)

Figure 7-265. ELM_ERROR_LOCATION_3_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECC_ERROR_LOCATION												
R-0h			R-0h												

LEGEND: R = Read Only; *-n* = value after reset

Table 7-695. ELM_ERROR_LOCATION_3_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	ECC_ERROR_LOCATION	R	0h	Error-location bit address

Table 7-696. Register Call Summary for ELM_ERROR_LOCATION_3_i

ELM Basic Programming Model • Use Case: ELM Used in Page Mode: [0] • Use Case: ELM Used in Continuous Mode: [0]
ELM Registers • ELM_ERROR_LOCATION_3_i Register (Offset = 88Ch + i*100h) [reset = 0h]: [0] • ELM Registers: [0]

7.4.5.20 ELM_ERROR_LOCATION_4_i Register (Offset = 890h + i*100h) [reset = 0h]

ELM_ERROR_LOCATION_4_i (where i = 0 to 7) is shown in Figure 7-266 and described in Table 7-698.

Error-location register 4.

Table 7-697. ELM_ERROR_LOCATION_4_i Instances

Instance	Physical Address
ELM	021C 8890h + (i*100h)

Figure 7-266. ELM_ERROR_LOCATION_4_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECC_ERROR_LOCATION												
R-0h			R-0h												

LEGEND: R = Read Only; *-n* = value after reset

Table 7-698. ELM_ERROR_LOCATION_4_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	ECC_ERROR_LOCATION	R	0h	Error-location bit address

Table 7-699. Register Call Summary for ELM_ERROR_LOCATION_4_i

ELM Registers

- ELM_ERROR_LOCATION_4_i Register (Offset = 890h + i*100h) [reset = 0h]: [0]
- ELM Registers: [0]

7.4.5.21 ELM_ERROR_LOCATION_5_i Register (Offset = 894h + i*100h) [reset = 0h]

ELM_ERROR_LOCATION_5_i (where i = 0 to 7) is shown in Figure 7-267 and described in Table 7-701.

Error-location register 5.

Table 7-700. ELM_ERROR_LOCATION_5_i Instances

Instance	Physical Address
ELM	021C 8894h + (i*100h)

Figure 7-267. ELM_ERROR_LOCATION_5_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECC_ERROR_LOCATION												
R-0h			R-0h												

LEGEND: R = Read Only; *-n* = value after reset

Table 7-701. ELM_ERROR_LOCATION_5_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	ECC_ERROR_LOCATION	R	0h	Error-location bit address

Table 7-702. Register Call Summary for ELM_ERROR_LOCATION_5_i

ELM Registers
- ELM_ERROR_LOCATION_5_i Register (Offset = 894h + i*100h) [reset = 0h]: [0]
- ELM Registers: [0]

7.4.5.22 ELM_ERROR_LOCATION_6_i Register (Offset = 898h + i*100h) [reset = 0h]

ELM_ERROR_LOCATION_6_i (where i = 0 to 7) is shown in Figure 7-268 and described in Table 7-704.

Error-location register 6.

Table 7-703. ELM_ERROR_LOCATION_6_i Instances

Instance	Physical Address
ELM	021C 8898h + (i*100h)

Figure 7-268. ELM_ERROR_LOCATION_6_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECC_ERROR_LOCATION												
R-0h			R-0h												

LEGEND: R = Read Only; *-n* = value after reset

Table 7-704. ELM_ERROR_LOCATION_6_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	ECC_ERROR_LOCATION	R	0h	Error-location bit address

Table 7-705. Register Call Summary for ELM_ERROR_LOCATION_6_i

ELM Registers
- ELM_ERROR_LOCATION_6_i Register (Offset = 898h + i*100h) [reset = 0h]: [0]
- ELM Registers: [0]

7.4.5.23 ELM_ERROR_LOCATION_7_i Register (Offset = 89Ch + i*100h) [reset = 0h]

ELM_ERROR_LOCATION_7_i (where i = 0 to 7) is shown in Figure 7-269 and described in Table 7-707.

Error-location register 7.

Table 7-706. ELM_ERROR_LOCATION_7_i Instances

Instance	Physical Address
ELM	021C 889Ch + (i*100h)

Figure 7-269. ELM_ERROR_LOCATION_7_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECC_ERROR_LOCATION												
R-0h			R-0h												

LEGEND: R = Read Only; *-n* = value after reset

Table 7-707. ELM_ERROR_LOCATION_7_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	ECC_ERROR_LOCATION	R	0h	Error-location bit address

Table 7-708. Register Call Summary for ELM_ERROR_LOCATION_7_i

ELM Registers

- ELM_ERROR_LOCATION_7_i Register (Offset = 89Ch + i*100h) [reset = 0h]: [0]
- ELM Registers: [0]

7.4.5.24 ELM_ERROR_LOCATION_8_i Register (Offset = 8A0h + i*100h) [reset = 0h]

ELM_ERROR_LOCATION_8_i (where i = 0 to 7) is shown in Figure 7-270 and described in Table 7-710.

Error-location register 8.

Table 7-709. ELM_ERROR_LOCATION_8_i Instances

Instance	Physical Address
ELM	021C 88A0h + (i*100h)

Figure 7-270. ELM_ERROR_LOCATION_8_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECC_ERROR_LOCATION												
R-0h			R-0h												

LEGEND: R = Read Only; *-n* = value after reset

Table 7-710. ELM_ERROR_LOCATION_8_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	ECC_ERROR_LOCATION	R	0h	Error-location bit address

Table 7-711. Register Call Summary for ELM_ERROR_LOCATION_8_i

ELM Registers
- ELM_ERROR_LOCATION_8_i Register (Offset = 8A0h + i*100h) [reset = 0h]: [0]
- ELM Registers: [0]

7.4.5.25 ELM_ERROR_LOCATION_9_i Register (Offset = 8A4h + i*100h) [reset = 0h]

ELM_ERROR_LOCATION_9_i (where i = 0 to 7) is shown in Figure 7-271 and described in Table 7-713.

Error-location register 9.

Table 7-712. ELM_ERROR_LOCATION_9_i Instances

Instance	Physical Address
ELM	021C 88A4h + (i*100h)

Figure 7-271. ELM_ERROR_LOCATION_9_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECC_ERROR_LOCATION												
R-0h			R-0h												

LEGEND: R = Read Only; *-n* = value after reset

Table 7-713. ELM_ERROR_LOCATION_9_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	ECC_ERROR_LOCATION	R	0h	Error-location bit address

Table 7-714. Register Call Summary for ELM_ERROR_LOCATION_9_i

ELM Registers
- ELM_ERROR_LOCATION_9_i Register (Offset = 8A4h + i*100h) [reset = 0h]: [0]
- ELM Registers: [0]

7.4.5.26 ELM_ERROR_LOCATION_10_i Register (Offset = 8A8h + i*100h) [reset = 0h]

ELM_ERROR_LOCATION_10_i (where i = 0 to 7) is shown in Figure 7-272 and described in Table 7-716.

Error-location register 10.

Table 7-715. ELM_ERROR_LOCATION_10_i Instances

Instance	Physical Address
ELM	021C 88A8h + (i*100h)

Figure 7-272. ELM_ERROR_LOCATION_10_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECC_ERROR_LOCATION												
R-0h			R-0h												

LEGEND: R = Read Only; *-n* = value after reset

Table 7-716. ELM_ERROR_LOCATION_10_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	ECC_ERROR_LOCATION	R	0h	Error-location bit address

Table 7-717. Register Call Summary for ELM_ERROR_LOCATION_10_i

ELM Registers

- ELM_ERROR_LOCATION_10_i Register (Offset = 8A8h + i*100h) [reset = 0h]: [0]
- ELM Registers: [0]

7.4.5.27 ELM_ERROR_LOCATION_11_i Register (Offset = 8ACh + i*100h) [reset = 0h]

ELM_ERROR_LOCATION_11_i (where i = 0 to 7) is shown in Figure 7-273 and described in Table 7-719.

Error-location register 11.

Table 7-718. ELM_ERROR_LOCATION_11_i Instances

Instance	Physical Address
ELM	021C 88ACh + (i*100h)

Figure 7-273. ELM_ERROR_LOCATION_11_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECC_ERROR_LOCATION												
R-0h			R-0h												

LEGEND: R = Read Only; *-n* = value after reset

Table 7-719. ELM_ERROR_LOCATION_11_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	ECC_ERROR_LOCATION	R	0h	Error-location bit address

Table 7-720. Register Call Summary for ELM_ERROR_LOCATION_11_i

ELM Registers
- ELM_ERROR_LOCATION_11_i Register (Offset = 8ACh + i*100h) [reset = 0h]: [0]
- ELM Registers: [0]

7.4.5.28 ELM_ERROR_LOCATION_12_i Register (Offset = 8B0h + i*100h) [reset = 0h]

ELM_ERROR_LOCATION_12_i (where i = 0 to 7) is shown in Figure 7-274 and described in Table 7-722.

Error-location register 12.

Table 7-721. ELM_ERROR_LOCATION_12_i Instances

Instance	Physical Address
ELM	021C 88B0h + (i*100h)

Figure 7-274. ELM_ERROR_LOCATION_12_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECC_ERROR_LOCATION												
R-0h			R-0h												

LEGEND: R = Read Only; *-n* = value after reset

Table 7-722. ELM_ERROR_LOCATION_12_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	ECC_ERROR_LOCATION	R	0h	Error-location bit address

Table 7-723. Register Call Summary for ELM_ERROR_LOCATION_12_i

ELM Registers

- ELM_ERROR_LOCATION_12_i Register (Offset = 8B0h + i*100h) [reset = 0h]: [0]
- ELM Registers: [0]

7.4.5.29 ELM_ERROR_LOCATION_13_i Register (Offset = 8B4h + i*100h) [reset = 0h]

ELM_ERROR_LOCATION_13_i (where i = 0 to 7) is shown in Figure 7-275 and described in Table 7-725.

Error-location register 13.

Table 7-724. ELM_ERROR_LOCATION_13_i Instances

Instance	Physical Address
ELM	021C 88B4h + (i*100h)

Figure 7-275. ELM_ERROR_LOCATION_13_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECC_ERROR_LOCATION												
R-0h			R-0h												

LEGEND: R = Read Only; *-n* = value after reset

Table 7-725. ELM_ERROR_LOCATION_13_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	ECC_ERROR_LOCATION	R	0h	Error-location bit address

Table 7-726. Register Call Summary for ELM_ERROR_LOCATION_13_i

ELM Registers

- ELM_ERROR_LOCATION_13_i Register (Offset = 8B4h + i*100h) [reset = 0h]: [0]
- ELM Registers: [0]

7.4.5.30 ELM_ERROR_LOCATION_14_i Register (Offset = 8B8h + i*100h) [reset = 0h]

ELM_ERROR_LOCATION_14_i (where i = 0 to 7) is shown in Figure 7-276 and described in Table 7-728.

Error-location register 14.

Table 7-727. ELM_ERROR_LOCATION_14_i Instances

Instance	Physical Address
ELM	021C 88B8h + (i*100h)

Figure 7-276. ELM_ERROR_LOCATION_14_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECC_ERROR_LOCATION												
R-0h			R-0h												

LEGEND: R = Read Only; *-n* = value after reset

Table 7-728. ELM_ERROR_LOCATION_14_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	ECC_ERROR_LOCATION	R	0h	Error-location bit address

Table 7-729. Register Call Summary for ELM_ERROR_LOCATION_14_i

ELM Registers
- ELM_ERROR_LOCATION_14_i Register (Offset = 8B8h + i*100h) [reset = 0h]: [0]
- ELM Registers: [0]

7.4.5.31 ELM_ERROR_LOCATION_15_i Register (Offset = 8BCh + i*100h) [reset = 0h]

ELM_ERROR_LOCATION_15_i (where i = 0 to 7) is shown in Figure 7-277 and described in Table 7-731.

Error-location register 15.

Table 7-730. ELM_ERROR_LOCATION_15_i Instances

Instance	Physical Address
ELM	021C 88BCh + (i*100h)

Figure 7-277. ELM_ERROR_LOCATION_15_i Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			ECC_ERROR_LOCATION												
R-0h			R-0h												

LEGEND: R = Read Only; *-n* = value after reset

Table 7-731. ELM_ERROR_LOCATION_15_i Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	ECC_ERROR_LOCATION	R	0h	Error-location bit address

Table 7-732. Register Call Summary for ELM_ERROR_LOCATION_15_i

ELM Basic Programming Model • Read Results: [0][1]
ELM Registers • ELM_ERROR_LOCATION_15_i Register (Offset = 8BCh + i*100h) [reset = 0h]: [0] • ELM Registers: [0]
ELM Functional Description • Processing Completion: [0] • Processing Sequence: [0][1] • Processing Initialization: [0]

Interprocessor Communication

This chapter describes the Message Manager and Semaphore modules for the device.

Topic **Page**

8.1 Message Manager

This section describes the Message Manager module on the SoC.

8.1.1 Message Manager Overview

8.1.1.1 Introduction

The SoC implements a single instance of the Message Manager to provide inter-processor communication between the various processing units:

- Arm A15
- C66x DSP
- PMMC
- PRU-ICSS

The Message Manager is a hardware engine used for queuing messages in a secure and self-contained manner. There is no limitation on the message format or content. It is software responsibility to define the message format.

The Message Manager provides a multi-core safe message interface which allows multiple users (message senders and receivers) to access the queues without the need for any mutual exclusion. It also allows for secure and authorized access to the queues.

8.1.1.2 Key Features

The general features of the Message Manager module include:

- Provides hardware acceleration for pushing/popping messages to/from logical queues
- Supports the following SoC configuration:
  - 64 queues
  - Up to 128 pending messages
  - 64-byte messages
  - 32 proxies (single proxy per page)
- Support for highly-pipelined push/pop operations
- Support for self-contained mode with zero SW initialization
- Provides a secure front-end for the queues
- Provides flexible message allocation with ability to store the same message multiple times in different queues or multiple times in the same queue
- Queue depth limited only by the maximum number of messages
- Support for little-endian (LE) operation only

Monitoring and trace functions include:

- Provides hardware signals to monitor the empty status for all transmit source queues
- Provides ability to read Linking RAM contents for debug purposes
- Provides ability to generate an interrupt when there are no free entries in the Linking RAM
- Provides ability to generate an interrupt due to a proxy fault

8.1.2 Message Manager Integration

This section describes the module integration in the device, including information about clocks, resets, and hardware requests.

Figure 8-1 shows the Message Manager integration.

Figure 8-1. Message Manager Integration

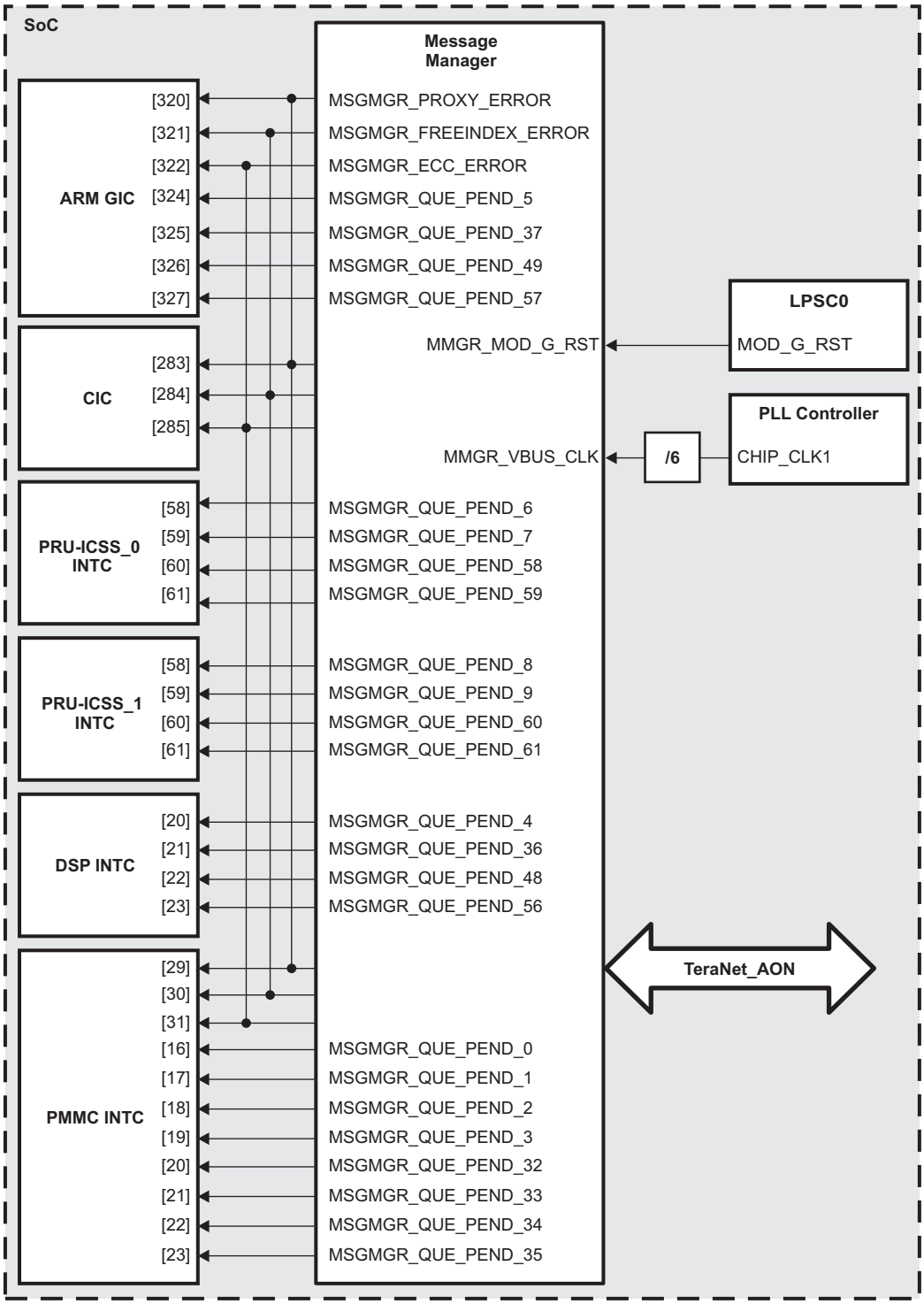


Table 8-1 through Table 8-3 summarize the integration of the module in the device.

Table 8-1. Message Manager Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
Message Manager	PD0	LPSC0	TeraNet_AON

Table 8-2. Message Manager Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
Message Manager	MMGR_VBUS_CLK	CHIP_CLK1 / 6	PLL Controller	Interface and Funtional Clock
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
Message Manager	MMGR_MOD_G_RST	MOD_G_RST	LPSC0	Module Reset

Table 8-3. Message Manager Hardware Requests

Module Instance	Event Name	Interrupt Requests						Description
		Mapped To Input Event [Number]						
		ARM GIC	CIC	PRU-ICSS_0 INTC	PRU-ICSS_1 INTC	DSP INTC	PMMC INTC	
Message Manager	MSGMGR_PROXY_ERROR	[320]	[283]	-	-	-	[29]	Proxy error interrupt
	MSGMGR_FREEINDEX_ERROR	[321]	[284]	-	-	-	[30]	No free index available interrupt
	MSGMGR_ECC_ERROR	[322]	[285]	-	-	-	[31]	ECC error interrupt
	MSGMGR_QUE_PEND_5	[324]	-	-	-	-	-	Queue pending signal - queue 5
	MSGMGR_QUE_PEND_37	[325]	-	-	-	-	-	Queue pending signal - queue 37
	MSGMGR_QUE_PEND_49	[326]	-	-	-	-	-	Queue pending signal - queue 49
	MSGMGR_QUE_PEND_57	[327]	-	-	-	-	-	Queue pending signal - queue 57
	MSGMGR_QUE_PEND_4	-	-	-	-	[20]		Queue pending signal - queue 4
	MSGMGR_QUE_PEND_36	-	-	-	-	[21]		Queue pending signal - queue 36
	MSGMGR_QUE_PEND_48	-	-	-	-	[22]		Queue pending signal - queue 48
	MSGMGR_QUE_PEND_56	-	-	-	-	[23]		Queue pending signal - queue 56
	MSGMGR_QUE_PEND_6	-	-	[58]	-	-	-	Queue pending signal - queue 6
	MSGMGR_QUE_PEND_7	-	-	[59]	-	-	-	Queue pending signal - queue 7
	MSGMGR_QUE_PEND_58	-	-	[60]	-	-	-	Queue pending signal - queue 58
	MSGMGR_QUE_PEND_59	-	-	[61]	-	-	-	Queue pending signal - queue 59
	MSGMGR_QUE_PEND_8	-	-	-	[58]	-	-	Queue pending signal - queue 8
	MSGMGR_QUE_PEND_9	-	-	-	[59]	-	-	Queue pending signal - queue 9
	MSGMGR_QUE_PEND_60	-	-	-	[60]	-	-	Queue pending signal - queue 60
	MSGMGR_QUE_PEND_61				[61]			Queue pending signal - queue 61
	MSGMGR_QUE_PEND_0	-	-	-	-	-	[16]	Queue pending signal - queue 0
	MSGMGR_QUE_PEND_1	-	-	-	-	-	[17]	Queue pending signal - queue 1
	MSGMGR_QUE_PEND_2	-	-	-	-	-	[18]	Queue pending signal - queue 2
	MSGMGR_QUE_PEND_3	-	-	-	-	-	[19]	Queue pending signal - queue 3
	MSGMGR_QUE_PEND_32	-	-	-	-	-	[20]	Queue pending signal - queue 32
	MSGMGR_QUE_PEND_33	-	-	-	-	-	[21]	Queue pending signal - queue 33
	MSGMGR_QUE_PEND_34	-	-	-	-	-	[22]	Queue pending signal - queue 34
	MSGMGR_QUE_PEND_35	-	-	-	-	-	[23]	Queue pending signal - queue 35

8.1.3 Message Manager Functional Description

8.1.3.1 Basics and SoC Implementation

The Message Manager provides a standard API interface between the various hosts on the system. A host can be kernel, hypervisor or even user processes.

The Message Manager provides a set of queues that can be used to temporarily store and pass messages between the hosts. The messages are allocated and stored internally automatically and can be placed into any queue, regardless of the data contents. The data returned is exactly the same as the data inserted into the queues. For each queue there is a dedicated *queue pending* signal to indicate if the queue is empty or not.

The current Message Manager configuration supports 64-byte messages, 64 queues and it has buffer space to accommodate 128 pending messages.

Table 8-4 summarizes the SoC configuration of the Message Manager.

Table 8-4. Message Manager SoC Configuration

Parameter Name	Parameter Description	SoC Value	Comment
QCNT	Number of queues	64	64 queues
MAX_MSG_SIZE	Message size (in bytes)	80	64B message(1)
MAX_MESSAGES	Number of messages	128	128 messages
Q_PROXIES	Number of queue proxies per page	1	Single proxy per page

(1) Total message size includes: 16B control words + 64B actual message

The Message Manager provides one proxy region per host. The proxy is a module that provides atomic queue pushes across the cores in the device. The purpose of the proxy is to accept a queue write by a given core without allowing another core to inject its own push.

The trusted OS is responsible to configure the proxy region for each host. A total of 32 proxy regions are supported and each proxy region is aligned in size with MMU page (64KB).

Each proxy supports 64 virtual queue slots:

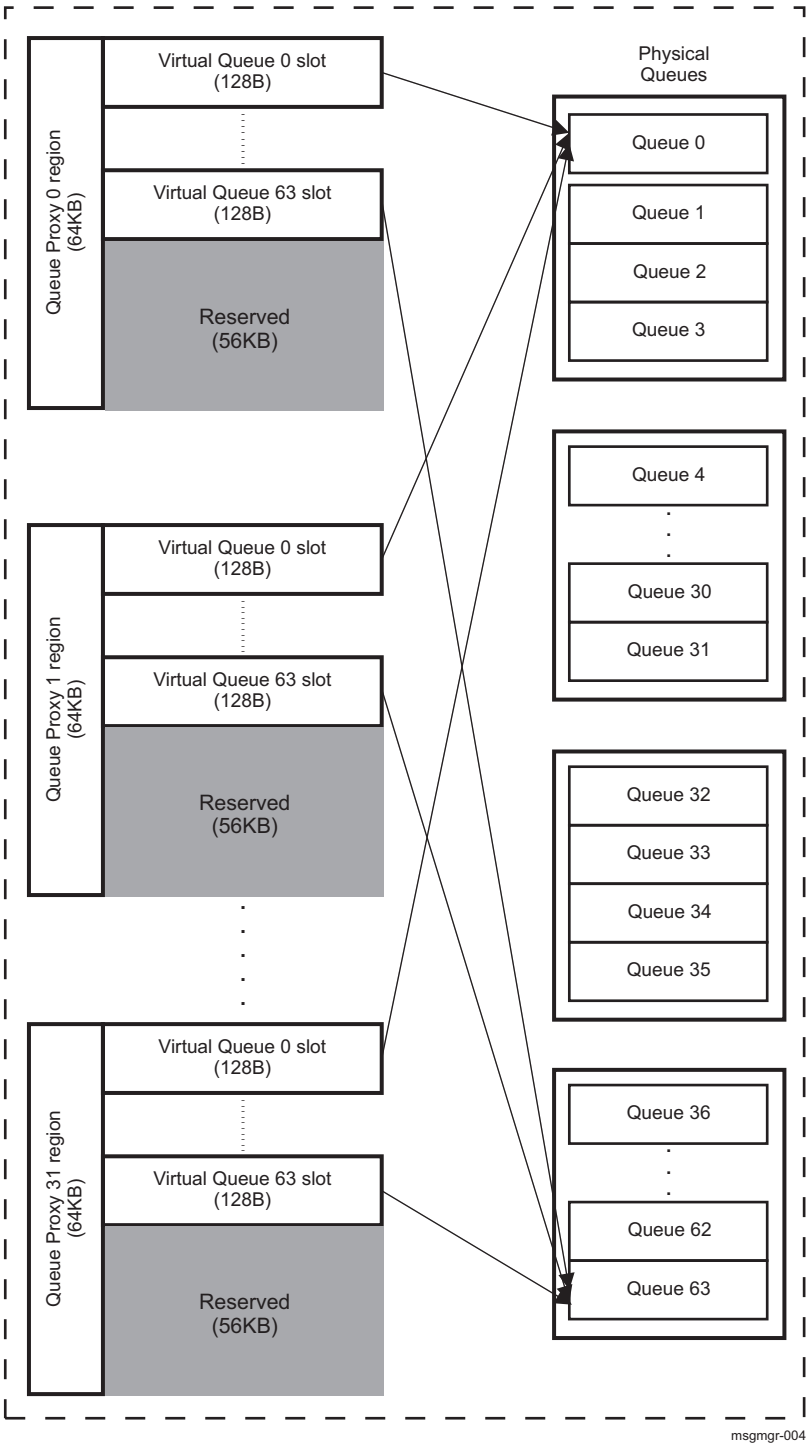
- Each virtual queue slot occupies 128B address space
- The same virtual queue slot in any proxy region is mapped to the same physical queue inside the Message Manager (that is, one-to-one mapping)
- Each virtual queue slot per proxy can be individually set access privilege to allow sending (write) or receiving (read) access.

NOTE: The actual size of each proxy region is 8KB. To align the proxy with a page size of 64KB, there is a 56KB reserved memory space.

A Memory Protection Unit (MPU_0) is placed in front of the Message Manager to protect the secure virtual queue region.

Figure 8-2 shows the proxy/queue mapping.

Figure 8-2. Message Manager Proxy/Queue Mapping



The SoC implementation supports a *single proxy per page* configuration and makes use of 9 of the 32 available pages/proxies. Proxies are assigned as shown in Table 8-5.

Table 8-5. Message Manager Proxy Assignment

Proxy	Usage
0	PMMC
1	DSP (non-secure)

Table 8-5. Message Manager Proxy Assignment (continued)

Proxy	Usage
2	A15 (non-secure)
3	PRU-ICSS_0_PRU0 (non-secure)
4	PRU-ICSS_0_PRU1 (non-secure)
5	PRU-ICSS_1_PRU0 (non-secure)
6	PRU-ICSS_1_PRU1 (non-secure)
7	DSP (secure)
8	A15 (secure)
9-31	Not used

Table 8-6 shows the recommended queue assignment based on the proxy mapping and queue pending event connectivity. Queue access is indicated as: *Read-Only (RO)*, *Write-Only (WO)*, or *No Read / No Write (none)*.

Table 8-6. Message Manager Queue Assignment

Queue	Usage	Proxy 0 PMMC	Proxy 1 DSP	Proxy 2 ARMSS	Proxy 3 PRU-ICSS_0 PRU0	Proxy 4 PRU-ICSS_0 PRU1	Proxy 5 PRU-ICSS_1 PRU0	Proxy 6 PRU-ICSS_1 PRU1	Proxy 7 DSP Secure	Proxy 8 ARMSS Secure
0	PMMC Non-Sec Fwd (Priority 0)	RO	WO	WO	WO	WO	WO	WO	None	None
1	PMMC Non-Sec Fwd (Priority 1)	RO	WO	WO	WO	WO	WO	WO	None	None
2	PMMC Non-Sec Fwd (Priority 2)	RO	WO	WO	WO	WO	WO	WO	None	None
3	PMMC Non-Sec Fwd (Priority 3)	RO	WO	WO	WO	WO	WO	WO	None	None
4	PMMC Non-Sec DSP Rtn	WO	RO	None	None	None	None	None	None	None
5	PMMC Non-Sec ARMSS Rtn	WO	None	RO	None	None	None	None	None	None
6	PMMC Non-Sec PRU-ICSS_0.PRU0 Rtn	WO	None	None	RO	None	None	None	None	None
7	PMMC Non-Sec PRU-ICSS_0.PRU1 Rtn	WO	None	None	None	RO	None	None	None	None
8	PMMC Non-Sec PRU-ICSS_1.PRU0 Rtn	WO	None	None	None	None	RO	None	None	None
9	PMMC Non-Sec PRU-ICSS_1.PRU1 Rtn	WO	None	None	None	None	None	RO	None	None
10-31	Reserved	None	None	None	None	None	None	None	None	None
32	PMMC Sec Fwd (Priority 0)	RO	None	None	None	None	None	None	WO	WO
33	PMMC Sec Fwd (Priority 1)	RO	None	None	None	None	None	None	WO	WO
34	PMMC Sec Fwd (Priority 2)	RO	None	None	None	None	None	None	WO	WO
35	PMMC Sec Fwd (Priority 3)	RO	None	None	None	None	None	None	WO	WO
36	PMMC Sec DSP Rtn	WO	None	None	None	None	None	None	RO	None
37	PMMC Sec ARMSS Rtn	WO	None	None	None	None	None	None	None	RO
38-47	Reserved	None	None	None	None	None	None	None	None	None
48	IPC Sec DSP	None	None	None	None	None	None	None	RO	WO
49	IPC Sec ARMSS	None	None	None	None	None	None	None	WO	RO
50-55	Reserved	None	None	None	None	None	None	None	None	None
56	IPC Non-Sec DSP	WO	RO	WO	WO	WO	WO	WO	RO	WO
57	IPC Non-Sec ARMSS	WO	WO	RO	WO	WO	WO	WO	WO	RO
58	IPC Non-Sec PRU-ICSS_0.PRU0	WO	WO	WO	RO	WO	WO	WO	WO	WO
59	IPC Non-Sec PRU-ICSS_0.PRU1	WO	WO	WO	WO	RO	WO	WO	WO	WO
60	IPC Non-Sec PRU-ICSS_1.PRU0	WO	WO	WO	WO	WO	RO	WO	WO	WO
61	IPC Non-Sec PRU-ICSS_1.PRU1	WO	WO	WO	WO	WO	WO	RO	WO	WO
62-63	Reserved	None	None	None	None	None	None	None	None	None

8.1.3.2 Queue Access

A queue access is a simple read or write burst on the bus to a particular offset. A burst is required to keep the queue accesses coherent and atomic from other queue accesses. When a burst cannot be achieved by an endpoint itself, this is guaranteed by the corresponding proxy region instead.

Each queue has a dedicated address space, and these spaces are packed together to form all the queues, allowing the offset to indicate which queue is being accessed.

A write burst will add the data to the tail of the selected queue. The write burst does not need to cover the entire data size, and any unwritten bytes will be assumed to be their default values or 0s otherwise.

A read burst will find the head data item on the selected queue, and return the stored data, removing it from the queue. If the read burst does not cover the entire data size, then any unread bytes will be lost. If there is a read burst to an empty queue, the data returned will be all 0s.

See Section 8.1.3.5 for more details on queuing / dequeuing messages.

8.1.3.3 Queue Processing

The queue processing is handled by two components: the dispatcher and the engine. There are also three structures that store state about the queues and items: the queue state memory (Queue State RAM), storage memory (Queue Data RAM) and the link memory (Linking RAM). All the links within the queues are in terms of links to internal item storage and not any pointers included in the data. These link values are automatically allocated during writes.

8.1.3.3.1 Free Index Allocation

For every push (write burst) operation, the Message Manager needs to allocate an index to the message before pushing it to a queue. This index ranges from 0 to 127. The Message Manager allocates this index from a free index pool. On every pop (read burst), the index is returned back to the free index pool.

NOTE: An error interrupt (MSGMGR_FREEINDEX_ERROR) occurs when a push is attempted and all the storage items have already been allocated. This is something that SW should avoid, as the data written causing the error is lost.

8.1.3.3.2 Queue Dispatcher and Engine

The Queue Dispatcher decodes the queue access and forwards it to the engine. It also checks for hazards when the next queue access is to the same queue that is already being processed by an engine. In this case, it will stall the dispatch until all processing on that queue has been completed.

In other words, the Queue Dispatcher is responsible for scheduling the push and pop accesses to the queues. All operations to the queues are done in order. Maximum performance is achieved when back-to-back pop operations are done from multiple queues. A back-to-back pop from the same queue will be stalled until the Queue Engine has fetched the next linking index from the Linking RAM and updated the internal Queue State RAM.

The Queue Engine takes the queue access and the current queue state and performs the requested operation.

The following sequence of operations is automatically done for a push:

1. Get a free index from the free index pool
2. Write the queue contents in the Queue Data RAM using the free index as the address
3. Update the next index in the Linking RAM
4. Update the Queue State RAM for the corresponding queue via the write-back interface
5. Update any statistics/monitoring information (such as queue empty status)

The following sequence of operations is automatically done for a pop:

1. Read the head index from the Queue State RAM
2. Read the queue contents from the Queue Data RAM and return the data on the VBUSM interface

3. Read the next index from the Linking RAM and update the Queue State RAM via the write-back interface
4. Return the current head index back to the free index pool
5. Update any statistics/monitoring information (queue empty status)

8.1.3.3.3 Queue Memory (Queue State RAM)

The Message Manager uses a Queue State RAM (64x21-bit) to record the value of head index and entry count for each queue. When a push/pop operation is performed, the Message Manager updates the word corresponding to the queue number to reflect its new head index and entry count.

The Queue State RAM is available for debug inspection only (see Section 8.1.5.7).

8.1.3.3.4 Link Memory (Linking RAM)

The Message Manager uses a Linking RAM (128x7-bit) to store information about how the messages are logically connected to one another in the various queues. Each location in the Linking RAM corresponds to one message index and stores information about the next linking index.

The linking information for all messages is stored in a contiguous fashion in the Linking RAM. That is, the information for message index 0 is stored at Linking RAM entry 0, the information for message index 1 is stored at Linking RAM entry 1, etc.

The Linking RAM is available for debug inspection only (see Section 8.1.5.6).

8.1.3.3.5 Storage Memory (Queue Data RAM)

All data that is queued will be stored internally in the module in local storage (Queue Data RAM, 1024x64-bit).

8.1.3.4 Queue Protection

Queue protection is achieved via the Queue Virtualization Unit (QVU). The QVU is a method to provide the application SW with an access to the Message Manager resources with low overhead but still ensure the access protections provided by the kernel. Software always accesses the Message Manager via the QVU.

The QVU provides the following functions:

- Access policy checks configured for a given 64KB page of queues that are dedicated to a user process. A total of 32 pages are supported (but only 9 of them are used in this SoC; see)
- Supports default screen door settings to remove the need for SW initialization
- Acts as a proxy to the SW user processes accessing the Message Manager. A single proxy is provided for every page. This implies that there is a single owner of the proxy for every page and the owner is either a single master/core or a single thread on a master
- Flags an error if any SW access violates the page policy checks
- Provides a message bus interface to the Message Manager internal resources

An important consideration is using the proxy in a multitasking environment. The proxy cannot differentiate writes having different sources within the same core (such as multiple threads). Since the proxy cannot distinguish messages between different threads, a partial push to a given queue by *thread X* followed by a completion write to the same queue by *thread Y* will be regarded as a single message write to that queue. No error will be flagged in this case. It is up to software to ensure that the accesses to the same queue are not interleaved between threads.

The Message Manager will flag an error and not queue the messages for the page if the following conditions are true:

- A partial push to a queue was followed by a completion write to a different queue from the same page. The Message Manager will not the queue the current or subsequent messages to any queues accessed from that page until the error status bit has been cleared
- A partial push to a queue was followed by a partial push to a different queue from the same page

8.1.3.5 Passing Messages

The Message Manager provides the mechanism to queue messages in a multi-core safe method. The messages are arranged as a set of 32-bit memory-mapped registers (MMRs).

8.1.3.5.1 Queuing Messages

Messages are queued onto a logical queue by writing a burst of information to the appropriate Queue Data Registers (described in Section 8.1.5.9).

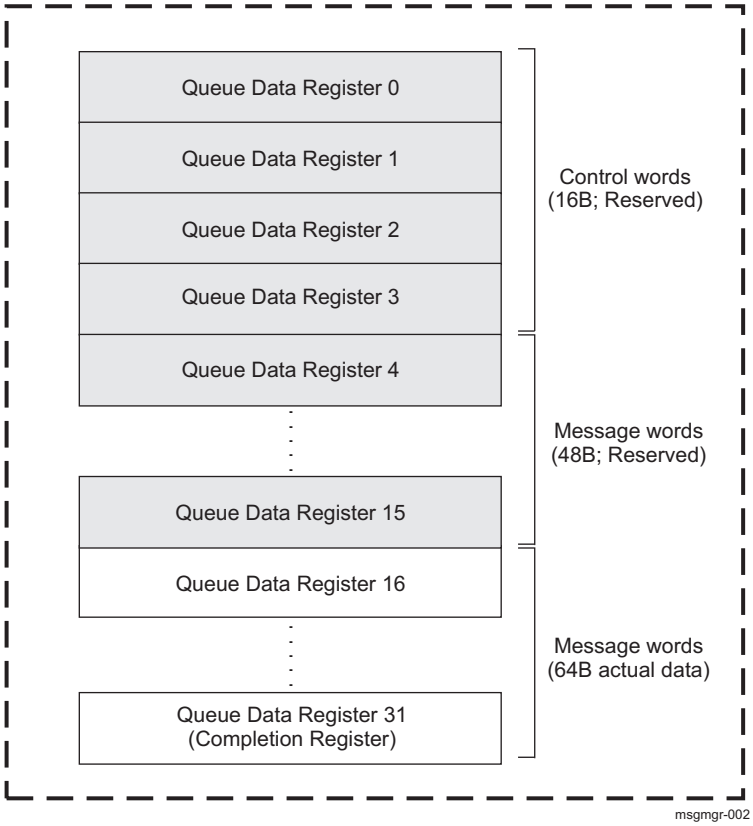
Figure 8-3 shows the message structure supported by the Message Manager. The current Message Manager implementation supports a 80B max message size which includes:

- **16B control words**: The control words are contained in Queue Data Registers 0–3. These registers are reserved for future use
- **64B actual message**: The actual message occupies one or more Queue Data Registers in the range 16–31

NOTE: Queue Data Registers 4–15 are reserved and an access to any of these registers will be ignored or return 0s.

Queue Data Register 31 is also called Completion Register because it must be written (or read) in oder to complete a push (or a pop). The value written in this register can be the last word in a message or simply a dummy write to trigger the push, in cases where the message is smaller than 64B. The push is completed only when a write is done to Queue Data Register 31.

Figure 8-3. Message Manager – Message Structure



With the exclusion of the control words, there is no restriction on the message content or format that is pushed into a queue. It is completely up to software to define the message protocol that the sender and the receiver can understand.

The Message Manager tracks the messages in a queue by an index. The index is a serial number assigned to each message that is internally allocated and deallocated by the Message Manager. To queue a message, the Message Manager allocates an index from its free index pool. The index is used to write the message into the Queue Data RAM. The queue push or pop is not complete until the Completion Register 31 is written or read.

Note that:

- The Message Manager will return stale data for the words that were not written in a message
- Any partial writes to a queue in a page will not be explicitly associated with that queue unless Queue Data Register 31 is written. In other words, if a partial push to a queue (e.g: *queue Y*) is followed by a push to Queue Data Register 31 in *queue X* from the same page, then the Message Manager will drop both messages and flag an error

Once the entire message is written, the Message Manager uses the physical index to link the queue message onto the message chain that is maintained for that logical queue by writing the linking information out to the Linking RAM. The Message Manager also updates the queue head and tail pointers. Since logical queues within the Message Manager are maintained using linked lists, queues cannot become full and no check for fullness is required before a message is queued.

8.1.3.5.2 Dequeuing Messages

Messages are dequeued from a logical queue by reading the corresponding Queue Data Registers. If the queue is empty, it returns a value of 0h. A read operation from an idle proxy will trigger a pop/dequeue from the head of a queue and will initiate an access to the Queue State RAM to get the head index which will then be used to read the contents from the Queue Data RAM. The Message Manager will then proceed to read the next link pointer in the Linking RAM and will stall any subsequent pop operations from the same queue until the next pointer has been read and loaded into the Queue State RAM.

Regardless of the message size, the proxy read operation is only completed if Queue Data Register 31 is read and any other reads are from the same message.

After the pop/dequeue operation, another proxy is free to read a message from the same queue and it will trigger another pop/dequeue to the queue which will return the next message on the queue. In this manner, two Users of the same queue will never read the same message.

NOTE: If the message queue is empty AND an user attempts to read any register != Queue Data Register 31, then the queue will not show the actual message when it arrives unless Queue Data Register 31 is read.

8.1.3.6 Interrupts

The Message Manager supports two kinds of interrupts/events:

- Error interrupts
- Queue pending events

The Message Manager will generate an error interrupt under the following conditions:

- When a proxy fault occurs due to either:
  - An access violating the queue read/write permissions, or
  - An access to a different queue than the current queue which has a partial message written
- When no free index is available in the linking RAM. This will only happen if more than 128 messages are queued. In this case, the Message Manager will ignore the access
- When there is an ECC-related error

The Message Manager also ouputs a group of queue pending signals (1 per queue) to indicate the empty status for each queue. When a given queue is not empty, the corresponding queue pending signal becomes active (asserted).

Table 8-7 shows the Message Manager interrupt types and their polarity.

Table 8-7. Message Manager Interrupt Types

Interrupt Name	Description	Type (At IP Boundary)
MSGMGR_PROXY_ERROR	Proxy error interrupt	Active-high PULSE
MSGMGR_FREEINDEX_ERROR	No free index available interrupt	Active-high PULSE
MSGMGR_ECC_ERROR	ECC error interrupt	Active-high PULSE
MSGMGR_QUE_PEND_[63:0]	Queue pending signals [63:0]	Active-high LEVEL

The Message Manager interrupt mapping is presented in Section 8.1.2.

The Message Manager registers related to the *proxy fault* and *no free index* interrupts are descibed in Section 8.1.5.4 and Section 8.1.5.5, respectively.

8.1.3.7 Power, Clock and Reset

The Message Manager resides in the ALWAYS_ON power domain (PD0).

The Message Manager runs on a single clock (CHIP_CLK1 / 6). All functional and VBUS operations are synchronized to this clock.

The Message Manager uses one hardware reset (LPSC0.MOD_G_RST). A hardware reset initializes all internal logic of the Message Manager and its configuration registers.

The Message Manager does not support a software reset at module level.

8.1.3.8 ECC Support

Error Correction Code (ECC) is a mechanism for providing increased system reliability (via reduction of memory soft errors) by allowing single bit errors to be detected and corrected, and double bit errors to be detected. Several memories within the Message Manager are ECC-protected by an ECC Wrapper which implements *Single Error Correction and Double Error Detection (SECDED)*:

- *Single Error Detection and Correction (SED/SEC)*: This logic detects and corrects a single bit error (1-bit error per ECC word or per ECC data segment). For memories that contain critical and/or persistent data, automatic (immediate or delayed) write-back of the corrected data to the corresponding memory address is supported. In addition, the ECC Wrapper also supports multiple options for partial word writes, such as read-modify-write (R-M-W) or multiple ECC code segments per word.
- *Double Error Detection (DED)*: This logic only detects (does not correct) double errors (2-bit errors per ECC word or per ECC data segment). If a double error is detected, the module can have unpredictable behavior and must be reset to guarantee correct behavior once again.

ECC Aggregator module (integrated inside the Message Manager) consolidates the ECC configuration and status bits for all the ECC-supported memories within the Message Manager. It provides a single End-of-Interrupt (EOI) handshake based interrupt to the host (for both single and double error detections) and a standard 32-bit VBUSP interface for configuring and querying the various ECC Wrappers via their respective register set.

Table 8-8 lists the Message Manager memories that are ECC-protected along with the ECC attribute for each memory. Basic features for all the Message Manager memories are as follows:

- ECC-enabled RAMs have delayed write-back
- All control and status is handled by the ECC Aggregator

Table 8-8. Message Manager RAMs and ECC Options

RAM Wrapper Name	Data Width (bits)	Width with ECC (bits)	Depth	R-M-W
Queue Data RAM	64	8	1024	Y
Queue State RAM	21	6	64	N
Linking RAM	7	5	128	Y
Proxy Config RAM	32	7	128	Y

For more details on ECC Wrapper and ECC Aggregator, see Section 11.14.4.19.1, *ECC Wrapper and ECC Aggregator*.

The ECC-related registers of the Message Manager are described in Section 8.1.5.8.

8.1.4 Message Manager Programming Guidelines

8.1.4.1 Proxy Queue Page Configuration

The first step in using the Message Manager is to configure the proxy queue page. The purpose of the proxy queue is to facilitate masters' autonomous operations so there is no conflict (no race conditions) between multiple masters that try to send multiple messages to the same queue. The Message Manager has 32 proxy queues but in this SoC only 9 proxy queues are available (see Table 8-5 for the allocation of proxy queue to a host).

Each master CPU is assigned a single proxy queue, so if the CPU employed multiple threads, they all share the same proxy queue. Note that non-secure ARMSS and secure ARMSS are considered as two separate masters, and the same is true for the DSP as well.

Each proxy supports mapping of 64 queues and each queue can hold up to 32 values (4 bytes * 32 registers = 128 bytes). Thus, the size of each proxy is 8KB (4 bytes * 32 registers * 64 queues). To align the proxy with a page size of 64KB (10000h), 56KB of addressed memory is reserved.

Table 8-9 summarizes the base physical memory of the proxy queues.

Table 8-9. Proxy Queue Physical Memory

Proxy Number and Master	Start Address	End Address
Proxy 0, PMMC	02A0 0000h	02A0 1FFCh
Proxy 1, DSP non-secure	02A1 0000h	02A1 1FFCh
Proxy 2, A15 non-secure	02A2 0000h	02A2 1FFCh
Proxy 3, PRU-ICSS_0_PRU0 non-secure	02A3 0000h	02A3 1FFCh
Proxy 4, PRU-ICSS_0_PRU1 non-secure	02A4 0000h	02A4 1FFCh
Proxy 5, PRU-ICSS_1_PRU0 non-secure	02A5 0000h	02A5 1FFCh
Proxy 6, PRU-ICSS_1_PRU1 non-secure	02A6 0000h	02A6 1FFCh
Proxy 7, DSP secure	02A7 0000h	02A7 1FFCh
Proxy 8, A15 secure	02A8 0000h	02A8 1FFCh
Proxy 9 to 31	Reserved	Reserved

To open a proxy queue, the software must associate logical memory with the physical memory from above. Thus, A15 non-secure code will define a logical page that will be mapped into physical address 02A2 0000h, and the DSP non-secure code will configure logical address to physical address 02A1 0000h.

8.1.4.2 Sending a Message

To send a message to a destination, a master must do the following:

1. Choose the destination queue. Each message queue is connected to a destination. Example: if the message should go only to the DSP, the sender can choose either queue 4, or queue 36, or queue 48, or queue 56.
2. Next, the software should allocate the logical address of the destination queue that was chosen. This is done by adding the destination queue offset to the base address of the proxy queue. The offset of queue N from the base address of the proxy is 80h*N. Examples:
   - The sender is the non-secure DSP and the physical address of the proxy queue is 02A1 0000h that is mapped to the same logical address. The destination is A15 and the user chooses queue 5. The offset of queue 5 is 5*80h = 280h, so the first register of queue 5 for non-secure DSP is at location 02A1 0280h.
   - The sender is the secure A15 and the physical address of the proxy queue is 02A8 0000h that is mapped into same logical memory. The destination is PMMC and the user chooses queue 32. The offset of queue 32 is 32*80h = 1000h, so the first register of queue 32 for secure A15 is at location 02A8 1000h.
3. The master is ready to write the message to the proxy queue. However, the proxy queues are not protected from multiple writers to the same proxy, and multiple writers (or reader and writer

simultaneously) can cause conflicts. Thus, systems with multiple threads or multiple tasks must protect the proxy queue write operation when one thread (or task) starts writing and prevents other threads from accessing the proxy queue until the write operation is done. The software must define software semaphore or similar lock mechanism to ensure autonomous write operation to a proxy queue.

4. After the writer master secures the semaphore (or any other lock mechanism) for the proxy queue, it can start writing the message. The message is up to 64 bytes long and resides in registers 16 to 31 of the queue message. Registers 0 to 3 are the header registers that are populated by the Message Manager hardware and registers 4 to 15 are reserved. The queue proxy pushes the message into the queue only after register 31 is written, so even if the message is short, the writer must write a dummy value to register 31 in order to initiate the push operation.
5. The message protocol is the sole responsibility of the software. The hardware is indifferent of the payload structure as long as it occupies registers 16 to 31.
    1. The non-secure DSP from above wants to send 64 bytes message to A15 using queue 5. It will write the sixteen 32-bit values starting at address 02A1 02C0h – this is the offset of register 16 (40h) in the queue 5 of the non-secure DSP proxy page. The last 32-bit write will be at address 02A1 027Ch – this is the offset of register 31 (7Ch) in the queue 5 of the non-secure DSP proxy page. Writing to the last register will push the message into queue 5.
    2. The secure ARMSS from above wants to send 4 bytes message (e.g. 12345678h) to PMMC using queue 32. It will write the value 12345678h to address 02A8 1040h – this is the offset of register 16 (40h) in the queue 32 of the secure A15 proxy page. In order to push the message into queue 32, the register 31 must be written as well, so the secure A15 must write a dummy value (e.g. BADEFACEh) to address 02A8 107Ch – this is the offset of register 31 (7Ch) in the queue 32 of the secure A15 proxy page.
    3. Note: For a message that is longer than 64 bytes, the software is responsible for defining a disassemble and assemble protocol and code
6. The Message Manager has a limited capacity of 128 messages at any given time. Each message is assigned an index value from 0 to 127. After writing to register 31, the software must check and verify that there was an available index (via the MSGMGR_FREEINDEX_ERROR signal). It is software responsibility to ensure that no error occurs. If an error occurs, the message is not pushed into the queue and the process should be repeated.
7. The MSGMGR_FREEINDEX_ERROR signal can be set by any of the writing masters. In order to identify which master write set the MSGMGR_FREEINDEX_ERROR interrupt, the system must use a hardware based system semaphore. The semaphore block of this SoC has 64 semaphores and one of them should be dedicated to Message Manager operation. The algorithm of using the semaphore for sending a message is as follows:
    1. Just before writing to register 31, the master asks for a global hardware semaphore using a blocking function
    2. After the semaphore is granted, the master writes the last word of the message to register 31
    3. Next, the master reads the value of the interrupt status register into a volatile variable. The master repeats the read operation twice to give enough time for the hardware to set bit 0 in the register
    4. If bit 0 is not set, the master releases the semaphore
    5. If bit 0 is set, the master clears bit 0, releases the semaphore and resends the message one more time (that is, repeat steps *(a)* to *(e)*)
    6. A watchdog-like mechanism should indicate when error persists and interrupt the system
8. After finishing the send process, the thread must release the software semaphore

8.1.4.3 Receiving a Message

All queues in the Message Manager are pending queues. Table 8-3 shows what signal in the receiving host is set when there is a message in the corresponding queue. The software can configure the signal as an interrupt or the signal can be pulled periodically to look for messages.

1. Before starting run-time execution, the software should allocate the logical address of the RX queues in which messages may arrive. This is done by adding the RX queue offset to the base address of the proxy queue. The offset of queue N from the base address of the proxy is 80h*N. Examples:
    1. The non-secure DSP received queues are 4, 36, 48 and 56. The software must calculate the base

address of the proxy queue for each queue that may contain a message. For queue number 36, the base address of the proxy queue is: (80h*36 + 02A1 0000h = 02A1 1200h).

2. The non-secure A15 received queues are 5, 37, 49 and 57. The software must calculate the base address of the proxy queue for each queue that may contain a message. For queue number 37, the base address of the proxy queue is: (80h*37 + 02A2 0000h = 02A2 1280h).

2. As mentioned above, each receiver can use a polling scheme or interrupt based scheme to access the proxy queue
3. The master is ready to read the message from the proxy queue. However, the proxy queues are not protected from multiple read from the same proxy, and multiple readers (or reader and writer simultaneously) can cause conflicts. Thus, systems with multiple threads or multiple tasks must protect the proxy queue read operation when one thread (or task) starts reading and prevents other threads from accessing the proxy queue until the read operation is done. The software must define software semaphore or similar lock mechanism to ensure autonomous read operation from a proxy queue.
4. After the reader master secures the semaphore (or any other lock mechanism) for the proxy queue, it can start reading the message. The message is up to 64 bytes long and the reader must read the last 4 bytes in order to free the message and delete it from the queue. The reader must first read one or more registers (protocol dependent) starting in register 16, and then read register 31. After register 31 is read, the hardware will pop the message as described in Section 8.1.3.3.2. The offset addresses for reading are the same as described above (Section 8.1.4.2; step #2)
5. After the reader master reads register 31, it must release the software semaphore

8.1.5 Message Manager Registers

This section describes the memory-mapped registers associated with the Message Manager.

8.1.5.1 Message Manager Memory Regions

Table 8-10 lists the base address and block size for each of the Message Manager memory regions.

Table 8-10. Message Manager Memory Regions

Memory Region Name	Memory Region Base Address	Size
MSGMGR_PROXY_PAGE_CONFIG	0288 0000h	128KB
MSGMGR_PROXY_CONFIG	028A 0000h	128KB
MSGMGR_PROXY_GLOBAL_CONFIG	028C 0000h	4KB
MSGMGR_GLOBAL_CONFIG	028C 1000h	4KB
MSGMGR_LINKING_RAM_DEBUG	028C 2000h	4KB
MSGMGR_ECC_AGGR	028C 3000h	1KB
MSGMGR_QUEUE_STATE_DEBUG	028C 3400h	512B
MSGMGR_QUEUE_PROXY	02A0 0000h	2MB

8.1.5.2 MSGMGR_PROXY_PAGE_CONFIG Registers

Table 8-11 lists the memory-mapped registers for the MSGMGR_PROXY_PAGE_CONFIG region. All register offset addresses not listed in Table 8-11 should be considered as reserved locations and the register contents should not be modified.

Table 8-11. MSGMGR_PROXY_PAGE_CONFIG Registers

Offset	Acronym	Register Name	Physical Address	Section
0h	PAGE_PROXY_CFG_REG_0_0	Page 0 Proxy 0 Map Registers	0288 0000h	Section 8.1.5.2.1
1000h	PAGE_PROXY_CFG_REG_1_0	Page 1 Proxy 0 Map Registers	0288 1000h	Section 8.1.5.2.1
2000h to 1F000h	PAGE_PROXY_CFG_REG_2_0 to PAGE_PROXY_CFG_REG_31_0	Page 2–31 Proxy 0 Map Registers	0288 2000h to 0289 F000h	Section 8.1.5.2.1

8.1.5.2.1 PAGE_PROXY_CFG_REG_0_0 to PAGE_PROXY_CFG_REG_31_0 Register (Offset = Table 8-11) [reset = 0h]

PAGE_PROXY_CFG_REG_0_0 to PAGE_PROXY_CFG_REG_31_0 is shown in Figure 8-4 and described in Table 8-12.

This register holds the mapping of virtual proxy Q (0–31) in page N (0–31) to a physical proxy. The current configuration is such that each page is given a single proxy, which implies the following:

- Only virtual proxy 0 is used for each page; virtual proxies 1–31 are not used
- The mapping between virtual and physical proxies is as follows:
  - Virtual proxy 0 in page 0 --> physical proxy 0;
  - Virtual proxy 0 in page 1 --> physical proxy 1;
  - ...
  - Virtual proxy 0 in page 31 --> physical proxy 31;

Figure 8-4. PAGE_PROXY_CFG_REG_0_0 to PAGE_PROXY_CFG_REG_31_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PHYSICAL_PROXY															
R-0h																R-N															

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-12. PAGE_PROXY_CFG_REG_0_0 to PAGE_PROXY_CFG_REG_31_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	PHYSICAL_PROXY	R	N	Physical proxy number for virtual proxy 0 (Virtual-to-physical mapping disabled; always N)

8.1.5.3 MSGMGR_PROXY_CONFIG Registers

Table 8-13 lists the memory-mapped registers for the MSGMGR_PROXY_CONFIG region. All register offset addresses not listed in Table 8-13 should be considered as reserved locations and the register contents should not be modified.

Table 8-13. MSGMGR_PROXY_CONFIG Registers

Offset	Acronym	Register Name	Physical Address	Section
0h to FCh	PROXY_QUEUE_CFG_REG_0_0 to PROXY_QUEUE_CFG_REG_0_63	Proxy 0 Queue 0–63 Config Registers	028A 0000h to 028A 00FCh	Section 8.1.5.3.1
0x1000 to 0x1FFFC	PROXY_QUEUE_CFG_REG_1_0 to PROXY_QUEUE_CFG_REG_31_63	Proxy 1-31 Queue 0–63 Config Registers	028A 1000h to 028B FFFCh	Section 8.1.5.3.1

8.1.5.3.1 PROXY_QUEUE_CFG_REG_0_0 to PROXY_QUEUE_CFG_REG_31_63 Register (Offset = Table 8-13) [reset = 0h]

PROXY_QUEUE_CFG_REG_0_0 to PROXY_QUEUE_CFG_REG_31_63 is shown in Figure 8-5 and described in Table 8-14.

This register holds the access permissions for Proxy 0–31, Queue 0–63. Disabling both the write and read permissions will disable the queue access for this proxy.

This register does not support byte enables and must be written as a full 32-bit access.

Figure 8-5. PROXY_QUEUE_CFG_REG_0_0 to PROXY_QUEUE_CFG_REG_31_63 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						R	W	PHYSICAL_QNUM																							
R-0h						R/W-1h	R/W-1h	R-Q																							

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-14. PROXY_QUEUE_CFG_REG_0_0 to PROXY_QUEUE_CFG_REG_31_63 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	R	R/W	1h	Queue read permission
24	W	R/W	1h	Queue write permission
23-0	PHYSICAL_QNUM	R	Q	Physical queue number for virtual queue Q (Virtual-to-physical mapping disabled; always Q)

8.1.5.4 MSGMGR_PROXY_GLOBAL_CONFIG Registers

Table 8-15 lists the memory-mapped registers for the MSGMGR_PROXY_GLOBAL_CONFIG region. All register offset addresses not listed in should be considered as reserved locations and the register contents should not be modified.

Table 8-15. MSGMGR_PROXY_GLOBAL_CONFIG Registers

Offset	Acronym	Register Name	Physical Address	Section
10h	INTR_RAW_STATUS_SET_REG	Interrupt Raw Status/Set Register	028C 0010h	Section 8.1.5.4.1
18h	INTR_ENABLED_STATUS_SET_REG	Interrupt Enabled Status/Set Register	028C 0018h	Section 8.1.5.4.2
20h	INTR_ENABLE_REG	Interrupt Enable Register	028C 0020h	Section 8.1.5.4.3
28h	INTR_CLEAR_REG	Interrupt Clear Register	028C 0028h	Section 8.1.5.4.4
30h	EOI_REG	End of Interrupt (EOI) Register	028C 0030h	Section 8.1.5.4.5
40h	PROXY_ERROR_STATUS_REG	Proxy Error Status Register	028C 0040h	Section 8.1.5.4.6

8.1.5.4.1 INTR_RAW_STATUS_SET_REG Register (Offset = 10h) [reset = 0h]

INTR_RAW_STATUS_SET_REG is shown in Figure 8-6 and described in Table 8-16.

The Interrupt Raw Status/Set register shows the raw interrupt status before enabling and allows setting the interrupt status. Each bit corresponds to an error associated with a given proxy.

This register is useful for software debugging.

Figure 8-6. INTR_RAW_STATUS_SET_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PROXY_ERROR																															
R/W1S-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-16. INTR_RAW_STATUS_SET_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PROXY_ERROR	R/W1S	0h	Each bit indicates an error associated with a given proxy. (Bit position [0] corresponds to proxy 0, bit position [1] corresponds to proxy 1, etc.) Write '1' to set the status for the associated proxy. Writing '0' has no effect.

8.1.5.4.2 INTR_ENABLED_STATUS_SET_REG Register (Offset = 18h) [reset = 0h]

INTR_ENABLED_STATUS_SET_REG is shown in Figure 8-7 and described in Table 8-17.

The Interrupt Enabled Status/Set register shows the interrupt enabled status and allows clearing the interrupt status. Each bit corresponds to an error associated with a given proxy.

Figure 8-7. INTR_ENABLED_STATUS_SET_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PROXY_ERROR																															
R/W1C-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-17. INTR_ENABLED_STATUS_SET_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PROXY_ERROR	R/W1C	0h	Each bit indicates an error associated with a given proxy. (Bit position [0] corresponds to proxy 0, bit position [1] corresponds to proxy 1, etc.) Write '1' to clear the status for the associated proxy. Writing '0' has no effect.

8.1.5.4.3 INTR_ENABLE_REG Register (Offset = 20h) [reset = 0h]

INTR_ENABLE_REG is shown in Figure 8-8 and described in Table 8-18.

The Interrupt Enable register allows setting the interrupt enable (that is, enable interrupt generation). Each bit corresponds to an error associated with a given proxy.

Figure 8-8. INTR_ENABLE_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PROXY_ERROR																															
R/W1S-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-18. INTR_ENABLE_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PROXY_ERROR	R/W1S	0h	Each bit corresponds to an error associated with a given proxy. (Bit position [0] corresponds to proxy 0, bit position [1] corresponds to proxy 1, etc.) Write '1' to set the enable for the associated proxy.

8.1.5.4.4 INTR_CLEAR_REG Register (Offset = 28h) [reset = 0h]

INTR_CLEAR_REG is shown in Figure 8-9 and described in Table 8-19.

The Interrupt Clear register allows clearing the interrupt enable (that is, disable interrupt generation). Each bit corresponds to an error associated with a given proxy.

Figure 8-9. INTR_CLEAR_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PROXY_ERROR																															
R/W1C-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-19. INTR_CLEAR_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PROXY_ERROR	R/W1C	0h	Each bit corresponds to an error associated with a given proxy. (Bit position [0] corresponds to proxy 0, bit position [1] corresponds to proxy 1, etc.) Write '1' to clear the enable for the associated proxy.

8.1.5.4.5 EOI_REG Register (Offset = 30h) [reset = 0h]

EOI_REG is shown in Figure 8-10 and described in Table 8-20.

The EOI register is used by software to write a vector associated with each interrupt to indicate that the interrupt has been serviced.

Figure 8-10. EOI_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								EOI_VECTOR							
R-0h																								R/W-0h							

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-20. EOI_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	EOI_VECTOR	R/W	0h	EOI vector value associated with each interrupt. Use the bit position of the interrupt that was cleared as this value.

8.1.5.4.6 PROXY_ERROR_STATUS_REG Register (Offset = 50h) [reset = 0h]

PROXY_ERROR_STATUS_REG is shown in Figure 8-11 and described in Table 8-21.

The Proxy Error Status register holds the proxy number and queue number for the first transfer that resulted in a proxy error.

Figure 8-11. PROXY_ERROR_STATUS_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PROXY_NUM								QUEUE_NUM							
R-0h								R-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUEUE_NUM															
R-0h															

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-21. PROXY_ERROR_STATUS_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	PROXY_NUM	R	0h	Indicates the proxy number associated with the queue access resulting in a proxy error • 0h = Proxy 0; • 1h = Proxy 1; • ... • 7Ch = Proxy 31; • 80h to FFh = Reserved;
23-0	QUEUE_NUM	R	0h	Indicates the queue number associated with the queue access resulting in a proxy error • 0h = Queue 0; • 1h = Queue 1; • ... • FCh = Queue 63; • 100h to FFFFFFh = Reserved;

8.1.5.5 MSGMGR_GLOBAL_CONFIG Registers

lists the memory-mapped registers for the MSGMGR_GLOBAL_CONFIG region. All register offset addresses not listed in should be considered as reserved locations and the register contents should not be modified.

Table 8-22. MSGMGR_GLOBAL_CONFIG Registers

Offset	Acronym	Register Name	Physical Address	Section
0h	REVISION_REG	Revision Register	028C 1000h	Section 11.13.5.2.3.1.1
10h	INTR_RAW_STATUS_SET_REG	Interrupt Raw Status/Set Register	028C 1010h	Section 8.1.5.5.2
14h	INTR_ENABLED_STATUS_SET_REG	Interrupt Enabled Status/Set Register	028C 1014h	Section 8.1.5.5.3
18h	INTR_ENABLE_REG	Interrupt Enable Register	028C 1018h	Section 8.1.5.5.4
1Ch	INTR_CLEAR_REG	Interrupt Clear Register	028C 101Ch	Section 8.1.5.5.5
20h	EOI_REG	End of Interrupt (EOI) Register	028C 1020h	Section 8.1.5.5.6

8.1.5.5.1 REVISION_REG Register (Offset = 0h) [reset = 0h]

REVISION_REG is shown in Figure 11-1055 and described in Table 11-2446.

The Revision register provides revision details of the module.

Figure 8-12. REVISION_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-4E641900h																															

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-23. REVISION_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4E641900	TI internal data. Identifies revision of peripheral.

8.1.5.5.2 INTR_RAW_STATUS_SET_REG Register (Offset = 10h) [reset = 0h]

INTR_RAW_STATUS_SET_REG is shown in Figure 8-13 and described in Table 8-24.

The Interrupt Raw Status/Set register shows the raw interrupt status before enabling and allows setting the interrupt status. Bit position [0] indicates that there is no free index available. This should only happen if more than 128 messages are queued.

This register is useful for software debugging.

Figure 8-13. INTR_RAW_STATUS_SET_REG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							FREE_INDEX_ERROR
R-0h							R/W1S-0h

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-24. INTR_RAW_STATUS_SET_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	FREE_INDEX_ERROR	R/W1S	0h	Indicates that there is no free index available. Write '1' to set the status. Writing '0' has no effect.

8.1.5.5.3 INTR_ENABLED_STATUS_SET_REG Register (Offset = 14h) [reset = 0h]

INTR_ENABLED_STATUS_SET_REG is shown in Figure 8-14 and described in Table 8-25.

The Interrupt Enabled Status/Set register shows the interrupt enabled status and allows clearing the interrupt status. Bit position [0] indicates that there is no free index available. This should only happen if more than 128 messages are queued.

Figure 8-14. INTR_ENABLED_STATUS_SET_REG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							FREE_INDEX_ERROR
R-0h							R/W1C-0h

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-25. INTR_ENABLED_STATUS_SET_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	FREE_INDEX_ERROR	R/W1C	0h	Write '1' to clear the status. Writing '0' has no effect.

8.1.5.5.4 INTR_ENABLE_REG Register (Offset = 18h) [reset = 0h]

INTR_ENABLE_REG is shown in Figure 8-15 and described in Table 8-26.

The Interrupt Enable register allows setting the interrupt enable (that is, enable interrupt generation). Bit position [0] corresponds to the free index error interrupt.

Figure 8-15. INTR_ENABLE_REG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							FREE_INDEX_ERROR
R-0h							R/W1S-0h

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-26. INTR_ENABLE_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	FREE_INDEX_ERROR	R/W1S	0h	Write '1' to set the enable

8.1.5.5.5 INTR_CLEAR_REG Register (Offset = 1Ch) [reset = 0h]

INTR_CLEAR_REG is shown in Figure 8-16 and described in Table 8-27.

The Interrupt Clear register allows clearing the interrupt enable (that is, disable interrupt generation). Bit position [0] corresponds to the free index error interrupt.

Figure 8-16. INTR_CLEAR_REG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							FREE_INDEX_ERROR
R-0h							R/W1C-0h

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-27. INTR_CLEAR_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	FREE_INDEX_ERROR	R/W1C	0h	Write '1' to clear the enable

8.1.5.5.6 EOI_REG Register (Offset = 20h) [reset = 0h]

EOI_REG is shown in Figure 8-17 and described in Table 8-28.

The EOI register is used by software to write a vector associated with each interrupt to indicate that the interrupt has been serviced.

Figure 8-17. EOI_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								EOI_VECTOR							
R-0h																								R/W-0h							

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-28. EOI_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	EOI_VECTOR	R/W	0h	EOI vector value associated with each interrupt. Use the bit position of the interrupt that was cleared as this value

8.1.5.6 MSGMGR_LINKING_RAM_DEBUG Registers

Table 8-29 lists the memory-mapped registers for the MSGMGR_LINKING_RAM_DEBUG region. All register offset addresses not listed in Table 8-29 should be considered as reserved locations and the register contents should not be modified.

Table 8-29. MSGMGR_LINKING_RAM_DEBUG Registers

Offset	Acronym	Register Name	Physical Address	Section
0h to 1FCh	NEXT_INDEX_REG_0 to NEXT_INDEX_REG_127	Linking RAM contents from 0 to 127	028C 2000h to 028C 21FCh	Section 8.1.5.6.1

8.1.5.6.1 NEXT_INDEX_REG_0 to NEXT_INDEX_REG_127 Register (Offset = 0h to 1FCh) [reset = 0h]

NEXT_INDEX_REG_0 to NEXT_INDEX_REG_127 is shown in Figure 8-18 and described in Table 8-30.

This register holds the contexts (next index) of Linking RAM Entry 0 to 127.

This register is intended to be used for debug purpose.

Figure 8-18. NEXT_INDEX_REG_0 to NEXT_INDEX_REG_127 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				NEXT_INDEX											
R-0h																				R-0h											

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-30. NEXT_INDEX_REG_0 to NEXT_INDEX_REG_127 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11-0	NEXT_INDEX	R	0h	Next linking index

8.1.5.7 MSGMGR_QUEUE_STATE_DEBUG Registers

Table 8-31 lists the memory-mapped registers for the MSGMGR_QUEUE_STATE_DEBUG region. All register offset addresses not listed in Table 8-31 should be considered as reserved locations and the register contents should not be modified.

Table 8-31. MSGMGR_QUEUE_STATE_DEBUG Registers

Offset	Acronym	Register Name	Physical Address	Section
0h to FCh	INDEX_REG_0 to INDEX_REG_63	Head index of Queues 0–63	028C 3400h to 028C 34FCh	Section 8.1.5.7.1

8.1.5.7.1 INDEX_REG_0 to INDEX_REG_63 Register (Offset = 0h to FCh) [reset = 0h]

INDEX_REG_0 to INDEX_REG_63 is shown in Figure 8-19 and described in Table 8-32.

This register holds the head index and entry count of Queues 0–63.

This register is intended to be used for debug purpose.

Figure 8-19. INDEX_REG_0 to INDEX_REG_63 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								ENTRY_COUNT												HEAD_INDEX											
R-0h								R-0h												R-0h											

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-32. INDEX_REG_0 to INDEX_REG_63 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-12	ENTRY_COUNT	R	0h	Entry count
11-0	HEAD_INDEX	R	0h	Head index

8.1.5.8 MSGMGR_ECC_AGGR Registers

lists the memory-mapped registers for the MSGMGR_ECC_AGGR region. All register offset addresses not listed in should be considered as reserved locations and the register contents should not be modified.

Table 8-33. MSGMGR_ECC_AGGR Registers

Offset	Acronym	Register Name	Physical Address	Section
0h	ECC_REVISION	ECC Revision Register	028C 3000h	Section 8.1.5.8.1
8h	ECC_VECTOR	ECC Vector Register	028C 3008h	Section 8.1.5.8.2
Ch	ECC_MISC_STATUS	ECC Miscellaneous Status Register	028C 300Ch	Section 8.1.5.8.3
10h	ECC_WRAPPER_REVISION	ECC Revision Wrapper Register	028C 3010h	Section 8.1.5.8.4
14h	ECC_CONTROL	ECC Control Register	028C 3014h	Section 8.1.5.8.5
18h	ECC_ERROR_CONTROL1	ECC Control 1 Register	028C 3018h	Section 8.1.5.8.6
1Ch	ECC_ERROR_CONTROL2	ECC Control 2 Register	028C 301Ch	Section 8.1.5.8.7
20h	ECC_ERROR_STATUS1	ECC Status 1 Register	028C 3020h	Section 8.1.5.8.8
24h	ECC_ERROR_STATUS2	ECC Status 2 Register	028C 3024h	Section 8.1.5.8.9
3Ch	ECC_EOI	ECC EOI Register	028C 303Ch	Section 8.1.5.8.10
40h to 7Ch	ECC_INT_STATUS_0 to ECC_INT_STATUS_15	ECC Interrupt Status Registers	028C 3040h to 028C 307Ch	Section 8.1.5.8.11
80h to BCh	ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15	ECC Interrupt Enable Registers	028C 3080h to 028C 30BCh	Section 8.1.5.8.12
C0h to FCh	ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15	ECC Interrupt Clear Registers	028C 30C0h to 028C 30FCh	Section 8.1.5.8.13

8.1.5.8.1 ECC_REVISION Register (Offset = 0h) [reset = 4E100001h]

ECC_REVISION is shown in Figure 8-20 and described in Table 8-34.

Figure 8-20. ECC_REVISION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-4E100001h																															

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-34. ECC_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4E100001h	TI internal data. Identifies revision of peripheral.

8.1.5.8.2 ECC_VECTOR Register (Offset = 8h) [reset = 0h]

ECC_VECTOR is shown in Figure 8-21 and described in Table 8-35.

Figure 8-21. ECC_VECTOR Register

31	30	29	28	27	26	25	24
RESERVED							READ_DONE
R-0h							R-0h
23	22	21	20	19	18	17	16
READ_ADDRESS							
R/W-0h							
15	14	13	12	11	10	9	8
TRIGGER_READ	RESERVED					RAM_ID	
R/W-0h	R-0h					R/W-0h	
7	6	5	4	3	2	1	0
RAM_ID							
R/W-0h							

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; *-n* = value after reset

Table 8-35. ECC_VECTOR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24	READ_DONE	R	0h	Status indicating that the read is complete.
23-16	READ_ADDRESS	R/W	0h	Read address: Can be any of the registers (10h to 24h).
15	TRIGGER_READ	R/W	0h	Trigger a read operation to the specified read address.
14-11	RESERVED	R	0h	Reserved
10-0	RAM_ID	R/W	0h	ECC RAM ID to select which ECC RAM to control or read status from.

8.1.5.8.3 ECC_MISC_STATUS Register (Offset = Ch) [reset = 4h]

ECC_MISC_STATUS is shown in Figure 8-22 and described in Table 8-36.

Figure 8-22. ECC_MISC_STATUS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					NUM_RAMS										
R-0h																					R-4h										

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-36. ECC_MISC_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Reserved
10-0	NUM_RAMS	R	4h	Number of ECC RAMs serviced by the aggregator.

8.1.5.8.4 ECC_WRAPPER_REVISION Register (Offset = 10h) [reset = 4E100001h]

ECC_WRAPPER_REVISION is shown in Figure 8-23 and described in Table 8-37.

Figure 8-23. ECC_WRAPPER_REVISION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-4E100001h																															

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-37. ECC_WRAPPER_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4E100001h	TI internal data. Identifies revision of peripheral.

8.1.5.8.5 ECC_CONTROL Register (Offset = 14h) [reset = 7h]

ECC_CONTROL is shown in Figure 8-24 and described in Table 8-38.

Figure 8-24. ECC_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	ERROR_ONCE	FORCE_N _ROW	FORCE_DED	FORCE_SEC	ENABLE_RMW	ECC_CHECK	ECC_ENABLE
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h	R/W-1h	R/W-1h

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; *-n* = value after reset

Table 8-38. ECC_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	ERROR_ONCE	R/W	0h	If this bit is set, the FORCE_SEC/FORCE_DED will inject an error to the specified row only once. The FORCE_SEC bit will be cleared once a writeback happens. If writeback is not enabled, this error will be cleared the cycle following the read when the data is corrected. For double-bit errors, the FORCE_DED bit will be cleared the cycle following the double-bit error. Any subsequent reads will not force an error.
5	FORCE_N_ROW	R/W	0h	Force single/double-bit error on the next RAM read. Note: Takes effect only when ECC_ENABLE is set.
4	FORCE_DED	R/W	0h	Force double-bit error. Cleared the cycle following the error if ERROR_ONCE is asserted. Note: Takes effect only when ECC_ENABLE is set.
3	FORCE_SEC	R/W	0h	Force single-bit error. Cleared on a writeback or the cycle following the error if ERROR_ONCE is asserted. Note: Takes effect only when ECC_ENABLE is set.
2	ENABLE_RMW	R/W	1h	Enable read-modify-write on partial word writes.
1	ECC_CHECK	R/W	1h	Enable ECC check. ECC is completely bypassed if both ECC_ENABLE and ECC_CHECK are 0.
0	ECC_ENABLE	R/W	1h	Enable ECC generation.

8.1.5.8.6 ECC_ERROR_CONTROL1 Register (Offset = 18h) [reset = 0h]

ECC_ERROR_CONTROL1 is shown in Figure 8-25 and described in Table 8-39.

Figure 8-25. ECC_ERROR_CONTROL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_BIT1																ECC_ROW															
R/W-0h																R/W-0h															

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; *-n* = value after reset

Table 8-39. ECC_ERROR_CONTROL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	ECC_BIT1	R/W	0h	Column/Data bit that needs to be flipped when FORCE_SEC or FORCE_DED is set.
15-0	ECC_ROW	R/W	0h	Row address where FORCE_SEC or FORCE_DED needs to be applied. This is ignored if FORCE_N_ROW is set.

8.1.5.8.7 ECC_ERROR_CONTROL2 Register (Offset = 1Ch) [reset = 0h]

ECC_ERROR_CONTROL2 is shown in Figure 8-26 and described in Table 8-40.

Figure 8-26. ECC_ERROR_CONTROL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ECC_BIT2															
R-0h																R/W-0h															

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; *-n* = value after reset

Table 8-40. ECC_ERROR_CONTROL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	ECC_BIT2	R/W	0h	Data bit that needs to be flipped when FORCE_DED is set.

8.1.5.8.8 ECC_ERROR_STATUS1 Register (Offset = 20h) [reset = 0h]

ECC_ERROR_STATUS1 is shown in Figure 8-27 and described in Table 8-41.

Figure 8-27. ECC_ERROR_STATUS1 Register

31	30	29	28	27	26	25	24
ECC_ROW							
R-0h							
23	22	21	20	19	18	17	16
ECC_ROW							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					CLR_ECC_OTHER	CLR_ECC_DED	CLR_ECC_SEC
R-0h					R/W1C-0h	R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
RESERVED					ECC_OTHER	ECC_DED	ECC_SEC
R-0h					R/W1S-0h	R/W1S-0h	R/W1S-0h

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; *-n* = value after reset

Table 8-41. ECC_ERROR_STATUS1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	ECC_ROW	R	0h	Indicates the row/address where the single or double-bit error occurred.
15-11	RESERVED	R	0h	Reserved
10	CLR_ECC_OTHER	R/W1C	0h	'1' indicates a successive single-bit error. Writing a '1' clears the status bit.
9	CLR_ECC_DED	R/W1C	0h	'1' indicates a pending double-bit error. Writing a '1' clears the status bit.
8	CLR_ECC_SEC	R/W1C	0h	'1' indicates a pending single-bit error. Writing a '1' clears the status bit.
7-3	RESERVED	R	0h	Reserved
2	ECC_OTHER	R/W1S	0h	'1' – Indicates that successive single-bit errors have occurred while a writeback is still pending. The software can also write a '1' to set the pending status and write a '1' to the corresponding clear bit to clear the status.
1	ECC_DED	R/W1S	0h	'1' – Indicates pending double-bit error status Since the double-bit error from the ECC logic is a pulsed interrupt, this is also a status set register. The software can also write a '1' to set the pending status and write a '1' to the corresponding clear bit to clear the status.
0	ECC_SEC	R/W1S	0h	'1' – Indicates pending single-bit error status Since the single-bit error from the ECC logic is a pulsed interrupt, this is also a status set register. The software can also write a '1' to set the pending status and write a '1' to the corresponding clear bit to clear the status.

8.1.5.8.9 ECC_ERROR_STATUS2 Register (Offset = 24h) [reset = 0h]

ECC_ERROR_STATUS2 is shown in Figure 8-28 and described in Table 8-42.

Figure 8-28. ECC_ERROR_STATUS2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ECC_BIT1															
R-0h																R-0h															

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; *-n* = value after reset

Table 8-42. ECC_ERROR_STATUS2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	ECC_BIT1	R	0h	Indicates the bit position in the ram data that is in error. For example: a value of 1 indicates that bit 1 in the data is in error. This is valid only for single bit errors (SEC).

8.1.5.8.10 ECC_EOI Register (Offset = 3Ch) [reset = 0h]

ECC_EOI is shown in Figure 8-29 and described in Table 8-43.

Figure 8-29. ECC_EOI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EOI_WR
R-0h							W1S

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-43. ECC_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EOI_WR	W1S	0h	Write to this register indicates that software has acknowledged the pending interrupt and the next interrupt can be sent to the host.

8.1.5.8.11 ECC_INT_STATUS_0 to ECC_INT_STATUS_15 Register (Offset = 40h to 7Ch) [reset = 0h]

ECC_INT_STATUS_0 to ECC_INT_STATUS_15 is shown in Figure 8-30 and described in Table 8-44.

Raw interrupt status from each of the ECC RAMs. Each bit corresponds to a level pending status from an ECC RAM.

These are the raw level interrupt status bits where each bit corresponds to the pending status from an ECC RAM. The bit associations with the ECC RAMs are assigned by the ECC aggregator module. There is 1 register for upto 32 ECC RAMs. The addresses increment for every additional 32-bit register required to hold the interrupt status for all ECC RAMs.

Figure 8-30. ECC_INT_STATUS_0 to ECC_INT_STATUS_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITMASK																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-44. ECC_INT_STATUS_0 to ECC_INT_STATUS_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITMASK	R	0h	Level interrupt status from each ECC RAM. 1 = Pending 0 = Not pending

8.1.5.8.12 ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15 Register (Offset = 80h to BCh) [reset = 0h]

ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15 is shown in Figure 8-31 and described in Table 8-45.

Interrupt Enable Set Registers.

These are interrupt enables associated with the interrupt from each of the ECC RAMs. Writing a '1' to a bit position in the register enables the interrupt from the associated ECC RAM. There is 1 register for upto 32 ECC RAMs. The addresses increment for every additional 32-bit register required for all ECC RAMs.

Figure 8-31. ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITMASK																															
W1S-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-45. ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITMASK	W1S	0h	Write '1' to enable interrupt from the associated ECC RAM.

8.1.5.8.13 ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15 Register (Offset = C0h to FCh) [reset = 0h]

ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15 is shown in Figure 8-32 and described in Table 8-46.

Interrupt Enable Clear Registers.

These are interrupt enable clear bits associated with the interrupt from each of the ECC RAMs. Writing a '1' to a bit position in the register disables the interrupt from the associated ECC RAM. There is 1 register for upto 32 ECC RAMs. The addresses increment for every additional 32-bit register required for all the ECC RAMs (0 to N-1).

Figure 8-32. ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITMASK																															
W1C-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-46. ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITMASK	W1C	0h	Write '1' to disable interrupt from the associated ECC RAM.

8.1.5.9 MSGMGR_QUEUE_PROXY Registers

Table 8-47 lists the memory-mapped registers for the MSGMGR_QUEUE_PROXY region. All register offset addresses not listed in Table 8-47 should be considered as reserved locations and the register contents should not be modified.

Table 8-47. MSGMGR_QUEUE_PROXY Registers

Offset	Acronym	Register Name	Physical Address	Section
0h to 7Ch	QUEUE_DATA_REG_0_0_0 to QUEUE_DATA_REG_0_0_31	Page 0 Proxy 0 Queue 0 Data Registers 0–31	02A0 0000h to 02A0 007Ch	Section 8.1.5.9.1
80h to 1FFCh	QUEUE_DATA_REG_0_1_0 to QUEUE_DATA_REG_0_63_31	Page 0 Proxy 0 Queue 1–63 Data Registers 0-31	02A0 0080h to 02A0 1FFCh	Section 8.1.5.9.1
10000h to 1FFFFCh	QUEUE_DATA_REG_1_0_0 to QUEUE_DATA_REG_31_63_31	Page 1-31 Proxy 0 Queues 32–63 Data Registers 0-31	02A1 0000h to 02BF FFFCh	Section 8.1.5.9.1

8.1.5.9.1 QUEUE_DATA_REG_0_0_0 to QUEUE_DATA_REG_31_64_31 Register (Offset = Table 8-47) [reset = 0h]

QUEUE_DATA_REG_0_0_0 to QUEUE_DATA_REG_31_64_31 is shown in Figure 8-33 and described in Table 8-48.

The Queue Data Register holds the contents (queue/message data) for Page 0–31, Queue 0–63, Entry 0–31.

Figure 8-33. QUEUE_DATA_REG_0_0_0 to QUEUE_DATA_REG_31_64_31 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1C = Write 1 to clear bit; W1S = Write 1 to set bit; -*n* = value after reset

Table 8-48. QUEUE_DATA_REG_0_0_0 to QUEUE_DATA_REG_31_64_31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	RW	0h	Queue/Message data

8.2 Semaphore Module

This chapter describes the operation of the Semaphore hardware module. The Semaphore module is accessible across all the cores on a multicore environment. The module supports up to 64 independent semaphores that help the application to implement shared-resource protection mechanism across multiple cores. Each of the semaphores can be accessed by the cores in direct, indirect, or combined modes.

8.2.1 Semaphore Overview

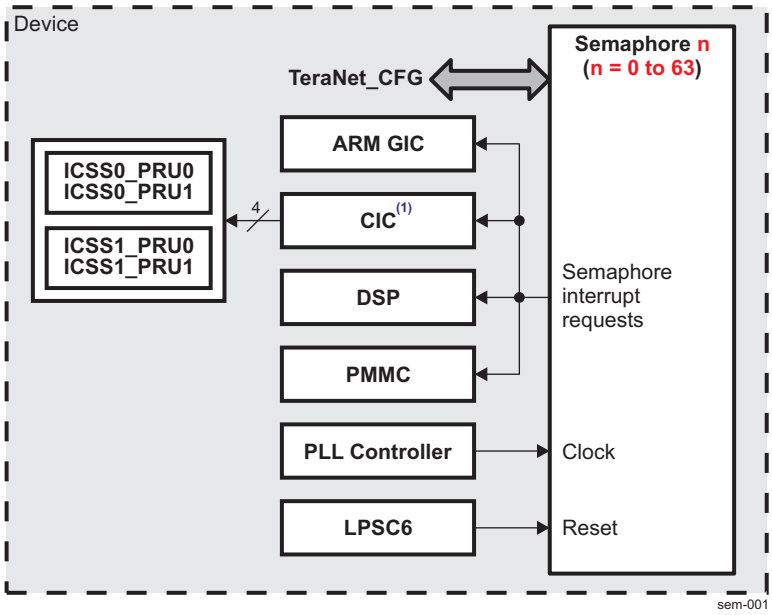
In a multicore environment where system resources must be shared it is important to control simultaneous accesses to the available resources. To ensure correct system operation, it is necessary to limit access to a resource by one and only one core at a time; that is, it is necessary to provide mutual exclusion for resources shared across multiple cores.

The Semaphore module provides a mechanism that applications can use to implement mutual exclusion of shared resources across multiple cores. The following CPU cores can be semaphore masters on this device:

- DSP C66x
- Arm Cortex-A15
- PMMC CPU
- ICSS0_PRU0
- ICSS0_PRU1
- ICSS1_PRU0
- ICSS1_PRU1

Figure 8-34 shows an overview of the Semaphore module.

Figure 8-34. Semaphore Module Overview



(1) The user may enable PRU-ICSS semaphore usage. This is possible by programming the CIC to route the appropriate events to one or more of the CIC outputs which are connected to the PRU-ICSS events inputs.

The Semaphore module supports the following features:

- Provides mutual exclusion for a shared resource
- A maximum of 16 semaphore masters (device cores)
- A maximum of 64 independent semaphores
- Semaphore request methods:
  - Direct request

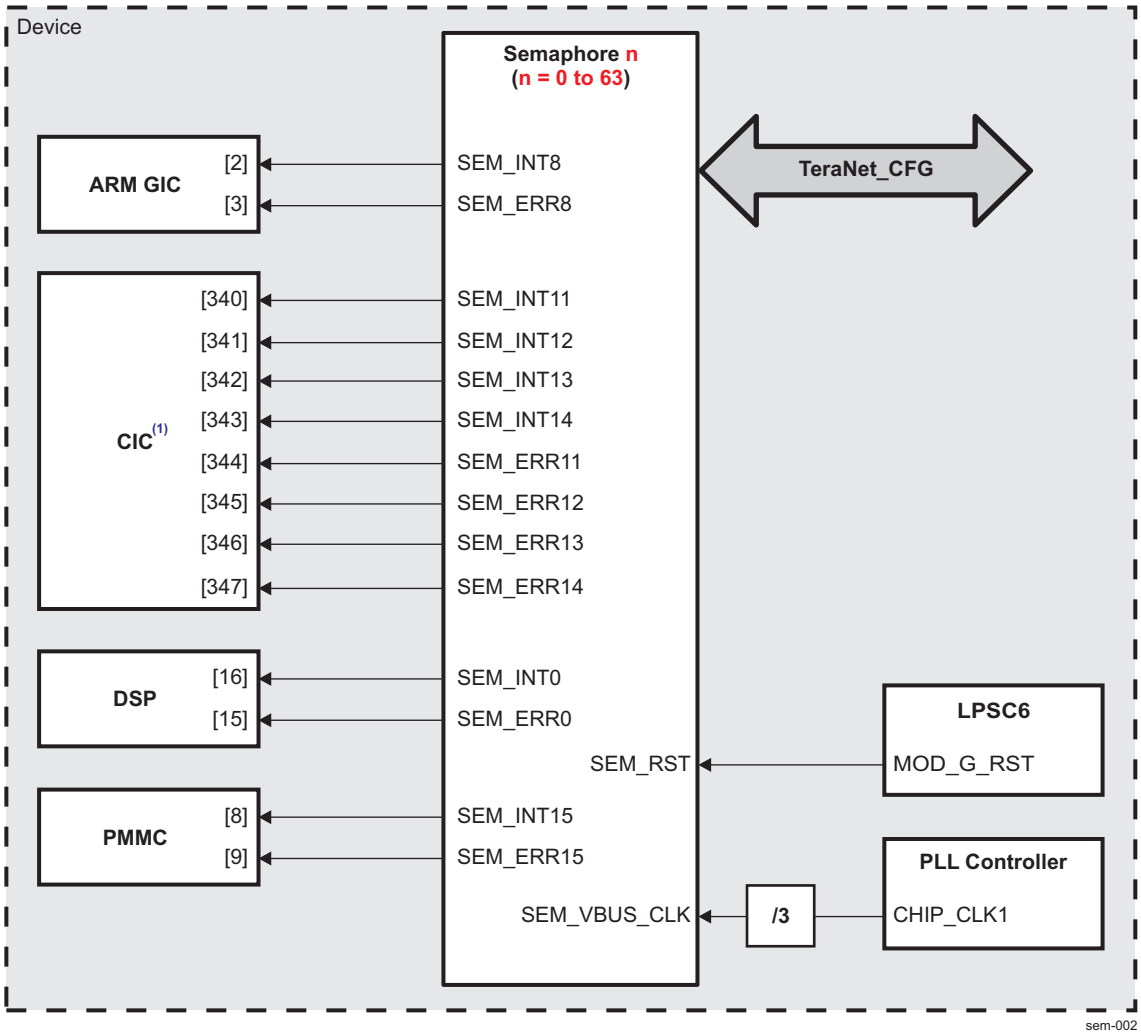
  - Indirect request
  - Combined request
- Endian independent
- Atomic semaphore access
- Lock-out mechanism for used semaphores
- Queued requests for used semaphores
- Semaphores access grant interrupt for queued requests
- Allows the application to check the status of any of the semaphores
- Error detection and interrupts

8.2.2 Semaphore Integration

This section describes the module integration in the device, including information about clocks, resets, and hardware requests.

Figure 8-35 shows the Semaphore module integration.

Figure 8-35. Semaphore Integration



(1) SEM_INT[14-11] and SEM_ERR[14-11] events are not routed directly to PRU-ICSS due to limited PRU-ICSS event availability. Instead these events are routed to the CIC. The user may enable PRU-ICSS semaphore usage by programming the CIC to route the appropriate events to one or more of the CIC outputs which are connected to the PRU-ICSS events inputs. For more information, see Chapter 9, *Interrupts*.

Table 8-49 through Table 8-51 summarize the integration of the module in the device.

Table 8-49. Semaphore Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
Semaphore	PD5	LPSC6	TeraNet_CFG

Table 8-50. Semaphore Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
Semaphore	SEM_VBUS_CLK	CHIP_CLK1 / 3	PLL Controller	Interface and Funtional Clock
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
Semaphore	SEM_RST	MOD_G_RST	LPSC6	Module Reset

Table 8-51. Semaphore Hardware Requests

Interrupt Requests						
Module Instance	**Event Name**	**Mapped To Input Event [Number]**				**Description**
		ARM GIC	**CIC**	**DSP**	**PMMC**	
Semaphore	SEM_INT0	-	-	[16]	-	Semaphore Grant Interrupt Request
	SEM_INT8	[2]	-	-	-	Semaphore Grant Interrupt Request
	SEM_INT11	-	[340]	-	-	Semaphore Grant Interrupt Request
	SEM_INT12	-	[341]	-	-	Semaphore Grant Interrupt Request
	SEM_INT13	-	[342]	-	-	Semaphore Grant Interrupt Request
	SEM_INT14	-	[343]	-	-	Semaphore Grant Interrupt Request
	SEM_INT15	-	-	-	[8]	Semaphore Grant Interrupt Request
	SEM_ERR0	-	-	[15]	-	Semaphore Error Interrupt Request
	SEM_ERR8	[3]	-	-	-	Semaphore Error Interrupt Request
	SEM_ERR11	-	[344]	-	-	Semaphore Error Interrupt Request
	SEM_ERR12	-	[345]	-	-	Semaphore Error Interrupt Request
	SEM_ERR13	-	[346]	-	-	Semaphore Error Interrupt Request
	SEM_ERR14	-	[347]	-	-	Semaphore Error Interrupt Request
	SEM_ERR15	-	-	-	[9]	Semaphore Error Interrupt Request

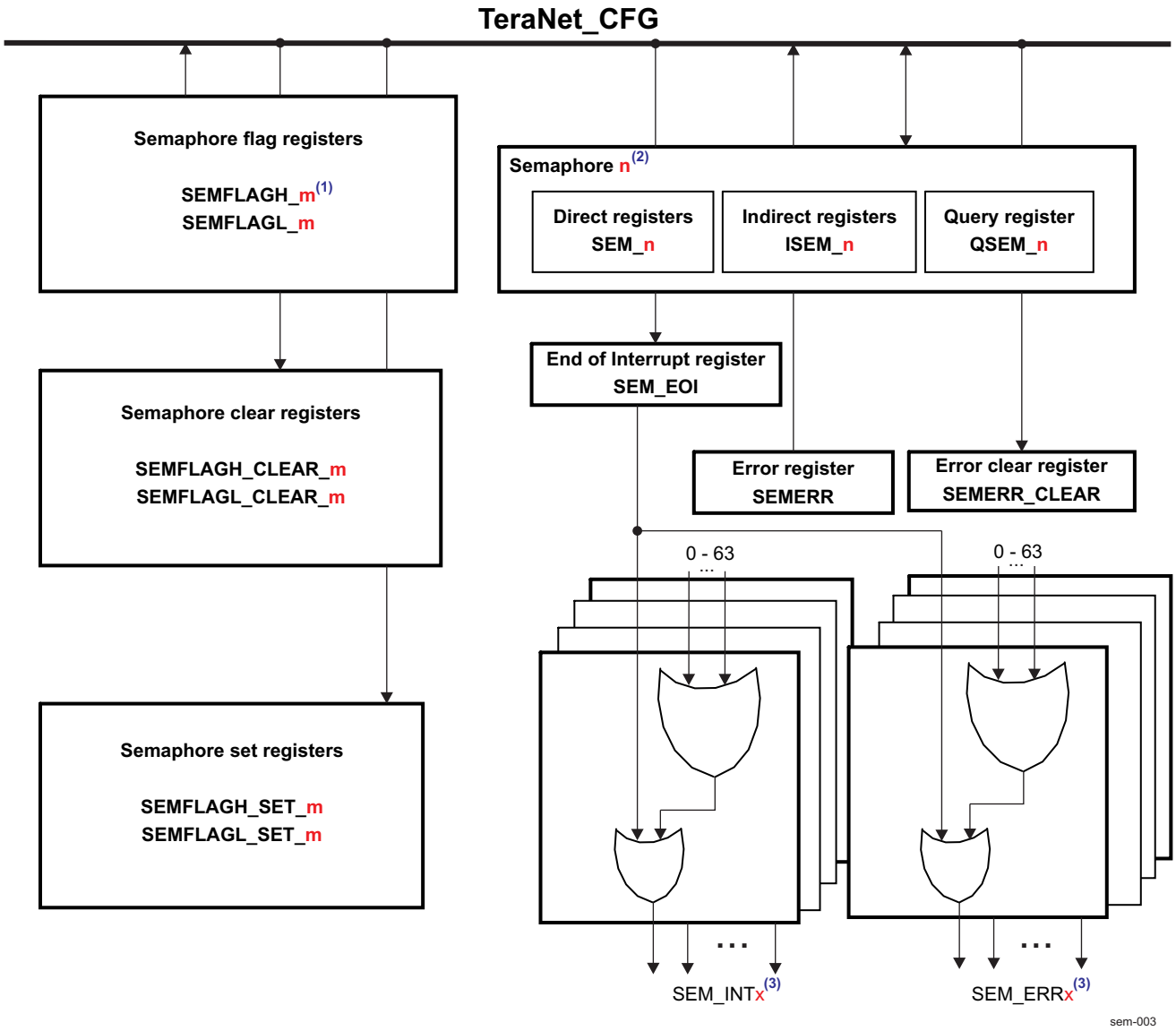
NOTE: For more information about Semaphore Grand and Error Interrupt Requests, see Section 8.2.3.3, *Interrupt Support*.

8.2.3 Semaphore Functional Description

8.2.3.1 Functional Block Diagram

Figure 8-36 shows the block diagram of the Semaphore module. The module supports up to 64 independent semaphores that can be accessed by all device cores (DSP C66x, Arm Cortex-A15, PMMC CPU, and the PRUs in the PRU-ICSS modules). The module generates two sets of interrupts: semaphore grant interrupt (SEM_INTx) and error interrupt (SEM_ERRx). In the figure, 'm' denotes the number of masters (device cores) available (maximum of 16) and 'n' denotes the number of semaphores (maximum of 64) available on a device.

Figure 8-36. Semaphore Module Block Diagram



(1) Number of Cores: m = 0 to 15

(2) Number of Semaphores: n = 0 to 63

(3) See Section 8.2.2, *Semaphore Integration*

The Semaphore module can be used as an arbiter to ensure mutual exclusivity when sharing resources over multiple cores in a multicore device. It provides up to 64 independent semaphores that can be acquired and released by the application.

Each of the semaphores is controlled by either reading from or writing to one of the three registers: SEM_n, ISEM_n, or QSEM_n. These three registers form a set and each semaphore is attached to one such set as shown in Figure 8-36. The Semaphore module can service requests using one of three methods: the direct request method, the indirect request method, or the combined request method. These three methods provide flexibility to the application to either implement a polling-based system or an interrupt-based callback mechanism for acquiring or locking a semaphore.

None of the semaphores on the module are mapped to any resource or module on the device. The application can choose to map any of the available semaphores to protect any resource. There is no support provided by the Semaphore module to store the semaphore to resource mapping information. The mapping information must be maintained by the software application.

8.2.3.2 Reset Considerations

8.2.3.2.1 Software Reset Considerations

The Semaphore module can be reset through software by writing to the SEM_RST_RUN[0] RESET bit. On reset, all semaphores are in 'FREE' state and all the errors cleared.

8.2.3.2.2 Hardware Reset Considerations

On reset, all semaphores are in 'FREE' state and all the errors cleared.

8.2.3.3 Interrupt Support

8.2.3.3.1 Interrupt Events and Requests

The Semaphore module can generate an interrupt to any of the cores on the device either to signal semaphore grant or an error. The module generates two sets of interrupts: semaphore grant (SEM_INTx) and error (SEM_ERRx) interrupts. Each set has the capability to interrupt any core m (m = 0 to 15) on the device. For more information about available semaphore masters (device cores) on this device, see Section 8.2.1, *Semaphore Overview*.

8.2.3.3.2 Error Generation and Handling

There are four types of errors the Semaphore module can generate:

- Already Free Error: This error is generated when an attempt is made to free a semaphore that is already free.
- Illegal Free Error: This error is generated when an attempt is made to free a semaphore by a core that is not the current owner.
- Already Own Error: This error is generated when a core attempts to acquire a semaphore that it already owns.
- Already Requested Error: This error is generated when a core attempts to acquire a semaphore for which it already has a pending request in the queue.

When an error is generated by the module, the SEMERR register is updated with the error code, semaphore number (n = 0 to 63) and the ID (m = 0 to 15) of the core that caused the error. Refer to Error Register (SEMERR) for a more detailed description.

The SEMERR register can be cleared by writing '1h' to the SEMERR_CLEAR[0] CLRERR bit.

NOTE: The SEMERR register is influenced by the SEMERR interrupt state. To capture and register errors continuously besides clearing the SEMERR register, the application must also reset the SEMERR interrupt state each time a new error is detected. Refer to the interrupt handling section for a description of the steps for resetting the interrupt (Section 8.2.3.3.3.2.1, *Servicing SEM_ERRx*).

8.2.3.3.3 Interrupts

The Semaphore module can generate two sets of interrupts (events):

- Semaphore Grant Interrupt (SEM_INTx)
- Semaphore Error Interrupt (SEM_ERRx)

Each set has independent lines capable of addressing any specific core on the device. See Section 8.2.2, *Semaphore Integration*.

8.2.3.3.3.1 Semaphore Grant Interrupt (SEM_INTx)

The SEM_INTx is generated whenever a pending request is granted for any of the semaphores. The interrupt is directed to the specific core that posted the original request. The bit marking the semaphore that was granted is set in either the SEMFLAGL_m (semaphore < 32) or the SEMFLAGH_m (semaphore >= 32) register corresponding to that core. Each time a SEM_INTx is generated, the SEM_INTx line to that core is disabled to prevent further semaphore interrupts from being generated.

8.2.3.3.3.1.1 Servicing SEM_INTx

Whenever SEM_INTx is received by a core, the following steps are performed by the application to clear the event and re-enable the interrupt:

1. Read the SEMFLAGL_m and/or the SEMFLAGH_m register to determine the semaphore that was granted.
2. Write '1h' to the SEMFLAGL_CLEAR_m or the SEMFLAGH_CLEAR_m register bit corresponding to the semaphore that was granted.

   This clears the corresponding bit in the flag register.
3. Write the core ID number to the End-of-Interrupt Register (SEM_EOI).

   This re-enables the SEM_INTx line to that core and enables the generation of further SEM_INTx interrupts.

8.2.3.3.3.2 Semaphore Error Interrupt (SEM_ERRx)

The SEM_ERRx interrupt is generated whenever an error condition is detected. Refer to Section 8.2.3.3.2, *Error Generation and Handling* for a more detailed description of the error conditions. The SEMERR register is updated with the error condition and core that caused the condition. Refer to the SEMERR register for a description of the fields. Each time a SEM_ERRx is generated, all the SEM_ERRx lines are disabled to prevent any further semaphore error interrupts from being generated.

8.2.3.3.3.2.1 Servicing SEM_ERRx

Whenever SEM_ERRx is received by a core, the following steps are performed by the application to clear the event and re-enable the error interrupts.

1. Read the SEMERR register to determine the error condition.
2. Clear the SEMERR register by writing '1h' to the SEMERR_CLEAR register.
3. Write '10h' to the SEM_EOI register to re-arm the semaphore error interrupts to all cores.

8.2.3.4 Emulation Considerations

During debug when using the emulator, the core or cores may be halted for debugging. During emulation halt, the debugger read to semaphore registers is ignored. That is, debugger read of semaphore registers like SEM_n or ISEM_n will not change the state of the semaphore peripheral.

However, during emulation halt, the debugger can always see the QSEM_n register to check the semaphore peripherals status.

8.2.3.5 Semaphore Request Methods

8.2.3.5.1 Direct Semaphore Request

Direct request is the simplest method used to request a semaphore. The request behaves as an atomic read and set operation. The result of a request is either to grant the semaphore to the requesting core or deny the request because the semaphore has already been granted to another core.

8.2.3.5.1.1 Issuing a Direct Request

A direct request is issued by reading the SEM_n[15-8] FREE8_15 field corresponding to the semaphore requested. As seen in Figure 8-36 there are up to 64 direct registers, one corresponding to each of the semaphores.

If the semaphore is free, the read of the SEM_n[15-8] FREE8_15 field returns '1h'; if the request is rejected, the read returns the ID of the core that currently owns the semaphore. See SEM_n register for a detailed description of the read value.

8.2.3.5.1.2 Interrupt Events

No semaphore grant (SEM_INTx) interrupts are generated when the direct request method is used. However, an error interrupt may be generated. For more information, about semaphore access errors, see Section 8.2.3.3.2, *Error Generation and Handling*.

8.2.3.5.2 Indirect Semaphore Request

The indirect request method acquires a semaphore with one request. If a semaphore was not free using the direct request method, the application has to keep trying until it is granted a semaphore. With the indirect method, the request for a semaphore is submitted to a queue. The request queue is processed sequentially by the Semaphore module on a 'first-in-first-out' basis. The semaphore is granted to the core whose request is at the top of the request queue and is marked busy until it is released, at which point the next request (if any) is processed.

8.2.3.5.2.1 Issuing an Indirect Request

An indirect request is issued by writing '0h' to the SEM_n[0] FREE0, ISEM_n[0] FREE0, or the QSEM_n[0] FREE0 bits corresponding to the semaphore requested. As shown in Figure 8-36, there are up to 64 sets of the three registers – one corresponding to each of the semaphores.

When a semaphore becomes available or is free, the semaphore is granted to the core that has its request at the top of the queue.

8.2.3.5.2.2 Interrupt Events

When a semaphore is granted to a core via a request made using the indirect method, a SEM_INTx interrupt to that core is generated.

8.2.3.5.3 Combined Semaphore Request

A combined semaphore request is a hybrid version of the direct and indirect methods. It behaves like a direct request if the semaphore is free; otherwise it behaves like an indirect request and posts a request in the queue.

8.2.3.5.3.1 Issuing an Indirect Request

A combined request is issued by reading the ISEM_n[15-8] FREE8_15 field corresponding to the semaphore being requested.

If the semaphore is free, it is granted to the core making the request and the read returns '1h'.

If the semaphore it is not free, the resulting read returns the ID of the core that currently owns the semaphore. A request for the semaphore is also posted to the queue for that semaphore. Refer to ISEM_n register for a detailed description of the read value. The semaphore is granted whenever the posted request is processed.

8.2.3.5.3.2 Interrupt Events

For a combined request where a semaphore is free at the time the request is made, a SEM_INTx interrupt is not generated when the semaphore is granted to the requestor. However, if it is posted to the request queue because the semaphore was not free at the time of request, a SEM_INTx interrupt is generated to that core when the semaphore is granted to it.

8.2.3.6 Releasing Semaphores

Whenever a core is finished using a shared resource, it must free the associated semaphore so that other processes waiting to use it can get access. To release a semaphore and set its state to 'FREE', the application must write '1h' to one of SEM_n[0] FREE, ISEM_n[0] FREE, or QSEM_n[0] FREE bits corresponding to that semaphore.

NOTE: A semaphore can be released only by a process that is running on the core that is currently its owner; any other core trying to release the semaphore will generate an error.

8.2.3.7 Querying Semaphore Status

It is reasonable that the application would need to check the status of a semaphore without acquiring it. The Semaphore module provides the ability to query the status of any of the semaphores.

To query the status of a semaphore, the application needs to read the QSEM_n[15-8] FREE8_15 field corresponding to the semaphore. The query returns information about whether the semaphore is free; if it is not free, the application returns the ID of the current owner.

8.2.4 Semaphore Registers

Table 8-53 lists the memory-mapped registers for the Semaphore module. All register offset addresses not listed in Table 8-53 should be considered as reserved locations and the register contents should not be modified.

Table 8-52. SEMAPHORE Instances

Instance	Base Address
SEMAPHORE	0264 0000h

Table 8-53. SEMAPHORE Registers

Offset	Acronym	Register Name	Semaphore Physical Address	Section
0h	SEM_PID	Peripheral Revision ID Register	0264 0000h	Section 8.2.4.1
8h	SEM_RST_RUN	Reset/Run Status Register	0264 0008h	Section 8.2.4.2
Ch	SEM_EOI	End of Interrupt Register	0264 000Ch	Section 8.2.4.3
100h to 1FCh	SEM_0 to SEM_63	Semaphore Direct Registers	0264 0100h to 0264 01FCh	Section 8.2.4.4
200h to 2FCh	ISEM_0 to ISEM_63	Semaphore Indirect Registers	0264 0200h to 0264 02FCh	Section 8.2.4.5
300h to 3FCh	QSEM_0 to QSEM_63	Semaphore Query Registers	0264 0300h to 0264 03FCh	Section 8.2.4.6
400h to 43Ch	SEMFLAGL_0 to SEMFLAGL_15(1)	Flag Low Registers	0264 0400h to 0264 043Ch	Section 8.2.4.7
400h to 43Ch	SEMFLAGL_CLEAR_0 to SEMFLAGL_CLEAR_15(1)	Flag Low Clear Registers	0264 0400h to 0264 043Ch	Section 8.2.4.8
440h to 47Ch	SEMFLAGH_0 to SEMFLAGH_15(1)	Flag High Registers	0264 0440h to 0264 047Ch	Section 8.2.4.9
440h to 47Ch	SEMFLAGH_CLEAR_0 to SEMFLAGH_CLEAR_15(1)	Flag High Clear Registers	0264 0440h to 0264 047Ch	Section 8.2.4.10
480h to 4BCh	SEMFLAGL_SET_0 to SEMFLAGL_SET_15(1)	Flag Low Set Registers	0264 0480h to 0264 04BCh	Section 8.2.4.11
4C0h to 4FCh	SEMFLAGH_SET_0 to SEMFLAGH_SET_15(1)	Flag High Set Registers	0264 04C0h to 0264 04FCh	Section 8.2.4.12
500h	SEMERR	Error Register	0264 0500h	Section 8.2.4.13
504h	SEMERR_CLEAR	Error Clear Register	0264 0504h	Section 8.2.4.14
508h	SEMERR_SET	Error Set Register	0264 0508h	Section 8.2.4.15

(1) The maximum number of masters (device cores) available for this device is 16. Only registers with index: 0, 8, 11, 12, 13, 14, or 15 are used. The rest registers are not used on this device. See Section 8.2.2, *Semaphore Integration*.

8.2.4.1 SEM_PID Register (Offset = 0h) [reset = 48021A00h]

SEM_PID is shown in Figure 8-37 and described in Table 8-55.

The Peripheral ID register contains the revision number and identification data of the semaphore peripheral.

Table 8-54. SEM_PID Instances

Instance	Physical Address
SEMAPHORE	0264 0000h

Figure 8-37. SEM_PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-48021A00h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 8-55. SEM_PID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	48021A00h	TI internal data. Identifies revision of peripheral.

Table 8-56. Register Call Summary for SEM_PID

Semaphore Registers

- SEM_PID Register (Offset = 0h) [reset = 48021A00h]: [0]
- Semaphore Registers: [0]

8.2.4.2 SEM_RST_RUN Register (Offset = 8h) [reset = 0h]

SEM_RST_RUN is shown in Figure 8-38 and described in Table 8-58.

This register contains the reset status of the semaphore module. This register is also used to soft reset the module.

Table 8-57. SEM_RST_RUN Instances

Instance	Physical Address
SEMAPHORE	0264 0008h

Figure 8-38. SEM_RST_RUN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RESET
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 8-58. SEM_RST_RUN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	RESET	R/W	0h	Reset bit 0h (R) = Module is not ready. 0h (W) = No affect. 1h (R) = Module ready. 1h (W) = Resets the module.

Table 8-59. Register Call Summary for SEM_RST_RUN

Semaphore Registers • Semaphore Registers: [0] • SEM_RST_RUN Register (Offset = 8h) [reset = 0h]: [0]
Semaphore Functional Description • Software Reset Considerations: [0]

8.2.4.3 SEM_EOI Register (Offset = Ch) [reset = 0h]

SEM_EOI is shown in Figure 8-39 and described in Table 8-61.

This register is used to re-arm the interrupts.

Table 8-60. SEM_EOI Instances

Instance	Physical Address
SEMAPHORE	0264 000Ch

Figure 8-39. SEM_EOI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								EOI							
R-0h																								W-0h							

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 8-61. SEM_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	EOI	W	0h	Interrupt End of Interrupt 0h (W) = Re-arm SEM_INT0. 1h (W) = Re-arm SEM_INT1. ... 10h (W) = Re-arm SEMERR register.

Table 8-62. Register Call Summary for SEM_EOI

Semaphore Registers • Semaphore Registers: [0] • SEM_EOI Register (Offset = Ch) [reset = 0h]: [0]
Semaphore Functional Description • Semaphore Grant Interrupt (SEM_INTx): [0] • Semaphore Error Interrupt (SEM_ERRx): [0]

8.2.4.4 SEM_0 to SEM_63 Register (Offset = 100h to 1FCh) [reset = 1h]

Semaphore Direct Register SEM_n is shown in Figure 8-40 and described in Table 8-64.

There are 64 direct registers (n = 0 to 63). Each one is associated with and used to manage the respective semaphore.

Table 8-63. SEM_0 to SEM_63 Instances

Instance	Physical Address
SEMAPHORE	0264 0100h to 0264 01FCh

Figure 8-40. SEM_0 to SEM_63 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
FREE8_15							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							FREE0
R-0h							R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 8-64. SEM_0 to SEM_63 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	FREE8_15	R	0h	Semaphore resource owner FREE0 = 0h: Read returns Core ID of current owner(1). FREE0 = 1h: Read returns 0h.
7-1	RESERVED	R	0h	Reserved
0	FREE0	R/W	1h	Direct Semaphore flag control, read/write action 0h (R) = Semaphore is not granted. 0h (W) = Request posted to queue. 1h (R) = Semaphore is granted to the reader. 1h (W) = Semaphore is set free(2).

(1) Core ID value: 0h-Fh.
(2) Only if writer is the current owner.

Table 8-65. Register Call Summary for SEM_0

Semaphore Registers
• Semaphore Registers: [0]

8.2.4.5 ISEM_0 to ISEM_63 Register (Offset = 200h to 2FCh) [reset = 1h]

Semaphore Indirect Register ISEM_n is shown in Figure 8-41 and described in Table 8-67.

There are 64 indirect registers (n = 0 to 63). Each one is associated with and used to manage the respective semaphore.

Table 8-66. ISEM_0 to ISEM_63 Instances

Instance	Physical Address
SEMAPHORE	0264 0200h to 0264 02FCh

Figure 8-41. ISEM_0 to ISEM_63 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
FREE8_15							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							FREE0
R-0h							R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 8-67. ISEM_0 to ISEM_63 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	FREE8_15	R	0h	Semaphore resource owner FREE0 = 0h: Read returns Core ID of current owner(1). FREE0 = 1h: Read returns 0h.
7-1	RESERVED	R	0h	Reserved
0	FREE0	R/W	1h	Indirect Semaphore flag control, read/write action 0h (R) = Semaphore is not granted, request posted to queue. 0h (W) = Request posted to queue. 1h (R) = Semaphore is granted to the reader. 1h (W) = Semaphore is set free(2).

(1) Core ID value: 0h-Fh.
(2) Only if writer is the current owner.

Table 8-68. Register Call Summary for ISEM_0

Semaphore Registers
• Semaphore Registers: [0]

8.2.4.6 QSEM_0 to QSEM_63 Register (Offset = 300h to 3FCh) [reset = 1h]

Semaphore Query Register QSEM_n is shown in Figure 8-42 and described in Table 8-70.

There are 64 query registers (n = 0 to 63). Each one is associated with and used to manage the respective semaphore.

Table 8-69. QSEM_0 to QSEM_63 Instances

Instance	Physical Address
SEMAPHORE	0264 0300h to 0264 03FCh

Figure 8-42. QSEM_0 to QSEM_63 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
FREE8_15							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							FREE0
R-0h							R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 8-70. QSEM_0 to QSEM_63 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	FREE8_15	R	0h	Semaphore resource owner FREE0 = 0h: Read returns Core ID of current owner(1). FREE0 = 1h: Read returns 0h.
7-1	RESERVED	R	0h	Reserved
0	FREE0	R/W	1h	Semaphore flag query and control, read/write action 0h (R) = Semaphore is not available. 0h (W) = Request posted to queue. 1h (R) = Semaphore is available. 1h (W) = Semaphore is set free(2).

(1) Core ID value: 0h-Fh.

(2) Only if writer is the current owner.

Table 8-71. Register Call Summary for QSEM_0

Semaphore Registers
• Semaphore Registers: [0]

8.2.4.7 SEMFLAGL_0 to SEMFLAGL_15 Register (Offset = 400h to 43Ch) [reset = 0h]

SEMFLAGL_m is shown in Figure 8-43 and described in Table 8-73.

The bits flag that an interrupt was generated to that core granting it the semaphore. If a bit is set, it does not particularly mean that the core still owns the semaphore.

The maximum number of masters (device cores) available for this device is 16. Only registers with index: 0, 8, 11, 12, 13, 14, or 15 are used. The rest registers are not used on this device. See Section 8.2.2, *Semaphore Integration*.

Table 8-72. SEMFLAGL_0 to SEMFLAGL_15 Instances

Instance	Physical Address
SEMAPHORE	0264 0400h to 0264 043Ch

Figure 8-43. SEMFLAGL_0 to SEMFLAGL_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
F31	F30	F29	F28	F27	F26	F25	F24	F23	F22	F21	F20	F19	F18	F17	F16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F15	F14	F13	F12	F11	F10	F9	F8	F7	F6	F5	F4	F3	F2	F1	F0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 8-73. SEMFLAGL_0 to SEMFLAGL_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Fn[(1)]	R	0h	Semaphore flag n[(1)] 0h (R) = No interrupt generated to core-m. 1h (R) = Interrupt was generated core-m to signal that semaphore-n was granted to it.

(1) n = 0 to 31

Table 8-74. Register Call Summary for SEMFLAGL_0

Semaphore Registers
• Semaphore Registers: [0]

8.2.4.8 SEMFLAGL_CLEAR_0 to SEMFLAGL_CLEAR_15 Register (Offset = 400h to 43Ch) [reset = 0h]

SEMFLAGL_CLEAR_m is shown in Figure 8-44 and described in Table 8-76.

The SEMFLAGL_CLEAR_m register is used to clear the flag bits in the SEMFLAGL_m register.

The maximum number of masters (device cores) available for this device is 16. Only registers with index: 0, 8, 11, 12, 13, 14, or 15 are used. The rest registers are not used on this device. See Section 8.2.2, *Semaphore Integration*.

Table 8-75. SEMFLAGL_CLEAR_0 to SEMFLAGL_CLEAR_15 Instances

Instance	Physical Address
SEMAPHORE	0264 0400h to 0264 043Ch

Figure 8-44. SEMFLAGL_CLEAR_0 to SEMFLAGL_CLEAR_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
F31	F30	F29	F28	F27	F26	F25	F24	F23	F22	F21	F20	F19	F18	F17	F16
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F15	F14	F13	F12	F11	F10	F9	F8	F7	F6	F5	F4	F3	F2	F1	F0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

LEGEND: W = Write Only; *-n* = value after reset

Table 8-76. SEMFLAGL_CLEAR_0 to SEMFLAGL_CLEAR_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Fn[(1)]	W	0h	Semaphore Flag Clear n[(1)] 0h (W) = No effect. 1h (W) = Flag bit n is cleared.

(1) n = 0 to 31

Table 8-77. Register Call Summary for SEMFLAGL_CLEAR_0

Semaphore Registers
• Semaphore Registers: [0]

8.2.4.9 SEMFLAGH_0 to SEMFLAGH_15 Register (Offset = 440h to 47Ch) [reset = 0h]

SEMFLAGH_m is shown in Figure 8-45 and described in Table 8-79.

The bits flag that an interrupt was generated to that core granting it the semaphore. If a bit is set, it does not particularly mean that the core still owns the semaphore.

The maximum number of masters (device cores) available for this device is 16. Only registers with index: 0, 8, 11, 12, 13, 14, or 15 are used. The rest registers are not used on this device. See Section 8.2.2, *Semaphore Integration*.

Table 8-78. SEMFLAGH_0 to SEMFLAGH_15 Instances

Instance	Physical Address
SEMAPHORE	0264 0440h to 0264 047Ch

Figure 8-45. SEMFLAGH_0 to SEMFLAGH_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
F63	F62	F61	F60	F59	F58	F57	F56	F55	F54	F53	F52	F51	F50	F49	F48
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F47	F46	F45	F44	F43	F42	F41	F40	F39	F38	F37	F36	F35	F34	F33	F32
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 8-79. SEMFLAGH_0 to SEMFLAGH_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Fn[(1)]	R	0h	Semaphore flag n[(1)] 0h (R) = No interrupt generated to core-m. 1h (R) = Interrupt was generated core-m to signal that semaphore-n was granted to it.

(1) n = 32 to 63

Table 8-80. Register Call Summary for SEMFLAGH_0

Semaphore Registers
• Semaphore Registers: [0]

8.2.4.10 SEMFLAGH_CLEAR_0 to SEMFLAGH_CLEAR_15 Register (Offset = 440h to 47Ch) [reset = 0h]

SEMFLAGH_CLEAR_m is shown in Figure 8-46 and described in Table 8-82.

The SEMFLAGH_CLEAR_m register is used to clear the flag bits in the SEMFLAGH_m register.

The maximum number of masters (device cores) available for this device is 16. Only registers with index: 0, 8, 11, 12, 13, 14, or 15 are used. The rest registers are not used on this device. See Section 8.2.2, *Semaphore Integration*.

Table 8-81. SEMFLAGH_CLEAR_0 to SEMFLAGH_CLEAR_15 Instances

Instance	Physical Address
SEMAPHORE	0264 0440h to 0264 047Ch

Figure 8-46. SEMFLAGH_CLEAR_0 to SEMFLAGH_CLEAR_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
F63	F62	F61	F60	F59	F58	F57	F56	F55	F54	F53	F52	F51	F50	F49	F48
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F47	F46	F45	F44	F43	F42	F41	F40	F39	F38	F37	F36	F35	F34	F33	F32
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

LEGEND: W = Write Only; *-n* = value after reset

Table 8-82. SEMFLAGH_CLEAR_0 to SEMFLAGH_CLEAR_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Fn[(1)]	W	0h	Semaphore Flag Clear n[(1)] 0h (W) = No effect. 1h (W) = Flag bit n is cleared.

(1) n = 32 to 63

Table 8-83. Register Call Summary for SEMFLAGH_CLEAR_0

Semaphore Registers
• Semaphore Registers: [0]

8.2.4.11 SEMFLAGL_SET_0 to SEMFLAGL_SET_15 Register (Offset = 480h to 4BCh) [reset = 0h]

SEMFLAGL_SET_m is shown in Figure 8-47 and described in Table 8-85.

The SEMFLAGL_SET_m register is used to set the flag bits in the SEMFLAGL_m register and also generate the associated SEM_INTx interrupt. This can be used for testing and debugging purposes.

The maximum number of masters (device cores) available for this device is 16. Only registers with index: 0, 8, 11, 12, 13, 14, or 15 are used. The rest registers are not used on this device. See Section 8.2.2, *Semaphore Integration*.

Table 8-84. SEMFLAGL_SET_0 to SEMFLAGL_SET_15 Instances

Instance	Physical Address
SEMAPHORE	0264 0480h to 0264 04BCh

Figure 8-47. SEMFLAGL_SET_0 to SEMFLAGL_SET_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
F31	F30	F29	F28	F27	F26	F25	F24	F23	F22	F21	F20	F19	F18	F17	F16
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F15	F14	F13	F12	F11	F10	F9	F8	F7	F6	F5	F4	F3	F2	F1	F0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

LEGEND: W = Write Only; *-n* = value after reset

Table 8-85. SEMFLAGL_SET_0 to SEMFLAGL_SET_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Fn[(1)]	W	0h	Semaphore Flag Set n[(1)] 0h (W) = No effect. 1h (W) = Flag bit n is set, SEM_INTx is generated.

(1) n = 0 to 31

Table 8-86. Register Call Summary for SEMFLAGL_SET_0

Semaphore Registers
• Semaphore Registers: [0]

8.2.4.12 SEMFLAGH_SET_0 to SEMFLAGH_SET_15 Register (Offset = 4C0h to 4FCh) [reset = 0h]

SEMFLAGH_SET_m is shown in Figure 8-48 and described in Table 8-88.

The SEMFLAGH_SET_m register is used to set the flag bits in the SEMFLAGH_m register and also generate the associated SEM_INTx interrupt. This can be used for testing and debugging purposes.

The maximum number of masters (device cores) available for this device is 16. Only registers with index: 0, 8, 11, 12, 13, 14, or 15 are used. The rest registers are not used on this device. See Section 8.2.2, *Semaphore Integration*.

Table 8-87. SEMFLAGH_SET_0 to SEMFLAGH_SET_15 Instances

Instance	Physical Address
SEMAPHORE	0264 04C0h to 0264 04FCh

Figure 8-48. SEMFLAGH_SET_0 to SEMFLAGH_SET_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
F63	F62	F61	F60	F59	F58	F57	F56	F55	F54	F53	F52	F51	F50	F49	F48
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F47	F46	F45	F44	F43	F42	F41	F40	F39	F38	F37	F36	F35	F34	F33	F32
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

LEGEND: W = Write Only; *-n* = value after reset

Table 8-88. SEMFLAGH_SET_0 to SEMFLAGH_SET_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Fn[(1)]	W	0h	Semaphore Flag Set n[(1)] 0h (W) = No effect. 1h (W) = Flag bit n is set, SEM_INTx is generated.

(1) n = 32 to 63

Table 8-89. Register Call Summary for SEMFLAGH_SET_0

Semaphore Registers
• Semaphore Registers: [0]

8.2.4.13 SEMERR Register (Offset = 500h) [reset = 0h]

SEMERR is shown in Figure 8-49 and described in Table 8-91.

The SEMERR register contains information pertaining to the last generated SEM_ERRx interrupt.

Table 8-90. SEMERR Instances

Instance	Physical Address
SEMAPHORE	0264 0500h

Figure 8-49. SEMERR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																FAULTID							SEMNUM						ERR		
R-0h																R-0h							R-0h						R-0h		

LEGEND: R = Read Only; *-n* = value after reset

Table 8-91. SEMERR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-9	FAULTID	R	0h	Master that caused the error Read returns ID of the core that is responsible for the error (0h-Fh).
8-3	SEMNUM	R	0h	Semaphore Number Read returns Semaphore number associated with error (0h-3Fh).
2-0	ERR	R	0h	Semaphore Error Code 0h (R) = No Error. 1h (R) = Already Free Error. 2h (R) = Illegal Free Error. 3h (R) = Already Own Error. 4h (R) = Already Requested Error.

Table 8-92. Register Call Summary for SEMERR

Semaphore Registers • Semaphore Registers: [0] • SEM_EOI Register (Offset = Ch) [reset = 0h]: [0] • SEMERR_CLEAR Register (Offset = 504h) [reset = 0h]: [0][1] • SEMERR_SET Register (Offset = 508h) [reset = 0h]: [0][1] • SEMERR Register (Offset = 500h) [reset = 0h]: [0][1]
Semaphore Functional Description • Error Generation and Handling: [0][1][2][3][4][5][6] • Semaphore Error Interrupt (SEM_ERRx): [0][1][2][3]

8.2.4.14 SEMERR_CLEAR Register (Offset = 504h) [reset = 0h]

SEMERR_CLEAR is shown in Figure 8-50 and described in Table 8-94.

This SEMERR_CLEAR register is used to clear the error condition in the SEMERR register.

Table 8-93. SEMERR_CLEAR Instances

Instance	Physical Address
SEMAPHORE	0264 0504h

Figure 8-50. SEMERR_CLEAR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CLRERR
R-0h							W-0h

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 8-94. SEMERR_CLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	CLRERR	W	0h	Clear Semaphore Error 0h (W) = No effect. 1h (W) = Clears the SEMERR register.

Table 8-95. Register Call Summary for SEMERR_CLEAR

Semaphore Registers • Semaphore Registers: [0] • SEMERR_CLEAR Register (Offset = 504h) [reset = 0h]: [0][1]
Semaphore Functional Description • Error Generation and Handling: [0] • Semaphore Error Interrupt (SEM_ERRx): [0]

8.2.4.15 SEMERR_SET Register (Offset = 508h) [reset = 0h]

SEMERR_SET is shown in Figure 8-51 and described in Table 8-97.

The SEMERR_SET register is used to generate an error and the associated SEMERR interrupt by setting the SEMERR register values. It can be used for testing and debug purposes.

Table 8-96. SEMERR_SET Instances

Instance	Physical Address
SEMAPHORE	0264 0508h

Figure 8-51. SEMERR_SET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																FAULTID							SEMNUM						ERR		
R-0h																W-0h							W-0h						W-0h		

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 8-97. SEMERR_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-9	FAULTID	W	0h	Set Master that caused the error ID of the core that will be responsible for the error (0h-Fh).
8-3	SEMNUM	W	0h	Set Semaphore Number Semaphore number associated with error (0h-3Fh).
2-0	ERR	W	0h	Set Semaphore error code 0h (W) = No Error. 1h (W) = Already Free Error. 2h (W) = Illegal Free Error. 3h (W) = Already Own Error. 4h (W) = Already Requested Error.

Table 8-98. Register Call Summary for SEMERR_SET

Semaphore Registers
- Semaphore Registers: [0]
- SEMERR_SET Register (Offset = 508h) [reset = 0h]: [0][1]

Interrupts

Topic **Page**

9.1 Chip-level Interrupt Controller (CIC)

This chapter describes the Chip-level Interrupt Controller (CIC) present on the device.

9.1.1 CIC Overview

The SoC has many peripherals and a large number of event sources (interrupt or DMA). The use of events is completely dependent on a user's specific application, which drives a need for maximum flexibility in which event sources are used in the system. It is also completely up to software control as to how the events are serviced.

The SoC integrates the following event-servicing modules (hosts):

- Arm Subsytem (single Cortex-A15 CPU), also known as Arm CorePac
- DSP Subsystem (single C66x CPU), also known as C66x CorePac
- Industrial Communications Subsystems (PRU-ICSS_0 and PRU-ICSS_1)
- EDMA Controllers (EDMA_0 and EDMA_1)
- Power Management Micro Controller (PMMC)

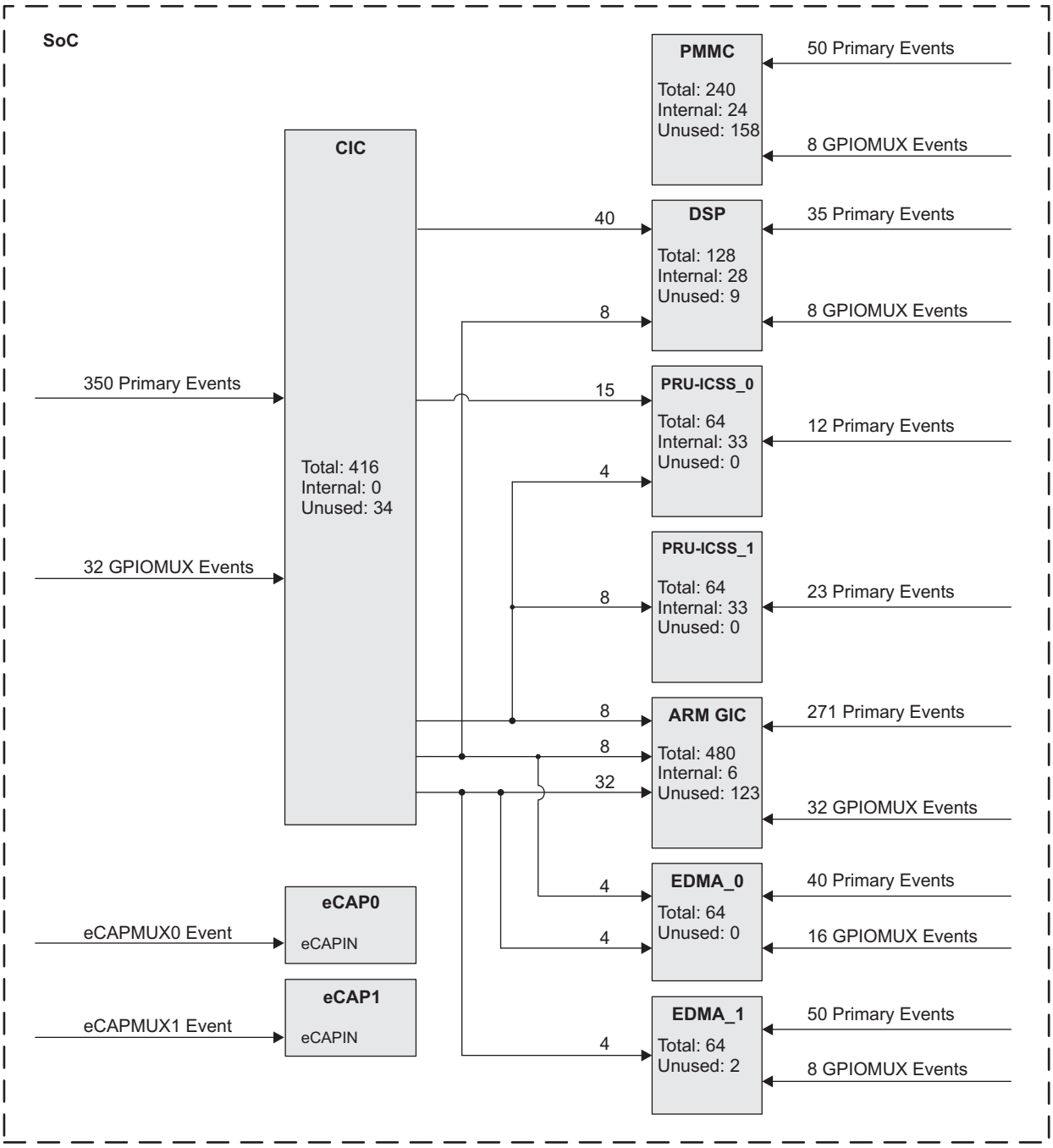
These hosts are capable of receiving events directly but the number of accepted events for each host is limited. To serve the large number of chip-level (system) event sources, the device instantiates a **Chip-level Interrupt Controller (CIC)**, which can be defined as a **system event router** – it aggregates the system events and routes them to the different hosts by using simple combination logic.

The CIC module has the following event input/output configuration for this SoC:

- 416 event inputs
- 104 event outputs

Figure 9-1 shows the SoC interrupt topology.

Figure 9-1. SoC Interrupt Topology



NOTE: This chapter covers only the functionality and interrupt mapping related to the CIC module. For functional description and interrupt mapping related to the different hosts shown on Figure 9-1 (and also GPIOMUX and eCAPMUX modules), see their respective chapters.

Some device modules have an interrupt that requires End-of-Interrupt (EOI) handshaking. The interrupt can be triggered by multiple conditions. After the host services the interrupt, the host needs to write a value into the EOI_VECTOR field in the corresponding EOI register (there is one such register for each peripheral and it is part of the module's register space) to acknowledge that it has cleared the interrupt condition before a new interrupt can be generated.

Table 9-1 shows the values that must be programmed in the corresponding module's EOI register.

Table 9-1. EOI Values

Module	EOI Value
MPU	0
TRACER	0
NSS_CDMA_STARVE	0
NSS_ESW_STAT_PEND0	1
NSS_ESW_STAT_PEND1	2
NSS_ESW_EVT_PEND	3
NSS_MDIO_LINK0	4
NSS_MDIO_LINK1	5
NSS_MDIO_USER0	6
NSS_MDIO_USER1	7

Additional event-generation features supported by the SoC are as follows:

- Inter-Processor Communication (IPC) – device cores can communicate with one another through inter-processor interrupts for core synchronization, allowing for direct notification from one core to another
- Non-Maskable Interrupt (NMI) generation to C66x DSP

For more details on these features, see Section 5.1, *Control Module (BOOT_CFG)*.

9.1.2 CIC Integration

This section describes the integration of the module in the device, including information about clocks, resets, and hardware requests.

Table 9-2 through Table 9-4 summarize the integration of the CIC module.

Table 9-2. CIC Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
CIC	PD5	LPSC6	TeraNet

Table 9-3. CIC Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
CIC	CIC_VBUS_CLK	CHIP_CLK1 / 6	PLL Controller	Functional and interface (VBUS) clock.
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
CIC	CIC_RST	MOD_G_RST	LPSC6	Hardware reset.

Table 9-4. CIC Hardware Requests

Interrupt Outputs								
Module Instance	**Event Name**	**Mapped To Input Event [Number]**						**Description**
		ARM GIC	**DSP INTC**	**PRU-ICSS_0 INTC**	**PRU-ICSS_1 INTC**	**EDMA_0**	**EDMA_1**	
CIC	CIC_0_OUT[0:23]	[336:359]	–	–	–	–	–	CIC_0_OUT[0:23] host interrupts
	CIC_0_OUT[24:25]	[360:361]	–	–	–	–	[22:23]	CIC_0_OUT[24:25] host interrupts
	CIC_0_OUT[26:27]	[362:363]	–	–	–	–	[46:47]	CIC_0_OUT[26:27] host interrupts
	CIC_0_OUT[28:31]	[364:367]	–	–	–	[36:39]	–	CIC_0_OUT[28:31] host interrupts
	CIC_0_OUT[32:35]	[368:371]	[48:51]	–	–	[28:31]	–	CIC_0_OUT[32:35] host interrupts
	CIC_0_OUT[36:39]	[372:375]	[52:55]	–	–	–	–	CIC_0_OUT[36:39] host interrupts
	CIC_0_OUT[40:79]	–	[56:95]	–	–	–	–	CIC_0_OUT[40:79] host interrupts
	CIC_0_OUT[80:83]	[376:379]	–	[36:39]	[45:48]	–	–	CIC_0_OUT[80:83] host interrupts
	CIC_0_OUT[84:87]	[380:383]	–	–	[49:52]	–	–	CIC_0_OUT[84:87] host interrupts
	CIC_0_OUT[88:102]	–	–	[40:54]	–	–	–	CIC_0_OUT[88:102] host interrupts
Interrupt Inputs								
See Section 9.2								

9.1.3 CIC Functional Description

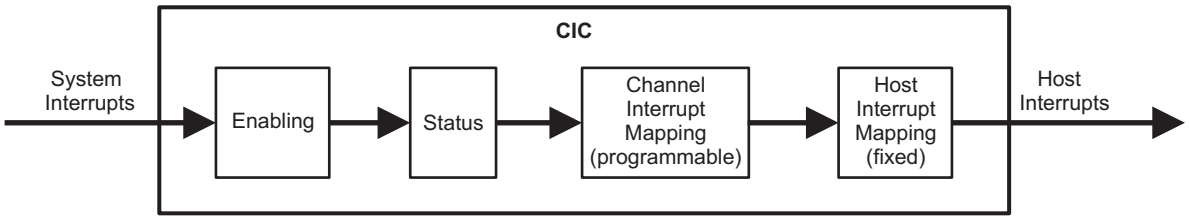
The CIC module controls the system interrupt mapping to the host interrupt interface. System interrupts are generated by device peripherals. System interrupts can be individual for a single core or can be shared by multiple cores.

The CIC receives the system interrupts and maps them to its internal channels. Channels are used to group interrupts together. These channels are then mapped onto the interrupt interfaces of the corresponding device hosts. The configuration of the CIC in the device has a fixed one-to-one mapping between channels and host interrupts.

The CIC encompasses several functions to process the system interrupts and prepare them for the host interface. These functions are:

- Enabling
- Status
- Channel mapping (programmable)
- Host interrupt mapping (fixed)

Figure 9-2. CIC Block Diagram



9.1.3.1 Interrupt Enabling

The first stage of the CIC is to enable the system interrupts based on programmed settings. Each system interrupt can be individually enabled or disabled by software. The system interrupts that are disabled will not generate any host interrupts.

The CIC receives 416 system interrupts at its inputs (see Section 9.2 for interrupt mapping).

9.1.3.2 Interrupt Status

The second stage of the CIC is to capture which system interrupts are pending. This status is captured into a status register that is readable by the system to identify the system interrupts that are pending and/or enabled. The pending status reflects whether the system interrupt occurred since the last time the status register bit was cleared. Each bit in the status register is individually clearable. Status is also cleared when the specific status is cleared. Software can also set interrupts without hardware triggering.

There are two kinds of status: *raw* status and *enabled* status. Raw status is the pending status of the system interrupt without regard to the enable bit for the system interrupt. Enabled status is the pending status of the system interrupts with the enable bits active. When the enable bit is inactive the enabled status will always be inactive. Only the enabled status will be used in later stages in the CIC, while raw status is provided for software or debug usage.

9.1.3.3 Channel Mapping

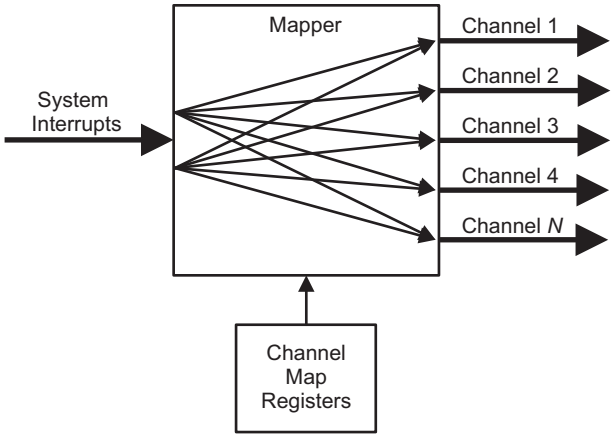
The third stage of the CIC is to map the system interrupts that have been enabled to CIC internal channels. Channels are used to group the system interrupts into a smaller number of interrupt inputs that can be provided to a host interface. When multiple system interrupts are mapped to the same channel, they are OR'd together so that when at least one is active, the output is active.

The CIC implements 104 internal channels.

Figure 9-3 shows the channel mapping function. Note that a system interrupt cannot be mapped to multiple channels but the connection is selected by the channel map value.

The mapping of the system interrupts to channels is programmable. Each system interrupt has a dedicated register to define its mapping.

Figure 9-3. Channel Mapping Block Diagram



9.1.3.4 Host Interrupt Mapping

The last stage of the CIC is host mapping. This stage maps the defined number of channels to the host interrupts.

The mapping of the channels to host interrupts is fixed (one-to-one mapping). Each channel has a dedicated register to define its host interrupt and the register is read-only.

9.1.3.5 Interrupt Service Sequence

One simple example of the interrupt service sequence is shown in Table 9-5. The interrupt service routine (ISR) executes when an interrupt is taken and services the pending interrupts before exiting.

Table 9-5. Interrupt Service Sequence

Step	Description	Example actions for C66x DSP
1	Receive the interrupt	Corresponding bit set in IFR register (by hardware) GIE bit cleared (by hardware) Execution from interrupt servicing table (IST) begins, entering to the ISR (by hardware) Corresponding bit cleared in IFR register (by hardware)
2	Disable the host interrupt	Disable the corresponding host interrupt output in the CIC Host Interrupt Enable Register
3	Determine the exact interrupt and clear the status of the system interrupt	Read the System Interrupt Status Enabled/Clear Register (for bit flags) Clear the system interrupt flag in CIC System Interrupt Status Clear Register and read back the register value.
4	Service the interrupt	Execute the user's code for servicing the interrupt
5	Re-enale the host interrupt	Enable the corresponding host interrupt output in the CIC Host Interrupt Enable Register
6	Return from interrupt	GIE bit set (by hardware) Return from ISR to the user's application

If the host can accept other host interrupts while processing the ISR then disabling all the host interrupts (in Step 2) would be suggested, assuming the ISR only wants to process one interrupt at a time. There is a global control bit (bit [0] in Global Enable Register) to enable/disable all the host interrupts to make the process easier (the individual host interrupt enables are maintained as well).

NOTE: When the CIC system interrupt flag is being cleared in ISR, if the same system interrupt happens again at the same cycle, the CIC system interrupt flag will not be cleared. By adding Step 2 (disabling the host interrupt) and Step 5 (re-enabling the host interrupt) in ISR will make sure to generate another interrupt to the host after exiting from the current ISR. So the new interrupt received during the ISR will not be missed. The dummy read in step 3 is added to make sure the system interrupt flag has been indeed cleared (writing to the register is completed) before further steps occur. It will prevent missed event as well.

9.1.4 CIC Registers

Table 9-7 lists the CIC configuration registers. All other register offset addresses not listed in Table 9-7 should be considered as reserved locations and the register contents should not be modified.

Table 9-6. CIC Instances

Instance	Base Address
CIC	0260 0000h

Table 9-7. CIC Registers

Offset	Acronym	Register Name	CIC Physical Address	Section
0h	CIC_REVISION_REG	Revision Register	0260 0000h	Section 9.1.4.1
10h	CIC_GLOBAL_ENABLE_HINT_REG	Global Enable Register	0260 0010h	Section 9.1.4.2
20h	CIC_STATUS_SET_INDEX_REG	System Interrupt Status Indexed Set Register	0260 0020h	Section 9.1.4.3
24h	CIC_STATUS_CLR_INDEX_REG	System Interrupt Status Indexed Clear Register	0260 0024h	Section 9.1.4.4
28h	CIC_ENABLE_SET_INDEX_REG	System Interrupt Enable Indexed Set Register	0260 0028h	Section 9.1.4.5
2Ch	CIC_ENABLE_CLR_INDEX_REG	System Interrupt Enable Indexed Clear Register	0260 002Ch	Section 9.1.4.6
34h	CIC_HINT_ENABLE_SET_INDEX_REG	Host Interrupt Enable Indexed Set Register	0260 0034h	Section 9.1.4.7
38h	CIC_HINT_ENABLE_CLR_INDEX_REG	Host Interrupt Enable Indexed Clear Register	0260 0038h	Section 9.1.4.8
200h to 230h	CIC_RAW_STATUS_REG0 to CIC_RAW_STATUS_REG12	System Interrupt Status Raw/Set Registers	0260 0200h to 0260 0230h	Section 9.1.4.9
280h to 2B0h	CIC_ENA_STATUS_REG0 to CIC_ENA_STATUS_REG12	System Interrupt Status Enabled/Clear Registers	0260 0280h to 0260 02B0h	Section 9.1.4.10
300h to 330h	CIC_ENABLE_REG0 to CIC_ENABLE_REG12	System Interrupt Enable Set Registers	0260 0300h to 0260 0330h	Section 9.1.4.11
380h to 3B0h	CIC_ENABLE_CLR_REG0 to CIC_ENABLE_CLR_REG12	System Interrupt Enable Clear Registers	0260 0380h to 0260 03B0h	Section 9.1.4.12
400h to 59Ch	CIC_CH_MAP_REG0 to CIC_CH_MAP_REG103	Channel Interrupt Map Registers	0260 0400h to 0260 059Ch	Section 9.1.4.13
800h to 864h	CIC_HINT_MAP_REG0 to CIC_HINT_MAP_REG25	Host Interrupt Map Registers	0260 0800h to 0260 0864h	Section 9.1.4.14
1500h to 150Ch	CIC_ENABLE_HINT_REG0 to CIC_ENABLE_HINT_REG3	Host Interrupt Enable Registers	0260 1500h to 0260 150Ch	Section 9.1.4.15

9.1.4.1 CIC_REVISION_REG Register (Offset = 0h) [reset = 4E82A900h]

The Revision Register (Figure 9-4) contains identification data for the peripheral.

Table 9-8. CIC_REVISION_REG Instances

Instance	Physical Address
CIC	0260 0000h

Figure 9-4. CIC_REVISION_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-4E82A900h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 9-9. CIC_REVISION_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4E82A900h	TI internal data. Identifies revision of peripheral.

Table 9-10. Register Call Summary for CIC_REVISION_REG

CIC Registers
- CIC Registers: [0]

9.1.4.2 CIC_GLOBAL_ENABLE_HINT_REG Register (Offset = 10h) [reset = 0h]

The Global Enable Register (Figure 9-5) enables all the host interrupts at once. Individual host interrupts are still enabled or disabled from their individual enables and are not overridden by the global enable.

Table 9-11. CIC_GLOBAL_ENABLE_HINT_REG Instances

Instance	Physical Address
CIC	0260 0010h

Figure 9-5. CIC_GLOBAL_ENABLE_HINT_REG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ENABLE
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 9-12. CIC_GLOBAL_ENABLE_HINT_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	ENABLE	R/W	0h	This field enables or disables all the host interrupts at once. Reads return the current global enable value. When written: • 0 = Disable all the host interrupts • 1 = Enable all the host interrupts

Table 9-13. Register Call Summary for CIC_GLOBAL_ENABLE_HINT_REG

CIC Registers
• CIC Registers: [0]

9.1.4.3 CIC_STATUS_SET_INDEX_REG Register (Offset = 20h) [reset = 0h]

The System Interrupt Status Indexed Set Register (Figure 9-6) allows setting the status of an interrupt. The interrupt to set is the index value written. This sets the System Interrupt Status Raw/Set Register bit of the given index.

Table 9-14. CIC_STATUS_SET_INDEX_REG Instances

Instance	Physical Address
CIC	0260 0020h

Figure 9-6. CIC_STATUS_SET_INDEX_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						INDEX									
R-0h																						R/W-0h									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 9-15. CIC_STATUS_SET_INDEX_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reads return 0 and writes have no effect.
9-0	INDEX	R/W	0h	This field allows setting the status of an interrupt. Reads return 0. Writes set the status of the interrupt given in the index value. • 0 = Set the status of the system interrupt 0 • 1h = Set the status of the system interrupt 1 • ... • 19Fh = Set the status of the system interrupt 415 • Others = Reserved

Table 9-16. Register Call Summary for CIC_STATUS_SET_INDEX_REG

CIC Registers
• CIC Registers: [0]

9.1.4.4 CIC_STATUS_CLR_INDEX_REG Register (Offset = 24h) [reset = 0h]

The System Interrupt Status Indexed Clear Register (Figure 9-7) allows clearing the status of an interrupt. The interrupt to clear is the index value written. This clears the System Interrupt Status Raw/Set Register bit of the given index.

Table 9-17. CIC_STATUS_CLR_INDEX_REG Instances

Instance	Physical Address
CIC	0260 0024h

Figure 9-7. CIC_STATUS_CLR_INDEX_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						INDEX									
R-0h																						R/W-0h									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 9-18. CIC_STATUS_CLR_INDEX_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reads return 0 and writes have no effect.
9-0	INDEX	R/W	0h	This field allows clearing the status of an interrupt. Reads return 0. Writes clear the status of the interrupt given in the index value. • 0 = Clear the status of the system interrupt 0 • 1h = Clear the status of the system interrupt 1 • ... • 19Fh = Clear the status of the system interrupt 415 • Others = Reserved

Table 9-19. Register Call Summary for CIC_STATUS_CLR_INDEX_REG

CIC Registers
• CIC Registers: [0]

9.1.4.5 CIC_ENABLE_SET_INDEX_REG Register (Offset = 28h) [reset = 0h]

The System Interrupt Enable Indexed Set Register (Figure 9-8) allows enabling an interrupt. The interrupt to enable is the index value written. This sets the System Interrupt Enable Set Register bit of the given index.

Table 9-20. CIC_ENABLE_SET_INDEX_REG Instances

Instance	Physical Address
CIC	0260 0028h

Figure 9-8. CIC_ENABLE_SET_INDEX_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						INDEX									
R-0h																						R/W-0h									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 9-21. CIC_ENABLE_SET_INDEX_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reads return 0 and writes have no effect.
9-0	INDEX	R/W	0h	This field allows enabling an interrupt. Reads return 0. Writes set the enable of the interrupt given in the index value. • 0 = Enable the system interrupt 0 • 1h = Enable the system interrupt 1 • ... • 19Fh = Enable the system interrupt 415 • Others = Reserved

Table 9-22. Register Call Summary for CIC_ENABLE_SET_INDEX_REG

CIC Registers
• CIC Registers: [0]

9.1.4.6 CIC_ENABLE_CLR_INDEX_REG Register (Offset = 2Ch) [reset = 0h]

The System Interrupt Enable Indexed Clear Register (Figure 9-9) allows disabling an interrupt. The interrupt to disable is the index value written. This clears the Enable Register bit of the given index.

Table 9-23. CIC_ENABLE_CLR_INDEX_REG Instances

Instance	Physical Address
CIC	0260 002Ch

Figure 9-9. CIC_ENABLE_CLR_INDEX_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						INDEX									
R-0h																						R/W-0h									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 9-24. CIC_ENABLE_CLR_INDEX_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reads return 0 and writes have no effect.
9-0	INDEX	R/W	0h	This field allows disabling an interrupt. Reads return 0. Writes clear the enable of the interrupt given in the index value. • 0 = Disable the system interrupt 0 • 1h = Disable the system interrupt 1 • ... • 19Fh = Disable the system interrupt 415 • Others = Reserved

Table 9-25. Register Call Summary for CIC_ENABLE_CLR_INDEX_REG

CIC Registers
• CIC Registers: [0]

9.1.4.7 CIC_HINT_ENABLE_SET_INDEX_REG Register (Offset = 34h) [reset = 0h]

The Host Interrupt Enable Indexed Set Register (Figure 9-10) allows enabling a host interrupt output. The host interrupt to enable is the index value written. This enables the host interrupt output or triggers the output again if already enabled.

Table 9-26. CIC_HINT_ENABLE_SET_INDEX_REG Instances

Instance	Physical Address
CIC	0260 0034h

Figure 9-10. CIC_HINT_ENABLE_SET_INDEX_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						INDEX									
R-0h																						R/W-0h									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 9-27. CIC_HINT_ENABLE_SET_INDEX_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reads return 0 and writes have no effect.
9-0	INDEX	R/W	0h	This field allows enabling a host interrupt output. Reads return 0. Writes set the enable of the host interrupt given in the index value. • 0 = Enable or trigger the host interrupt output 0 • 1h = Enable or trigger the host interrupt output 1 • ... • 67h = Enable or trigger the host interrupt output 103 • Others = Reserved

Table 9-28. Register Call Summary for CIC_HINT_ENABLE_SET_INDEX_REG

CIC Registers
• CIC Registers: [0]

9.1.4.8 CIC_HINT_ENABLE_CLR_INDEX_REG Register (Offset = 38h) [reset = 0h]

The Host Interrupt Enable Indexed Clear Register (Figure 9-11) allows disabling a host interrupt output. The host interrupt to disable is the index value written. This disables the host interrupt output.

Table 9-29. CIC_HINT_ENABLE_CLR_INDEX_REG Instances

Instance	Physical Address
CIC	0260 0038h

Figure 9-11. CIC_HINT_ENABLE_CLR_INDEX_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						INDEX									
R-0h																						R/W-0h									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 9-30. CIC_HINT_ENABLE_CLR_INDEX_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reads return 0 and writes have no effect.
9-0	INDEX	R/W	0h	This field allows disabling a host interrupt output. Reads return 0. Writes clear the enable of the host interrupt given in the index value. • 0 = Disable the host interrupt output 0 • 1h = Disable the host interrupt output 1 • ..... • 67h = Disable the host interrupt output 103 • Others = Reserved

Table 9-31. Register Call Summary for CIC_HINT_ENABLE_CLR_INDEX_REG

CIC Registers
• CIC Registers: [0]

9.1.4.9 CIC_RAW_STATUS_REG0 to CIC_RAW_STATUS_REG12 Register (Offset = 200h to 230h) [reset = 0h]

The System Interrupt Status Raw/Set Registers (Figure 9-12) show the pending raw status of the system interrupts. Software can write to the Status Set Registers to set a system interrupt without a hardware trigger. There are 13 registers (CIC_RAW_STATUS_REG0 to CIC_RAW_STATUS_REG12) used for the 416 system interrupts.

Table 9-32. CIC_RAW_STATUS_REG0 to CIC_RAW_STATUS_REG12 Instances

Instance	Physical Address
CIC	0260 0200h to 0260 0230h

Figure 9-12. CIC_RAW_STATUS_REG0 to CIC_RAW_STATUS_REG12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAW_STATUS																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 9-33. CIC_RAW_STATUS_REG0 to CIC_RAW_STATUS_REG12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RAW_STATUS	R/W	0h	System interrupt raw status and setting. Reads return the raw status. When written (per bit position): • 0 = No effect • 1 = Set the status of the system interrupt

Table 9-34. Register Call Summary for CIC_RAW_STATUS_REG0

CIC Registers
- CIC Registers: [0]
- CIC_RAW_STATUS_REG0 to CIC_RAW_STATUS_REG12 Register (Offset = 200h to 230h) [reset = 0h]: [0]

9.1.4.10 CIC_ENA_STATUS_REG0 to CIC_ENA_STATUS_REG12 Register (Offset = 280h to 2B0h) [reset = 0h]

The System Interrupt Status Enabled/Clear Registers (Figure 9-13) show the pending enabled status of the system interrupts. Software can write to the Status Clear Registers to clear a system interrupt after it has been serviced. If a system interrupt status is not cleared, then another host interrupt may not be triggered or another host interrupt may be triggered incorrectly. There is one bit per system interrupt. There are 13 registers (CIC_ENA_STATUS_REG0 to CIC_ENA_STATUS_REG12) used for the 416 system interrupts.

Table 9-35. CIC_ENA_STATUS_REG0 to CIC_ENA_STATUS_REG12 Instances

Instance	Physical Address
CIC	0260 0280h to 0260 02B0h

Figure 9-13. CIC_ENA_STATUS_REG0 to CIC_ENA_STATUS_REG12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENABLED_STATUS																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 9-36. CIC_ENA_STATUS_REG0 to CIC_ENA_STATUS_REG12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ENABLED_STATUS	R/W	0h	System interrupt enabled status and clearing. Reads return the enabled status (before enabling with the System Interrupt Enable Set Registers). When written (per bit position): • 0 = No effect • 1 = Clear the status of the system interrupt.

Table 9-37. Register Call Summary for CIC_ENA_STATUS_REG0

CIC Registers
- CIC Registers: [0]
- CIC_ENA_STATUS_REG0 to CIC_ENA_STATUS_REG12 Register (Offset = 280h to 2B0h) [reset = 0h]: [0]

9.1.4.11 CIC_ENABLE_REG0 to CIC_ENABLE_REG12 Register (Offset = 300h to 330h) [reset = 0h]

The System Interrupt Enable Set Registers (Figure 9-14) enable system interrupts to trigger outputs. System interrupts that are not enabled do not interrupt the host. There is a bit per system interrupt. There are 13 registers (CIC_ENABLE_REG0 to CIC_ENABLE_REG12) used for the 416 system interrupts.

Table 9-38. CIC_ENABLE_REG0 to CIC_ENABLE_REG12 Instances

Instance	Physical Address
CIC	0260 0300h to 0260 0330h

Figure 9-14. CIC_ENABLE_REG0 to CIC_ENABLE_REG12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENABLE																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 9-39. CIC_ENABLE_REG0 to CIC_ENABLE_REG12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ENABLE	R/W	0h	System interrupt enables. Read returns the enable value: • 0 = Interrupt disabled • 1 = Interrupt enabled When written (per bit position): • 0 = No effect • 1 = Enable the interrupt

Table 9-40. Register Call Summary for CIC_ENABLE_REG0

CIC Registers
- CIC Registers: [0]
- CIC_ENABLE_REG0 to CIC_ENABLE_REG12 Register (Offset = 300h to 330h) [reset = 0h]: [0]

9.1.4.12 CIC_ENABLE_CLR_REG0 to CIC_ENABLE_CLR_REG12 Register (Offset = 380h to 3B0h) [reset = 0h]

The System Interrupt Enable Clear Registers (Figure 9-15) disable system interrupts to map to channels. System interrupts that are not enabled do not interrupt the host. There is one bit per system interrupt. There are 13 registers (CIC_ENABLE_CLR_REG0 to CIC_ENABLE_CLR_REG12) used for the 416 system interrupts.

Table 9-41. CIC_ENABLE_CLR_REG0 to CIC_ENABLE_CLR_REG12 Instances

Instance	Physical Address
CIC	0260 0380h to 0260 03B0h

Figure 9-15. CIC_ENABLE_CLR_REG0 to CIC_ENABLE_CLR_REG12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENABLE																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 9-42. CIC_ENABLE_CLR_REG0 to CIC_ENABLE_CLR_REG12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ENABLE	R/W	0h	System interrupt disables. Read returns the enable value: • 0 = Interrupt disabled • 1 = Interrupt enabled When written (per bit position): • 0 = No effect • 1 = Disable the interrupt

Table 9-43. Register Call Summary for CIC_ENABLE_CLR_REG0

CIC Registers
- CIC Registers: [0]
- CIC_ENABLE_CLR_REG0 to CIC_ENABLE_CLR_REG12 Register (Offset = 380h to 3B0h) [reset = 0h]: [0]

9.1.4.13 CIC_CH_MAP_REG0 to CIC_CH_MAP_REG103 Register (Offset = 400h to 59Ch) [reset = 0h]

The Channel Map Registers (Figure 9-16) define the channel for each system interrupt. There is one register per four system interrupts. There are 104 registers (CIC_CH_MAP_REG0 to CIC_CH_MAP_REG103) used for the 416 system interrupts.

Table 9-44. CIC_CH_MAP_REG0 to CIC_CH_MAP_REG103 Instances

Instance	Physical Address
CIC	0260 0400h to 0260 059Ch

Figure 9-16. CIC_CH_MAP_REG0 to CIC_CH_MAP_REG103 Register

31	30	29	28	27	26	25	24
CH3_MAP							
R/W-0h							
23	22	21	20	19	18	17	16
CH2_MAP							
R/W-0h							
15	14	13	12	11	10	9	8
CH1_MAP							
R/W-0h							
7	6	5	4	3	2	1	0
CH0_MAP							
R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 9-45. CIC_CH_MAP_REG0 to CIC_CH_MAP_REG103 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	CH3_MAP	R/W	0h	Set the channel for the system interrupt N + 3
23-16	CH2_MAP	R/W	0h	Set the channel for the system interrupt N + 2
15-8	CH1_MAP	R/W	0h	Set the channel for the system interrupt N + 1
7-0	CH0_MAP	R/W	0h	Set the channel for the system interrupt N

Table 9-46. Register Call Summary for CIC_CH_MAP_REG0

CIC Registers
- CIC Registers: [0]
- CIC_CH_MAP_REG0 to CIC_CH_MAP_REG103 Register (Offset = 400h to 59Ch) [reset = 0h]: [0]

9.1.4.14 CIC_HINT_MAP_REG0 to CIC_HINT_MAP_REG25 Register (Offset = 800h to 864h) [reset = 0h]

The Host Interrupt Map Registers (Figure 9-17) define the host interrupt for each channel. There is one register per four channels. It is read-only because the channel-to-host mapping is fixed. There are 26 registers (CIC_HINT_MAP_REG0 to CIC_HINT_MAP_REG25) used for the 104 channels.

Table 9-47. CIC_HINT_MAP_REG0 to CIC_HINT_MAP_REG25 Instances

Instance	Physical Address
CIC	0260 0800h to 0260 0864h

Figure 9-17. CIC_HINT_MAP_REG0 to CIC_HINT_MAP_REG25 Register

31	30	29	28	27	26	25	24
HINT3_MAP							
R-0h							
23	22	21	20	19	18	17	16
HINT2_MAP							
R-0h							
15	14	13	12	11	10	9	8
HINT1_MAP							
R-0h							
7	6	5	4	3	2	1	0
HINT0_MAP							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 9-48. CIC_HINT_MAP_REG0 to CIC_HINT_MAP_REG25 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	HINT3_MAP	R	0h	Set the host interrupt for channel N + 3
23-16	HINT2_MAP	R	0h	Set the host interrupt for channel N + 2
15-8	HINT1_MAP	R	0h	Set the host interrupt for channel N + 1
7-0	HINT0_MAP	R	0h	Set the host interrupt for channel N

Table 9-49. Register Call Summary for CIC_HINT_MAP_REG0

CIC Registers
- CIC Registers: [0]
- CIC_HINT_MAP_REG0 to CIC_HINT_MAP_REG25 Register (Offset = 800h to 864h) [reset = 0h]: [0]

9.1.4.15 CIC_ENABLE_HINT_REG0 to CIC_ENABLE_HINT_REG3 Register (Offset = 1500h to 150Ch) [reset = 0h]

The Host Interrupt Enable Registers (Figure 9-18) enable or disable individual host interrupts. These work separately from the global enables. There is one bit per host interrupt. These bits are updated when writing to the Host Interrupt Enable Index Set and Host Interrupt Enable Index Clear registers. There are 4 registers (CIC_ENABLE_HINT_REG0 to CIC_ENABLE_HINT_REG3) used for the 104 host interrupts.

Table 9-50. CIC_ENABLE_HINT_REG0 to CIC_ENABLE_HINT_REG3 Instances

Instance	Physical Address
CIC	0260 1500h to 0260 150Ch

Figure 9-18. CIC_ENABLE_HINT_REG0 to CIC_ENABLE_HINT_REG3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENABLES																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 9-51. CIC_ENABLE_HINT_REG0 to CIC_ENABLE_HINT_REG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ENABLES	R/W	0h	This field enables the host interrupts. When written (per bit position): • 0 = Disable the host interrupt • 1 = Enable the host interrupt

Table 9-52. Register Call Summary for CIC_ENABLE_HINT_REG0

CIC Registers
- CIC_ENABLE_HINT_REG0 to CIC_ENABLE_HINT_REG3 Register (Offset = 1500h to 150Ch) [reset = 0h]: [0]
- CIC Registers: [0]

9.2 Interrupt Sources

NOTE: This section covers only the mapping of system interrupts to CIC inputs. For information about interrupt mapping of other device hosts (Arm Subsystem, DSP Subsystem, etc), see their respective chapters.

NOTE: ELM is not supported on this family of devices.

Table 9-53 shows the system events mapping to CIC inputs.

Table 9-53. CIC Input Events Mapping

Event No.	Event Source Name	Description
0	PRU-ICSS_0_HOST_INT0	PRU-ICSS_0 host interrupt 0
1	PRU-ICSS_0_HOST_INT1	PRU-ICSS_0 host interrupt 1
2	PRU-ICSS_0_HOST_INT2	PRU-ICSS_0 host interrupt 2
3	PRU-ICSS_0_HOST_INT3	PRU-ICSS_0 host interrupt 3
4	PRU-ICSS_0_HOST_INT4	PRU-ICSS_0 host interrupt 4
5	RESERVED	Reserved
6	PRU-ICSS_0_HOST_INT6	PRU-ICSS_0 host interrupt 6
7	PRU-ICSS_0_HOST_INT7	PRU-ICSS_0 host interrupt 7
8	PRU-ICSS_1_HOST_INT0	PRU-ICSS_1 host interrupt 0
9	PRU-ICSS_1_HOST_INT1	PRU-ICSS_1 host interrupt 1
10	PRU-ICSS_1_HOST_INT2	PRU-ICSS_1 host interrupt 2
11	PRU-ICSS_1_HOST_INT3	PRU-ICSS_1 host interrupt 3
12	PRU-ICSS_1_HOST_INT4	PRU-ICSS_1 host interrupt 4
13	RESERVED	Reserved
14	PRU-ICSS_1_HOST_INT6	PRU-ICSS_1 host interrupt 6
15	PRU-ICSS_1_HOST_INT7	PRU-ICSS_1 host interrupt 7
16	MSMC_MPF_ERROR0	MSMC memory protection fault indicator for system master PrivID = 0 (C66x DSP)
17	MSMC_MPF_ERROR1	MSMC memory protection fault indicator for system master PrivID = 1 (ARMSS)
18	MSMC_MPF_ERROR2	MSMC memory protection fault indicator for system master PrivID = 2 (PRU-ICSS_0)
19	MSMC_MPF_ERROR3	MSMC memory protection fault indicator for system master PrivID = 3 (PRU-ICSS_1)
20	MSMC_MPF_ERROR4	MSMC memory protection fault indicator for system master PrivID = 4 (NSS)
21	MSMC_MPF_ERROR5	MSMC memory protection fault indicator for system master PrivID = 5 (PCIE)
22	MSMC_MPF_ERROR6	MSMC memory protection fault indicator for system master PrivID = 6 (USB[1:0])
23	MSMC_MPF_ERROR7	MSMC memory protection fault indicator for system master PrivID = 7 (not assigned)
24	MSMC_MPF_ERROR8	MSMC memory protection fault indicator for system master PrivID = 8 (MLB)
25	MSMC_MPF_ERROR9	MSMC memory protection fault indicator for system master PrivID = 9 (PMMC)
26	MSMC_MPF_ERROR10	MSMC memory protection fault indicator for system master PrivID = 10 (DSS)
27	MSMC_MPF_ERROR11	MSMC memory protection fault indicator for system master PrivID = 11 (MMC[1:0])
28	MSMC_MPF_ERROR12	MSMC memory protection fault indicator for system master PrivID = 12 (DAP)
29	MSMC_MPF_ERROR13	MSMC memory protection fault indicator for system master PrivID = 13 (reserved)
30	MSMC_MPF_ERROR14	MSMC memory protection fault indicator for system master PrivID = 14 (not assigned)
31	MSMC_MPF_ERROR15	MSMC memory protection fault indicator for system master PrivID = 15 (not assigned)

Table 9-53. CIC Input Events Mapping (continued)

Event No.	Event Source Name	Description
32	MSMC_DEDC_CERROR	MSMC correctable (1-bit) soft error detected on SRAM read
33	MSMC_SCRUB_CERROR	MSMC correctable (1-bit) soft error detected during scrub cycle
34	MSMC_DEDC_NC_ERROR	MSMC non-correctable (2-bit) soft error detected on SRAM read
35	MSMC_SCRUB_NC_ERROR	MSMC non-correctable (2-bit) soft error detected during scrub cycle
36	DBGTBR_DMAINT	System trace buffer DMA event
37	DBGTBR_ACQCOMP	System trace buffer acquisition complete interrupt
38	BOOTCFG_INT	BOOT_CFG boot error interrupt
39	RESERVED	Reserved
40	MPU_0_INT	MPU_0 violation interrupt
41	MPU_1_INT	MPU_1 violation interrupt
42	MPU_2_INT	MPU_2 violation interrupt
43	MPU_3_INT	MPU_3 violation interrupt
44	MPU_4_INT	MPU_4 violation interrupt
45	MPU_5_INT	MPU_5 violation interrupt
46	MPU_6_INT	MPU_6 violation interrupt
47	MPU_7_INT	MPU_7 violation interrupt
48	MPU_8_INT	MPU_8 violation interrupt
49	MPU_9_INT	MPU_9 violation interrupt
50	MPU_10_INT	MPU_10 violation interrupt
51	MPU_11_INT	MPU_11 violation interrupt
52	MPU_12_INT	MPU_12 violation interrupt
53	MPU_13_INT	MPU_13 violation interrupt
54	MPU_14_INT	MPU_14 violation interrupt
55	MPU_15_INT	MPU_15 violation interrupt
56	TRACER_MSMC_0_INT	CP_TRACER sliding time window interrupt for MSMC SRAM bank0
57	TRACER_DDR_INT	CP_TRACER sliding time window interrupt for DDR
58	TRACER_CORE_INT	CP_TRACER sliding time window interrupt for CORE
59	TRACER_PCIE_EVT	CP_TRACER sliding time window interrupt for PCIE
60	RESERVED	Reserved
61	TRACER_MCASP_MCBSP_INT	CP_TRACER sliding time window interrupt for MCASP/MCBSP
62	TRACER_GPMC_MMC_QSPI_INT	CP_TRACER sliding time window interrupt for GPMC/MMC/QSPI
63	TRACER_EDMACC_01_INT	CP_TRACER sliding time window interrupt for EDMACC_0/1
64	TRACER_CFG_INT	CP_TRACER sliding time window interrupt for CFG
65	TRACER_ALWAYSON_CFG_INT	CP_TRACER sliding time window interrupt for ALWAYSON_CFG
66	TRACER_GIC_INT	CP_TRACER sliding time window interrupt for ARM_GIC
67	TRACER_CIC_INT	CP_TRACER sliding time window interrupt for CIC
68	TRACER_ROM_SPI_INT	CP_TRACER sliding time window interrupt for ROM/SPI
69	TRACER_ALWAYSON_INT	CP_TRACER sliding time window interrupt for ALWAYSON_MAIN
70	TRACER_PRU-ICSS	CP_TRACER sliding time window interrupt for PRU-ICSS
71	RESERVED	Reserved
72	TIMER_0_INTL	TIMER_0 low interrupt
73	TIMER_0_INTH	TIMER_0 high interrupt
74	TIMER_1_INTL	TIMER_1 low interrupt
75	TIMER_1_INTH	TIMER_1 high interrupt
76	TIMER_2_INTL	TIMER_2 low interrupt
77	TIMER_2_INTH	TIMER_2 high interrupt
78	TIMER_3_INTL	TIMER_3 low interrupt

Table 9-53. CIC Input Events Mapping (continued)

Event No.	Event Source Name	Description
79	TIMER_3_INTH	TIMER_3 high interrupt
80	TIMER_4_INTL	TIMER_4 low interrupt
81	TIMER_4_INTH	TIMER_4 high interrupt
82	TIMER_5_INTL	TIMER_5 low interrupt
83	TIMER_5_INTH	TIMER_5 high interrupt
84	TIMER_6_INTL	TIMER_6 low interrupt
85	TIMER_6_INTH	TIMER_6 high interrupt
86	RESERVED	Reserved
87	RESERVED	Reserved
88	DCAN_0_INT0	DCAN_0 error/data/MO interrupt
89	DCAN_0_INT1	DCAN_0 message object (MO) interrupt
90	DCAN_0_ERR_INT	DCAN_0 parity error interrupt
91	DCAN_1_INT0	DCAN_1 error/data/MO interrupt
92	DCAN_1_INT1	DCAN_1 message object (MO) interrupt
93	DCAN_1_ERR_INT	DCAN_1 parity error interrupt
94	QSPI_INT	QSPI interrupt
95	DSS_INT	DSS interrupt
96	USB_0_INT00	USBSS_0 event ring 0 interrupt
97	USB_0_INT01	USBSS_0 event ring 1 interrupt
98	USB_0_INT02	USBSS_0 event ring 2 interrupt
99	USB_0_INT03	USBSS_0 event ring 3 interrupt
100	USB_0_INT04	USBSS_0 event ring 4 interrupt
101	USB_0_INT05	USBSS_0 event ring 5 interrupt
102	USB_0_INT06	USBSS_0 event ring 6 interrupt
103	USB_0_INT07	USBSS_0 event ring 7 interrupt
104	USB_0_INT08	USBSS_0 event ring 8 interrupt
105	USB_0_INT09	USBSS_0 event ring 9 interrupt
106	USB_0_INT10	USBSS_0 event ring 10 interrupt
107	USB_0_INT11	USBSS_0 event ring 11 interrupt
108	USB_0_INT12	USBSS_0 event ring 12 interrupt
109	USB_0_INT13	USBSS_0 event ring 13 interrupt
110	USB_0_INT14	USBSS_0 event ring 14 interrupt
111	USB_0_INT15	USBSS_0 event ring 15 interrupt
112	USB_1_INT00	USBSS_1 event ring 0 interrupt
113	USB_1_INT01	USBSS_1 event ring 1 interrupt
114	USB_1_INT02	USBSS_1 event ring 2 interrupt
115	USB_1_INT03	USBSS_1 event ring 3 interrupt
116	USB_1_INT04	USBSS_1 event ring 4 interrupt
117	USB_1_INT05	USBSS_1 event ring 5 interrupt
118	USB_1_INT06	USBSS_1 event ring 6 interrupt
119	USB_1_INT07	USBSS_1 event ring 7 interrupt
120	USB_1_INT08	USBSS_1 event ring 8 interrupt
121	USB_1_INT09	USBSS_1 event ring 9 interrupt
122	USB_1_INT10	USBSS_1 event ring 10 interrupt
123	USB_1_INT11	USBSS_1 event ring 11 interrupt
124	USB_1_INT12	USBSS_1 event ring 12 interrupt
125	USB_1_INT13	USBSS_1 event ring 13 interrupt

Table 9-53. CIC Input Events Mapping (continued)

Event No.	Event Source Name	Description
126	USB_1_INT14	USBSS_1 event ring 14 interrupt
127	USB_1_INT15	USBSS_1 event ring 15 interrupt
128	USB_0_OABSINT	USBSS_0 OABS interrupt
129	USB_0_MISCINT	USBSS_0 miscellaneous interrupt
130	USB_1_OABSINT	USBSS_1 OABS interrupt
131	USB_1_MISCINT	USBSS_1 miscellaneous interrupt
132	UART_0_UARTINT	UART_0 interrupt
133	UART_1_UARTINT	UART_1 interrupt
134	UART_2_UARTINT	UART_2 interrupt
135	RESERVED	Reserved
136	EDMACC_0_REGION_0_INT	EDMACC_0 region 0 DMA completion interrupt
137	EDMACC_0_REGION_1_INT	EDMACC_0 region 1 DMA completion interrupt
138	EDMACC_0_REGION_2_INT	EDMACC_0 region 2 DMA completion interrupt
139	EDMACC_0_REGION_3_INT	EDMACC_0 region 3 DMA completion interrupt
140	EDMACC_0_REGION_4_INT	EDMACC_0 region 4 DMA completion interrupt
141	EDMACC_0_REGION_5_INT	EDMACC_0 region 5 DMA completion interrupt
142	EDMACC_0_REGION_6_INT	EDMACC_0 region 6 DMA completion interrupt
143	EDMACC_0_REGION_7_INT	EDMACC_0 region 7 DMA completion interrupt
144	EDMACC_1_REGION_0_INT	EDMACC_1 region 0 DMA completion interrupt
145	EDMACC_1_REGION_1_INT	EDMACC_1 region 1 DMA completion interrupt
146	EDMACC_1_REGION_2_INT	EDMACC_1 region 2 DMA completion interrupt
147	EDMACC_1_REGION_3_INT	EDMACC_1 region 3 DMA completion interrupt
148	EDMACC_1_REGION_4_INT	EDMACC_1 region 4 DMA completion interrupt
149	EDMACC_1_REGION_5_INT	EDMACC_1 region 5 DMA completion interrupt
150	EDMACC_1_REGION_6_INT	EDMACC_1 region 6 DMA completion interrupt
151	EDMACC_1_REGION_7_INT	EDMACC_1 region 7 DMA completion interrupt
152	EDMACC_0_MPINT	EDMACC_0 memory protection interrupt
153	EDMACC_0_ERRINT	EDMACC_0 error interrupt
154	EDMACC_0_GINT	EDMACC_0 global completion interrupt
155	EDMACC_1_MPINT	EDMACC_1 memory protection interrupt
156	EDMACC_1_ERRINT	EDMACC_1 error interrupt
157	EDMACC_1_GINT	EDMACC_1 global completion interrupt
158	EDMACC_0_TC_0_INT	EDMACC_0_TC0 completion interrupt
159	EDMACC_0_TC_1_INT	EDMACC_0_TC1 completion interrupt
160	EDMACC_0_TC_0_ERRINT	EDMACC_0_TC0 error interrupt
161	EDMACC_0_TC_1_ERRINT	EDMACC_0_TC1 error interrupt
162	EDMACC_1_TC_0_INT	EDMACC_1_TC0 completion interrupt
163	EDMACC_1_TC_1_INT	EDMACC_1_TC1 completion interrupt
164	EDMACC_1_TC_0_ERRINT	EDMACC_1_TC0 error interrupt
165	EDMACC_1_TC_1_ERRINT	EDMACC_1_TC1 error interrupt
166	ARM_TBR_DMA	ARMSS trace buffer DMA event
167	ARM_TBR_ACQ	ARMSS trace buffer acquisition complete interrupt
168	ARM_NCNTPNSIRQ0	ARMSS NCNTPNSIRQ0
169	ARM_NCNTVIRQ0	ARMSS NCNTVIRQ0
170	RESERVED	Reserved
171	MPU_16_INT	MPU_16 violation interrupt
172	McASP_0_XINT	McASP_0 transmit interrupt

Table 9-53. CIC Input Events Mapping (continued)

Event No.	Event Source Name	Description
173	McASP_0_RINT	McASP_0 receive interrupt
174	McASP_1_XINT	McASP_1 transmit nterrupt
175	McASP_1_RINT	McASP_1 receive interrupt
176	McASP_2_XINT	McASP_2 transmit interrupt
177	McASP_2_RINT	McASP_2 receive interrupt
178	McBSP_XINT	McBSP transmit interrupt
179	McBSP_RINT	McBSP receive interrupt
180	I2C_0_INT	I2C_0 interrupt
181	I2C_1_INT	I2C_1 interrupt
182	I2C_2_INT	I2C_2 interrupt
183	RESERVED	Reserved
184	ASRC_STREAM_IN_INT	ASRC input FIFO interrupt
185	ASRC_STREAM_OUT_INT	ASRC output FIFO interrupt
186	ASRC_STREAM_ERR_INT	ASRC error interrupt
187	ASRC_GROUP_IN_INT	ASRC Group input FIFO interrupt
188	ASRC_GROUP_OUT_INT	ASRC Group output FIFO interrupt
189	MLB_DMA_CH_INT0	MLB DMA channel interrupt 0
190	MLB_DMA_CH_INT1	MLB DMA channel interrupt 1
191	MLB_SYS_INT	MLB system interrupt
192	MMC_SD_0_INT	MMC/SD_0 interrupt
193	MMC_SD_1_INT	MMC/SD_1 interrupt
194	RESERVED	Reserved
195	RESERVED	Reserved
196	GPMC_INT	GPMC interrupt
197	ELM_INT	ELM error location process complete interrupt
198	DDR3_ERR	EMIF error interrupt
199	RESERVED	Reserved
200	EPWM_0_INT	ePWM_0 event trigger (ET) interrupt
201	EPWM_1_INT	ePWM_1 event trigger (ET) interrupt
202	EPWM_2_INT	ePWM_2 event trigger (ET) interrupt
203	EPWM_3_INT	ePWM_3 event trigger (ET) interrupt
204	EPWM_4_INT	ePWM_4 event trigger (ET) interrupt
205	EPWM_5_INT	ePWM_5 event trigger (ET) interrupt
206	EPWM_0_TRIP_ZONE	ePWM_0 tripzone (TZ) interrupt
207	EPWM_1_TRIP_ZONE	ePWM_1 tripzone (TZ) interrupt
208	EPWM_2_TRIP_ZONE	ePWM_2 tripzone (TZ) interrupt
209	EPWM_3_TRIP_ZONE	ePWM_3 tripzone (TZ) interrupt
210	EPWM_4_TRIP_ZONE	ePWM_4 tripzone (TZ) interrupt
211	EPWM_5_TRIP_ZONE	ePWM_5 tripzone (TZ) interrupt
212	EQEP_0_INT	eQEP_0 interrupt
213	EQEP_1_INT	eQEP_1 interrupt
214	EQEP_2_INT	eQEP_2 interrupt
215	ECAP_0_INT	eCAP_0 interrupt
216	ECAP_1_INT	eCAP_1 interrupt
217	RESERVED	Reserved
218	TETB_HFULLINT0	TETB half-full flag
219	TETB_FULLINT0	TETB full flag

Table 9-53. CIC Input Events Mapping (continued)

Event No.	Event Source Name	Description
220	TETB_ACQINT0	TETB acquisition complete interrupt
221	TETB_OVFLINT0	TETB overflow interrupt
222	TETB_UNFLINT0	TETB underflow interrupt
223	PSC_ALLINT	Power & Sleep Controller (PSC) interrupt
224	PCIE_INT0	PCIe interrupt 0
225	PCIE_INT1	PCIe interrupt 1
226	PCIE_INT2	PCIe interrupt 2
227	PCIE_INT3	PCIe interrupt 3
228	PCIE_INT4	PCIe interrupt 4
229	PCIE_INT5	PCIe interrupt 5
230	PCIE_INT6	PCIe interrupt 6
231	PCIE_INT7	PCIe interrupt 7
232	PCIE_INT8	PCIe interrupt 8
233	PCIE_INT9	PCIe interrupt 9
234	PCIE_INT10	PCIe interrupt 10
235	PCIE_INT11	PCIe interrupt 11
236	PCIE_INT12	PCIe interrupt 12
237	PCIE_INT13	PCIe interrupt 13
238	PCIE_ECC_ERROR	PCIe ECC error interrupt
239	QSPI_ECC_ERR	QSPI ECC interrupt
255–240	RESERVED	Reserved
256	SPI_0_INT0	SPI_0 Level 0 interrupt
257	SPI_0_INT1	SPI_0 Level 1 interrupt
258	SPI_1_INT0	SPI_1 Level 0 interrupt
259	SPI_1_INT1	SPI_1 Level 1 interrupt
260	SPI_2_INT0	SPI_2 Level 0 interrupt
261	SPI_2_INT1	SPI_2 Level 1 interrupt
262	SPI_3_INT0	SPI_3 Level 0 interrupt
263	SPI_3_INT1	SPI_3 Level 1 interrupt
264	NSS_CDMA_STARVE_INT	NSS CDMA buffer starvation interrupt
265	NSS_SWITCH_STAT_INT0	NSS status pending interrupt
266	NSS_SWITCH_STAT_INT1	NSS status pending interrupt
267	NSS_NAVSS_ECC_INT	NSS NAVSS ECC error interrupt
268	NSS_SA_UL_ECC_INT	NSS SA_UL ECC error interrupt
269	NSS_SWITCH_ECC_INT	NSS ENET switch ECC error interrupt
270	NSS_MDIO_LINK_INT0	NSS MDIO link interrupt 0
271	NSS_CPTS_EVENT_INT	NSS CPTS event pending interrupt
272	NSS_MDIO_USER_INT0	NSS MDIO user interrupt 0
273	NSS_MDIO_USER_INT1	NSS MDIO user interrupt 1
274	RESERVED	Reserved
275	RESERVED	Reserved
276	PMMC_INT0	PMMC interrupt 0
277	PMMC_INT1	PMMC interrupt 1
278	PMMC_FAULTDET_INT	PMMC safety interrupt
279	RESERVED	Reserved
280	PMMCTBR_ACQCOMP	PMMC trace buffer acquisition complete interrupt
281	PMMCTBR_DMAINT	PMMC trace buffer DMA event

Table 9-53. CIC Input Events Mapping (continued)

Event No.	Event Source Name	Description
282	RESERVED	Reserved
283	MSGMGR_PROXY_ERROR	Message Manager proxy error interrupt
284	MSGMGR_FREEINDEX_ERROR	Message Manager no free index available interrupt
285	MSGMGR_ECC_ERROR	Message Manager ECC error interrupt
286	NSS_QPEND0	NSS transmit queue 0 pending interrupt
287	NSS_QPEND1	NSS transmit queue 1 pending interrupt
288	NSS_QPEND2	NSS transmit queue 2 pending interrupt
289	NSS_QPEND3	NSS transmit queue 3 pending interrupt
290	NSS_QPEND4	NSS transmit queue 4 pending interrupt
291	NSS_QPEND5	NSS transmit queue 5 pending interrupt
292	NSS_QPEND6	NSS transmit queue 6 pending interrupt
293	NSS_QPEND7	NSS transmit queue 7 pending interrupt
294	NSS_QPEND8	NSS transmit queue 8 pending interrupt
295	NSS_QPEND9	NSS transmit queue 9 pending interrupt
296	NSS_QPEND10	NSS transmit queue 10 pending interrupt
297	NSS_QPEND11	NSS transmit queue 11 pending interrupt
298	NSS_QPEND12	NSS transmit queue 12 pending interrupt
299	NSS_QPEND13	NSS transmit queue 13 pending interrupt
300	NSS_QPEND14	NSS transmit queue 14 pending interrupt
301	NSS_QPEND15	NSS transmit queue 15 pending interrupt
302	NSS_QPEND16	NSS transmit queue 16 pending interrupt
303	NSS_QPEND17	NSS transmit queue 17 pending interrupt
304	NSS_QPEND18	NSS transmit queue 18 pending interrupt
305	NSS_QPEND19	NSS transmit queue 19 pending interrupt
306	NSS_QPEND20	NSS transmit queue 20 pending interrupt
307	NSS_QPEND21	NSS transmit queue 21 pending interrupt
308	NSS_QPEND22	NSS transmit queue 22 pending interrupt
309	NSS_QPEND23	NSS transmit queue 23 pending interrupt
310	NSS_QPEND24	NSS transmit queue 24 pending interrupt
311	NSS_QPEND25	NSS transmit queue 25 pending interrupt
312	NSS_QPEND26	NSS transmit queue 26 pending interrupt
313	NSS_QPEND27	NSS transmit queue 27 pending interrupt
314	NSS_QPEND28	NSS transmit queue 28 pending interrupt
315	NSS_QPEND29	NSS transmit queue 29 pending interrupt
316	NSS_QPEND30	NSS transmit queue 30 pending interrupt
317	NSS_QPEND31	NSS transmit queue 31 pending interrupt
318	NSS_QPEND56	NSS transmit queue 56 pending interrupt
319	NSS_QPEND57	NSS transmit queue 57 pending interrupt
320	NSS_QPEND58	NSS transmit queue 58 pending interrupt
321	NSS_QPEND59	NSS transmit queue 59 pending interrupt
322	NSS_QPEND60	NSS transmit queue 60 pending interrupt
323	NSS_QPEND61	NSS transmit queue 61 pending interrupt
324	NSS_QPEND62	NSS transmit queue 62 pending interrupt
325	NSS_QPEND63	NSS transmit queue 63 pending interrupt
326	NSS_QPEND64	NSS transmit queue 64 pending interrupt
327	NSS_QPEND65	NSS transmit queue 65 pending interrupt
328	NSS_QPEND66	NSS transmit queue 66 pending interrupt

Table 9-53. CIC Input Events Mapping (continued)

Event No.	Event Source Name	Description
329	NSS_QPEND67	NSS transmit queue 67 pending interrupt
330	NSS_QPEND68	NSS transmit queue 68 pending interrupt
331	NSS_QPEND69	NSS transmit queue 69 pending interrupt
332	NSS_QPEND70	NSS transmit queue 70 pending interrupt
333	NSS_QPEND71	NSS transmit queue 71 pending interrupt
334	RESERVED	Reserved
335	RESERVED	Reserved
336	RESERVED	Reserved
337	RESERVED	Reserved
338	PWM_SOCA	Start of conversion A event
339	PWM_SOCB	Start of conversion B event
340	SEM_INT11	Semaphore Master 11 grant interrupt
341	SEM_INT12	Semaphore Master 12 grant interrupt
342	SEM_INT13	Semaphore Master 13 grant interrupt
343	SEM_INT14	Semaphore Master 14 grant interrupt
344	SEM_ERR11	Semaphore Master 11 error interrupt
345	SEM_ERR12	Semaphore Master 12 error interrupt
346	SEM_ERR13	Semaphore Master 13 error interrupt
347	SEM_ERR14	Semaphore Master 14 error interrupt
369–348	RESERVED	Reserved
370	GPIO_0_BANK0_INT	GPIO_0 bank 0 interrupt
371	GPIO_0_BANK1_INT	GPIO_0 bank 1 interrupt
372	GPIO_0_BANK2_INT	GPIO_0 bank 2 interrupt
373	GPIO_0_BANK3_INT	GPIO_0 bank 3 interrupt
374	GPIO_0_BANK4_INT	GPIO_0 bank 4 interrupt
375	GPIO_0_BANK5_INT	GPIO_0 bank 5 interrupt
376	GPIO_0_BANK6_INT	GPIO_0 bank 6 interrupt
377	GPIO_0_BANK7_INT	GPIO_0 bank 7 interrupt
378	GPIO_0_BANK8_INT	GPIO_0 bank 8 interrupt
379	GPIO_1_BANK0_INT	GPIO_1 bank 0 interrupt
380	GPIO_1_BANK1_INT	GPIO_1 bank 1 interrupt
381	GPIO_1_BANK2_INT	GPIO_1 bank 2 interrupt
382	GPIO_1_BANK3_INT	GPIO_1 bank 3 interrupt
383	GPIO_1_BANK4_INT	GPIO_1 bank 4 interrupt
384	GPIOMUX_INT0	GPIOMUX0 output
385	GPIOMUX_INT1	GPIOMUX1 output
386	GPIOMUX_INT2	GPIOMUX2 output
387	GPIOMUX_INT3	GPIOMUX3 output
388	GPIOMUX_INT4	GPIOMUX4 output
389	GPIOMUX_INT5	GPIOMUX5 output
390	GPIOMUX_INT6	GPIOMUX6 output
391	GPIOMUX_INT7	GPIOMUX7 output
392	GPIOMUX_INT8	GPIOMUX8 output
393	GPIOMUX_INT9	GPIOMUX9 output
394	GPIOMUX_INT10	GPIOMUX10 output
395	GPIOMUX_INT11	GPIOMUX11 output
396	GPIOMUX_INT12	GPIOMUX12 output

Table 9-53. CIC Input Events Mapping (continued)

Event No.	Event Source Name	Description
397	GPIOMUX_INT13	GPIOMUX13 output
398	GPIOMUX_INT14	GPIOMUX14 output
399	GPIOMUX_INT15	GPIOMUX15 output
400	GPIOMUX_INT16	GPIOMUX16 output
401	GPIOMUX_INT17	GPIOMUX17 output
402	GPIOMUX_INT18	GPIOMUX18 output
403	GPIOMUX_INT19	GPIOMUX19 output
404	GPIOMUX_INT20	GPIOMUX20 output
405	GPIOMUX_INT21	GPIOMUX21 output
406	GPIOMUX_INT22	GPIOMUX22 output
407	GPIOMUX_INT23	GPIOMUX23 output
408	GPIOMUX_INT24	GPIOMUX24 output
409	GPIOMUX_INT25	GPIOMUX25 output
410	GPIOMUX_INT26	GPIOMUX26 output
411	GPIOMUX_INT27	GPIOMUX27 output
412	GPIOMUX_INT28	GPIOMUX28 output
413	GPIOMUX_INT29	GPIOMUX29 output
414	GPIOMUX_INT30	GPIOMUX30 output
415	GPIOMUX_INT31	GPIOMUX31 output

Enhanced Direct Memory Access (EDMA) Controller

This chapter describes the Enhanced Direct Memory Access (EDMA) controllers in the device.

Topic **Page**

10.1 EDMA Overview

10.1.1 Introduction

The primary purpose of the Enhanced Direct Memory Access (EDMA) controller is to service user-programmed data transfers between two memory-mapped slave endpoints on the device.

Typical usage of the EDMA controller includes:

- Servicing software-driven paging transfers (e.g., data movement between external memory [such as SDRAM] and internal memory [such as DSP L2 SRAM])
- Servicing event-driven peripherals, such as a serial port
- Performing sorting or sub-frame extraction of various data structures
- Offloading data transfers from the main device CPUs, such as the C66x DSP CorePac or the Arm CorePac

The EDMA controller consists of two major principle blocks:

- EDMA Channel Controller
- EDMA Transfer Controller(s)

The **EDMA Channel Controller (EDMACC)** serves as the user interface for the EDMA controller. The EDMACC includes parameter RAM (PaRAM), channel control registers, and interrupt control registers. The EDMACC serves to prioritize incoming software requests or events from peripherals and submits transfer requests (TR) to the EDMA transfer controller.

The **EDMA Transfer Controller (EDMATC)** is responsible for data movement. The transfer request packets (TRP) submitted by the EDMACC contain the transfer context, based on which the transfer controller issues read/write commands to the source and destination addresses programmed for a given transfer.

There are two EDMA controllers present on this device:

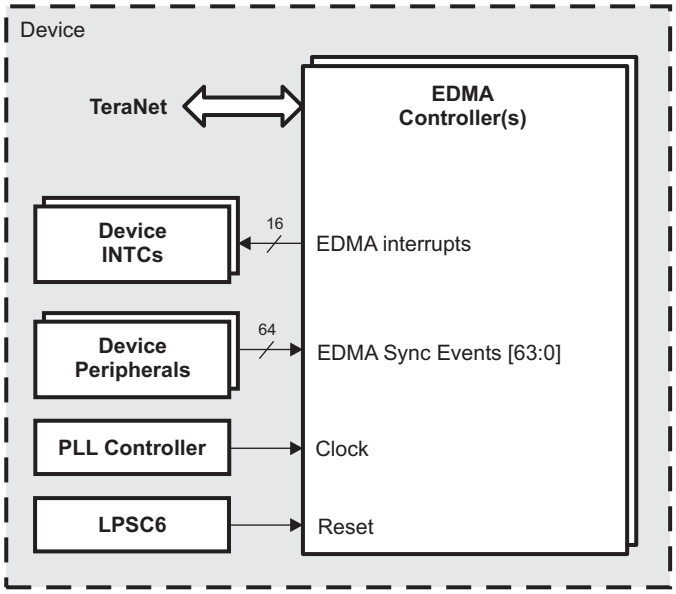
- **EDMA_0**, integrating:
  - 1 Channel Controller, referenced as: **EDMACC_0**
  - 2 Transfer Controllers, referenced as: **EDMACC_0_TC_0** (or **EDMATC_0**) and **EDMACC_0_TC_1** (or **EDMATC_1**)
- **EDMA_1**, integrating:
  - 1 Channel Controller, referenced as: **EDMACC_1**
  - 2 Transfer Controllers, referenced as: **EDMACC_1_TC_0** (or **EDMATC_2**) and **EDMACC_1_TC_1** (or **EDMATC_3**)

The two EDMA channel controllers (EDMACC_0 and EDMACC_1) are functionally identical. For simplification, the unified name **EDMACC** shall be regularly used throughout this chapter when referring to **EDMA Channel Controllers** functionality and features.

The four EDMA transfer controllers (EDMACC_0_TC_0, EDMACC_0_TC_1, EDMACC_1_TC_0 and EDMACC_1_TC_1) are functionally identical. For simplification, the unified name **EDMATC** shall be regularly used throughout this chapter when referring to **EDMA Transfer Controllers** functionality and features.

Figure 10-1 shows an overview of the EDMA controllers.

Figure 10-1. EDMA Controllers Overview



10.1.2 EDMA Controllers Features

Each **EDMACC** has the following features:

- Fully orthogonal transfer description
  - 3 transfer dimensions:
    - Array (multiple bytes)
    - Frame (multiple arrays)
    - Block (multiple frames)
  - Single event can trigger transfer of array, frame, or entire block
  - Independent indexes on source and destination
- Flexible transfer definition
  - Increment or constant addressing modes
  - Linking mechanism allows automatic PaRAM set update
  - Chaining allows multiple transfers to execute with one event
- 64 DMA channels
  - Channels triggered by either:
    - Event synchronization
    - Manual synchronization (CPU write to event set register)
    - Chain synchronization (completion of one transfer triggers another transfer)
  - Support for programmable DMA Channel to PaRAM mapping
- 8 Quick DMA (QDMA) channels
  - QDMA channels are triggered automatically upon writing to PaRAM set entry
  - Support for programmable QDMA channel to PaRAM mapping
- 512 PaRAM sets
  - Each PaRAM set can be used for a DMA channel, QDMA channel, or link set
- 2 transfer controllers/event queues
  - 16 event entries per event queue
- Interrupt generation based on:

  - Transfer completion
  - Error conditions
- Debug visibility
  - Queue water marking/threshold
  - Error and status recording to facilitate debug
- Memory protection support
  - Proxied memory protection for TR submission
  - Active memory protection for accesses to PaRAM and registers

Each **EDMATC** has the following features:

- Supports 2-dimensional (2D) transfers with independent indexes on source and destination (EDMACC manages the 3rd dimension)
- Up to 4 in-flight transfer requests (TR)
- Programmable priority levels
- Support for increment or constant addressing mode transfers
- Interrupt and error support
- Supports only little-endian operation in this device
- Memory mapped register (MMR) bit fields are fixed position in 32-bit MMR

10.1.3 EDMA Controllers Configuration

Table 10-1 summarizes the configuration for each of the EDMA channel controllers present on the device.

Table 10-1. EDMA Channel Controllers Configuration

Parameter	EDMACC_0 Configuration	EDMACC_1 Configuration
Number of DMA channels (NUM_DMACH)	64	64
Number of QDMA channels (NUM_QDMACH)	8	8
Number of interrupt channels (NUM_INTCH)	64	64
Number of PaRAM set entries (NUM_PARAMENTRY)	512	512
Number of event queues (NUM_EVQUE)	2	2
Number of transfer controllers (NUM_TC)	2	2
Memory protection existence (MPEXIST)	Yes	Yes
Number of memory protection and shadow regions (NUM_REGIONS)	8	8
Channel mapping existence (CHMAPEXIST)	Yes	Yes

Table 10-2 summarizes the configuration of each of the EDMA transfer controllers present on the device.

Table 10-2. EDMA Transfer Controllers Configuration

Parameter	EDMACC_0_TC_0 Configuration	EDMACC_0_TC_1 Configuration	EDMACC_1_TC_0 Configuration	EDMACC_1_TC_1 Configuration
Data FIFO size (FIFOSIZE)	1024 bytes	1024 bytes	1024 bytes	1024 bytes
Bus width (BUSBYTE)	16 bytes	16 bytes	16 bytes	16 bytes
Number of destination FIFO register sets (DSTREGDEPTH)	4 entries	4 entries	4 entries	4 entries
Default burst size (DBS)	64 bytes	64 bytes	64 bytes	64 bytes

See Section 10.3 for functional details related to the EDMACC and EDMATC configuration parameters.

10.1.4 Terminology Used in This Chapter

Table 10-3 presents a brief explanation of some terms that are used in this chapter.

Table 10-3. EDMA Controller – Terms and Definitions

Term	Meaning
A-synchronized transfer	A transfer type where one dimension is serviced per synchronization event.
AB-synchronized transfer	A transfer type where two dimensions are serviced per synchronization event.
Chaining	A trigger mechanism in which a transfer can be initiated at the completion of another transfer or sub-transfer.
DMA channel	A channel that can be triggered by external, manual, and chained events. All DMA channels exist in the EDMACC.
DSP(s)	Digital signal processor(s).
Dummy set or dummy PaRAM set	A PaRAM set for which at least one of the count fields is equal to 0 and at least one of the count fields is nonzero. All of the count fields are cleared in a null PaRAM set.
Dummy transfer	A dummy set results in the EDMACC performing a dummy transfer. This is not an error condition. A null set results in an error condition.
EDMA channel controller(s) (EDMACC)	The EDMACC is the portion of the EDMA that is user-programmable. The EDMACC contains the parameter RAM (PaRAM), event processing logic, DMA/QDMA channels, and event queues. The EDMACC service events (external, manual, chained, and QDMA) and is responsible for submitting transfer requests to the transfer controllers (EDMATC) that perform the actual transfer.
EDMA programmer	Any entity on the chip that has read/write access to the EDMA registers and can program an EDMA transfer.
EDMA transfer controller(s) (EDMATC)	Transfer controllers are the transfer engines for the EDMA controller. They perform the read/writes, as dictated by the EDMACC's transfer requests.
Enhanced direct memory access (EDMA) controller	EDMA consists of the EDMA channel controller(s) (EDMACC) and the EDMA transfer controller(s) (EDMATC), referred to as EDMA in this document.
ITCCHEN	Intermediate transfer completion chaining enable.
ITCINTEN	Intermediate transfer completion interrupt enable.
Link parameter set	A PaRAM set that is used for linking.
Linking	The mechanism of reloading a PaRAM set with new transfer characteristics on completion of the current transfer.
Memory-mapped slaves	All on-chip memories, off-chip memories, and slave peripherals. These typically rely on the EDMA (or other master peripheral) to perform transfers to and from them.
Master peripherals	All peripherals that are capable of initiating read and write transfers to the system that may not solely rely on the EDMA for their data transfers.
Null set or null PaRAM	A PaRAM set that has all count fields cleared (except for the link field). A dummy PaRAM set has at set least one of the count fields nonzero.
Null transfer	A trigger event for a null PaRAM set results in the EDMACC performing a null transfer. This is an error condition. A dummy transfer is not an error condition.
Parameter RAM (PaRAM)	Programmable RAM that stores PaRAM sets that DMA channels, QDMA channels, and linking uses.
Parameter RAM (PaRAM) set	The PaRAM set is a 32-byte EDMA channel transfer definition. Each parameter set consists of eight words (that are four bytes each) that store the context for a DMA/QDMA/link transfer. A PaRAM set includes source address, destination address, counts, indexes, and options.
Parameter RAM (PaRAM) set entry	A PaRAM set entry occurs when one of the eight four-byte components of the parameter set.
QDMA channel	A channel that can be triggered when writing to the trigger word (TRWORD) of a PaRAM set. All QDMA channels exist in the EDMACC.
Slave end points	Slave end points are all on-chip memories, off-chip memories, and slave peripherals. Slave end points may rely on the EDMA to perform transfers to and from them.
SYNCDIM	Transfer synchronization dimension.
TCCHEN	Transfer complete chaining enable.
TCINTEN	Transfer complete interrupt enable.
Transfer request (TR)	A command for data movement that is issued from the EDMACC to the EDMATC. A TR includes source and destination addresses, counts, indexes, and options.
Trigger event	A trigger event is an action that causes the EDMACC to service the channel and to submit a transfer request to the EDMATC. Trigger events for the DMA channels include events that are triggered manually, externally, and by chain. Trigger events for QDMA channels include events that are triggered automatically and by link.

Table 10-3. EDMA Controller – Terms and Definitions (continued)

Term	Meaning
Trigger word	For QDMA channels, the trigger word specifies the PaRAM set entry that results in a QDMA trigger event when it is written. The trigger word is programmed via the QDMA channel map register (QCHMAP) and can point to any of the PaRAM set entries.
TR synchronization (sync) event	See Trigger event.

10.2 EDMA Integration

This section describes the integration of the module in the device, including information about clocks, resets, and hardware requests.

Figure 10-2 and Figure 10-2 show the integration of the EDMA controllers on the device.

Figure 10-2. EDMA_0 Controller Integration

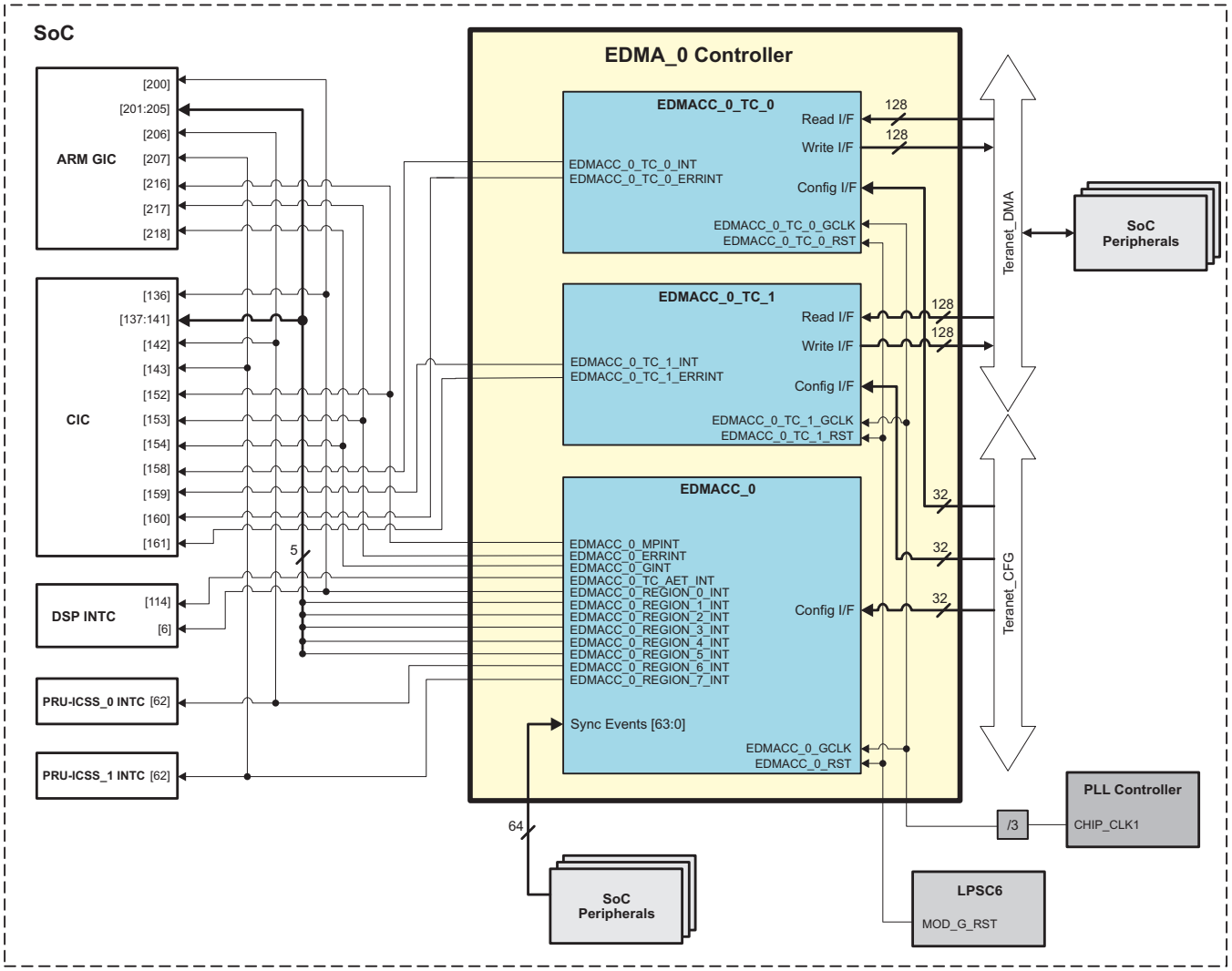


Figure 10-3. EDMA_1 Controller Integration

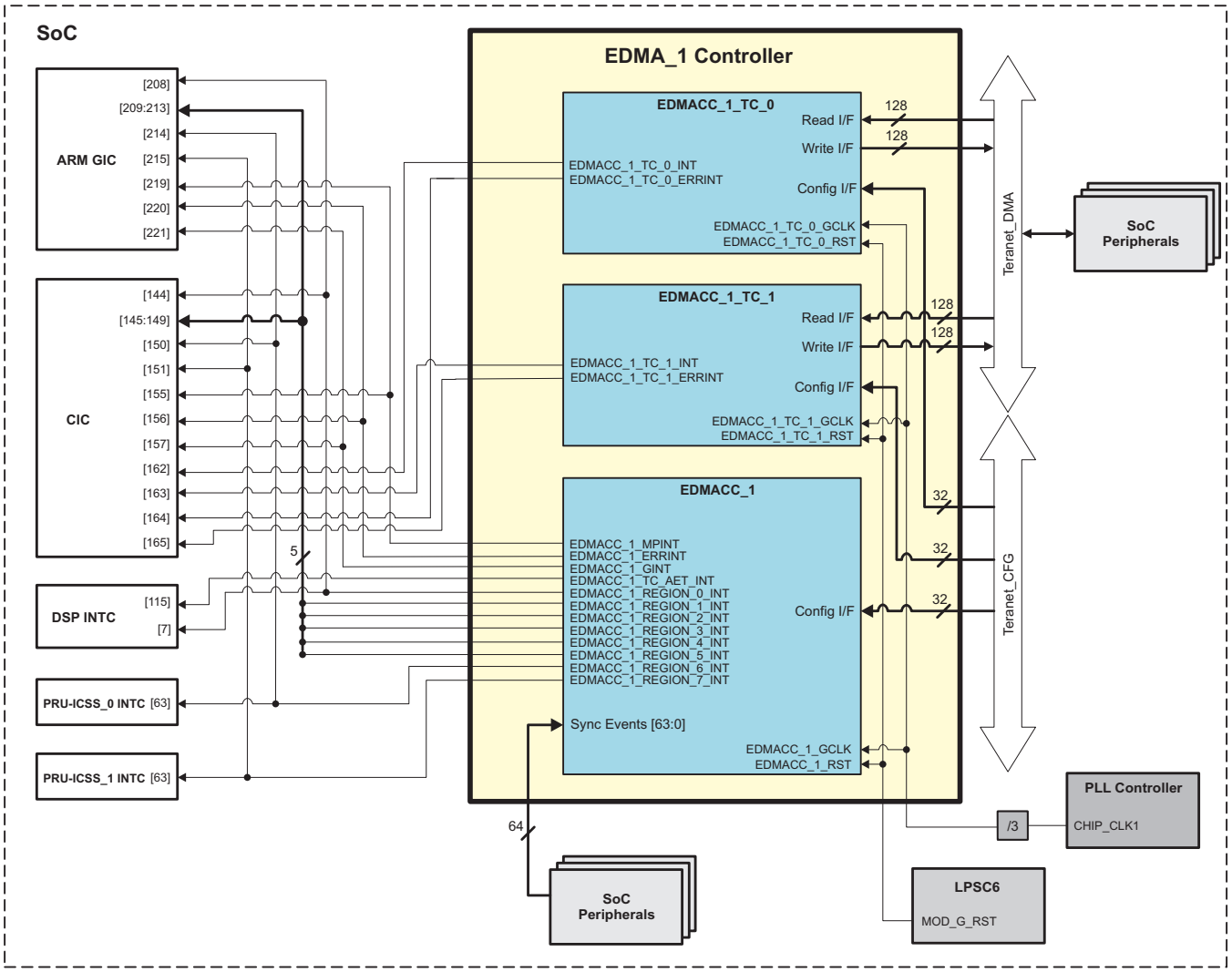


Table 10-4 through Table 10-6 summarize the integration of the EDMA controllers.

Table 10-4. EDMA Integration Attributes

Module Instance		Attributes		
		Power Domain	Module Domain	Interconnect
EDMA_0	EDMACC_0 EDMACC_0_TC_0 EDMACC_0_TC_1	PD5	LPSC6	TeraNet_DMA TeraNet_CFG
EDMA_1	EDMACC_1 EDMACC_1_TC_0 EDMACC_1_TC_1	PD5	LPSC6	TeraNet_DMA TeraNet_CFG

Table 10-5. EDMA Clocks and Resets

Clocks				
Module Instance	**Module Clock Name**	**Source Clock Name**	**Source**	**Description**
EDMACC_0	EDMACC_0_GCLK	CHIP_CLK1 / 3	PLL Controller	EDMA interface/functional clock. This clock is used for all interface and functional operations.
EDMACC_1	EDMACC_1_GCLK			
EDMACC_0_TC_0	EDMACC_0_TC_0_GCLK			
EDMACC_0_TC_1	EDMACC_0_TC_1_GCLK			
EDMACC_1_TC_0	EDMACC_1_TC_0_GCLK			
EDMACC_1_TC_1	EDMACC_1_TC_1_GCLK			
Resets				
Module Instance	**Module Reset Name**	**Source Reset Name**	**Source**	**Description**
EDMACC_0	EDMACC_0_RST	MOD_G_RST	LPSC6	EDMA hardware reset. See Section 10.3.17.
EDMACC_1	EDMACC_1_RST			
EDMACC_0_TC_0	EDMACC_0_TC_0_RST			
EDMACC_0_TC_1	EDMACC_0_TC_1_RST			
EDMACC_1_TC_0	EDMACC_1_TC_0_RST			
EDMACC_1_TC_1	EDMACC_1_TC_1_RST			

Table 10-6. EDMA Hardware Requests

Interrupt Requests								
Module Instance	**Event Name**	**Mapped To Input Event [Number]**						**Description**
		ARM GIC	**CIC**	**PRU-ICSS_0 INTC**	**PRU-ICSS_1 INTC**	**DSP INTC**	**PMMC INTC**	
EDMA_0	EDMACC_0_REGION_0_INT	[200]	[136]	–	–	[6]	–	EDMACC_0 region 0 DMA completion interrupt
	EDMACC_0_REGION_1_INT	[201]	[137]	–	–	–	–	EDMACC_0 region 1 DMA completion interrupt
	EDMACC_0_REGION_2_INT	[202]	[138]	–	–	–	–	EDMACC_0 region 2 DMA completion interrupt
	EDMACC_0_REGION_3_INT	[203]	[139]	–	–	–	–	EDMACC_0 region 3 DMA completion interrupt
	EDMACC_0_REGION_4_INT	[204]	[140]	–	–	–	–	EDMACC_0 region 4 DMA completion interrupt
	EDMACC_0_REGION_5_INT	[205]	[141]	–	–	–	–	EDMACC_0 region 5 DMA completion interrupt
	EDMACC_0_REGION_6_INT	[206]	[142]	[62]	–	–	–	EDMACC_0 region 6 DMA completion interrupt
	EDMACC_0_REGION_7_INT	[207]	[143]	–	[62]	–	–	EDMACC_0 region 7 DMA completion interrupt
	EDMACC_0_MPINT	[216]	[152]	–	–	–	–	EDMACC_0 memory protection interrupt
	EDMACC_0_ERRINT	[217]	[153]	–	–	–	–	EDMACC_0 error interrupt
	EDMACC_0_GINT	[218]	[154]	–	–	–	–	EDMACC_0 global completion interrupt
	EDMACC_0_TC_AET_INT	–	–	–	–	[114]	–	EDMACC_0 advanced event trigger (AET) interrupt
	EDMACC_0_TC_0_INT	–	[158]	–	–	–	–	EDMACC_0_TC_0 completion interrupt
	EDMACC_0_TC_1_INT	–	[159]	–	–	–	–	EDMACC_0_TC_1 completion interrupt
	EDMACC_0_TC_0_ERRINT	–	[160]	–	–	–	–	EDMACC_0_TC_0 error interrupt

Table 10-6. EDMA Hardware Requests (continued)

	EDMACC_0_TC_1_ERRINT	–	[161]	–	–	–	–	EDMACC_0_TC_1 error interrupt
EDMA_1	EDMACC_1_REGION_0_INT	[208]	[144]	–	–	[7]	–	EDMACC_1 region 0 DMA completion interrupt
	EDMACC_1_REGION_1_INT	[209]	[145]	–	–	–	–	EDMACC_1 region 1 DMA completion interrupt
	EDMACC_1_REGION_2_INT	[210]	[146]	–	–	–	–	EDMACC_1 region 2 DMA completion interrupt
	EDMACC_1_REGION_3_INT	[211]	[147]	–	–	–	–	EDMACC_1 region 3 DMA completion interrupt
	EDMACC_1_REGION_4_INT	[212]	[148]	–	–	–	–	EDMACC_1 region 4 DMA completion interrupt
	EDMACC_1_REGION_5_INT	[213]	[149]	–	–	–	–	EDMACC_1 region 5 DMA completion interrupt
	EDMACC_1_REGION_6_INT	[214]	[150]	[63]	–	–	–	EDMACC_1 region 6 DMA completion interrupt
	EDMACC_1_REGION_7_INT	[215]	[151]	–	[63]	–	–	EDMACC_1 region 7 DMA completion interrupt
	EDMACC_1_MPINT	[219]	[155]	–	–	–	–	EDMACC_1 memory protection interrupt
	EDMACC_1_ERRINT	[220]	[156]	–	–	–	–	EDMACC_1 error interrupt
	EDMACC_1_GINT	[221]	[157]	–	–	–	–	EDMACC_1 global completion interrupt
	EDMACC_1_TC_AET_INT	–	–	–	–	[115]	–	EDMACC_1 advanced event trigger (AET) interrupt
	EDMACC_1_TC_0_INT	–	[162]	–	–	–	–	EDMACC_1_TC_0 completion interrupt
	EDMACC_1_TC_1_INT	–	[163]	–	–	–	–	EDMACC_1_TC_1 completion interrupt
	EDMACC_1_TC_0_ERRINT	–	[164]	–	–	–	–	EDMACC_1_TC_0 error interrupt
	EDMACC_1_TC_1_ERRINT	–	[165]	–	–	–	–	EDMACC_1_TC_1 error interrupt

10.2.1 EDMA Synchronization Events

The EDMA supports up to 64 DMA channels (per EDMACC instance) that can be used to service system peripherals and to move data between system memories. DMA channels can be triggered by synchronization events generated by system peripherals. Table 10-7 and Table 10-8 list the source of the synchronization event associated with each of the EDMACC DMA channels.

For more information about how EDMA events are enabled, captured, processed, prioritized, linked, chained, and cleared, see Section 10.3.

10.2.1.1 EDMACC_0 Synchronization Events

Table 10-7 lists the EDMACC_0 synchronization events.

Table 10-7. EDMACC_0 Synchronization Events

Event No.	Event Name	Description
0	TIMER_2_INTL	TIMER_2 low interrupt
1	TIMER_2_INTH	TIMER_2 high interrupt
2	TIMER_3_INTL	TIMER_3 low interrupt
3	TIMER_3_INTH	TIMER_3 high interrupt
4	SPI_0_XEVT	SPI_0 level 0 interrupt
5	SPI_0_REVT	SPI_0 level 1 interrupt
6	SPI_1_XEVT	SPI_1 level 0 interrupt
7	SPI_1_REVT	SPI_1 level 1 interrupt
8	I2C_0_XEVT	I2C_0 transmit event
9	I2C_0_REVT	I2C_0 receive event
10	QSPI_INT	QSPI interrupt
11	PMMCTBR_DMAINT	PMMC trace buffer DMA event
12	ARM_TBR_DMA	ARMSS trace buffer DMA event
14	TETB_HFULLINT0	TETB half-full flag
13	DBGTBR_DMAINT	System trace buffer DMA event
15	TETB_FULLINT0	TETB full flag
16	UART_0_UTXEVT	UART_0 transmit event
17	UART_0_URXEVT	UART_0 receive event
18	UART_1_UTXEVT	UART_1 transmit event
19	UART_1_URXEVT	UART_1 receive event
20	UART_2_UTXEVT	UART_2 transmit event
21	UART_2_URXEVT	UART_2 receive event
22	DSS_DISPCDMA_REQ	DSS line-trigger DMA event
23	DSS_RFBIDMA_REQ	DSS RFBI FIFO DMA event
24	McASP_0_XEVT	McASP_0 transmit event
25	McASP_0_REVT	McASP_0 receive event
26	McBSP_XEVT	McBSP transmit event
27	McBSP_REVT	McBSP receive event
28	CIC_0_OUT32	CIC output 32 event
29	CIC_0_OUT33	CIC output 33 event
30	CIC_0_OUT34	CIC output 34 event
31	CIC_0_OUT35	CIC output 35 event
32	ASRC_GROUP_IN_0_EVT	ASRC Group 0 input event
33	ASRC_GROUP_IN_1_EVT	ASRC Group 1 input event
34	ASRC_GROUP_OUT_0_EVT	ASRC Group 0 output event
35	ASRC_GROUP_OUT_1_EVT	ASRC Group 1 output event
36	CIC_0_OUT28	CIC output 28 event

Table 10-7. EDMACC_0 Synchronization Events (continued)

Event No.	Event Name	Description
37	CIC_0_OUT29	CIC output 29 event
38	CIC_0_OUT30	CIC output 30 event
39	CIC_0_OUT31	CIC output 31 event
40	NSS_QPEND48	Transmit queue 48 pending event
41	NSS_QPEND49	Transmit queue 49 pending event
42	NSS_QPEND50	Transmit queue 50 pending event
43	NSS_QPEND51	Transmit queue 51 pending event
44	NSS_QPEND64	Transmit queue 64 pending event
45	NSS_QPEND65	Transmit queue 65 pending event
46	NSS_QPEND66	Transmit queue 66 pending event
47	NSS_QPEND67	Transmit queue 67 pending event
48	GPIOMUX_INT48	GPIOMUX48 output event
49	GPIOMUX_INT49	GPIOMUX49 output event
50	GPIOMUX_INT50	GPIOMUX50 output event
51	GPIOMUX_INT51	GPIOMUX51 output event
52	GPIOMUX_INT52	GPIOMUX52 output event
53	GPIOMUX_INT53	GPIOMUX53 output event
54	GPIOMUX_INT54	GPIOMUX54 output event
55	GPIOMUX_INT55	GPIOMUX55 output event
56	GPIOMUX_INT32	GPIOMUX32 output event
57	GPIOMUX_INT33	GPIOMUX33 output event
58	GPIOMUX_INT34	GPIOMUX34 output event
59	GPIOMUX_INT35	GPIOMUX35 output event
60	GPIOMUX_INT36	GPIOMUX36 output event
61	GPIOMUX_INT37	GPIOMUX37 output event
62	GPIOMUX_INT38	GPIOMUX38 output event
63	GPIOMUX_INT39	GPIOMUX39 output event

10.2.1.2 EDMACC_1 Synchronization Events

Table 10-8 lists the EDMACC_1 synchronization events.

Table 10-8. EDMACC_1 Synchronization Events

Event No.	Event Name	Description
0	TIMER_4_INTL	TIMER_4 low interrupt
1	TIMER_4_INTH	TIMER_4 high interrupt
2	TIMER_1_INTL	TIMER_1 low interrupt
3	TIMER_1_INTH	TIMER_1 high interrupt
4	SPI_2_XEVT	SPI_2 level 0 interrupt
5	SPI_2_REVT	SPI_2 level 1 interrupt
6	SPI_3_XEVT	SPI_3 level 0 interrupt
7	SPI_3_REVT	SPI_3 level 1 interrupt
8	I2C_1_XEVT	I2C_1 transmit event
9	I2C_1_REVT	I2C_1 receive event
10	I2C_2_XEVT	I2C_2 transmit event
11	I2C_2_REVT	I2C_2 receive event
12	Reserved	Reserved

Table 10-8. EDMACC_1 Synchronization Events (continued)

Event No.	Event Name	Description
13	Reserved	Reserved
14	Reserved	Reserved
15	Reserved	Reserved
16	DCAN_0_INTERFACE1_EVT	DCAN_0 interface 1 event
17	DCAN_0_INTERFACE2_EVT	DCAN_0 interface 2 event
18	DCAN_0_INTERFACE3_EVT	DCAN_0 interface 3 event
19	DCAN_1_INTERFACE1_EVT	DCAN_1 interface 1 event
20	DCAN_1_INTERFACE2_EVT	DCAN_1 interface 2 event
21	DCAN_1_INTERFACE3_EVT	DCAN_1 interface 3 event
22	CIC_0_OUT24	CIC output 24 event
23	CIC_0_OUT25	CIC output 25 event
24	MMC_SD_0_TX_EVT	MMC/SD0 transmit event
25	MMC_SD_0_RX_EVT	MMC/SD0 receive event
26	MMC_SD_1_TX_EVT	MMC/SD1 transmit event
27	MMC_SD_1_RX_EVT	MMC/SD1 receive event
28	EQEP_0_INT	eQEP_0 interrupt
29	EQEP_1_INT	eQEP_1 interrupt
30	EQEP_2_INT	eQEP_2 interrupt
31	GPMC_DMA_REQ	GPMC DMA event
32	EPWM_0_INT	ePWM_0 event trigger (ET) interrupt
33	EPWM_1_INT	ePWM_1 event trigger (ET) interrupt
34	EPWM_2_INT	ePWM_2 event trigger (ET) interrupt
35	EPWM_3_INT	ePWM_3 event trigger (ET) interrupt
36	EPWM_4_INT	ePWM_4 event trigger (ET) interrupt
37	EPWM_5_INT	ePWM_5 event trigger (ET) interrupt
38	EPWM_0_TRIP_ZONE	ePWM_0 tripzone (TZ) interrupt
39	EPWM_1_TRIP_ZONE	ePWM_1 tripzone (TZ) interrupt
40	EPWM_2_TRIP_ZONE	ePWM_2 tripzone (TZ) interrupt
41	EPWM_3_TRIP_ZONE	ePWM_3 tripzone (TZ) interrupt
42	EPWM_4_TRIP_ZONE	ePWM_4 tripzone (TZ) interrupt
43	EPWM_5_TRIP_ZONE	ePWM_5 tripzone (TZ) interrupt
44	ECAP_0_INT	eQEP_0 interrupt
45	ECAP_1_INT	eQEP_1 interrupt
46	CIC_0_OUT26	CIC output 26 event
47	CIC_0_OUT27	CIC output 27 event
48	McASP_1_XEVT	McASP_1 transmit nterrupt
49	McASP_1_REVT	McASP_1 receive interrupt
50	McASP_2_XEVT	McASP_2 transmit interrupt
51	McASP_2_REVT	McASP_2 receive interrupt
52	ASRC_GROUP_IN_2_EVT	ASRC Group 2 input event
53	ASRC_GROUP_IN_3_EVT	ASRC Group 3 input event
54	ASRC_GROUP_OUT_2_EVT	ASRC Group 2 output event
55	ASRC_GROUP_OUT_3_EVT	ASRC Group 3 output event
56	GPIOMUX_INT40	GPIOMUX40 output event
57	GPIOMUX_INT41	GPIOMUX41 output event
58	GPIOMUX_INT42	GPIOMUX42 output event
59	GPIOMUX_INT43	GPIOMUX43 output event

Table 10-8. EDMACC_1 Synchronization Events (continued)

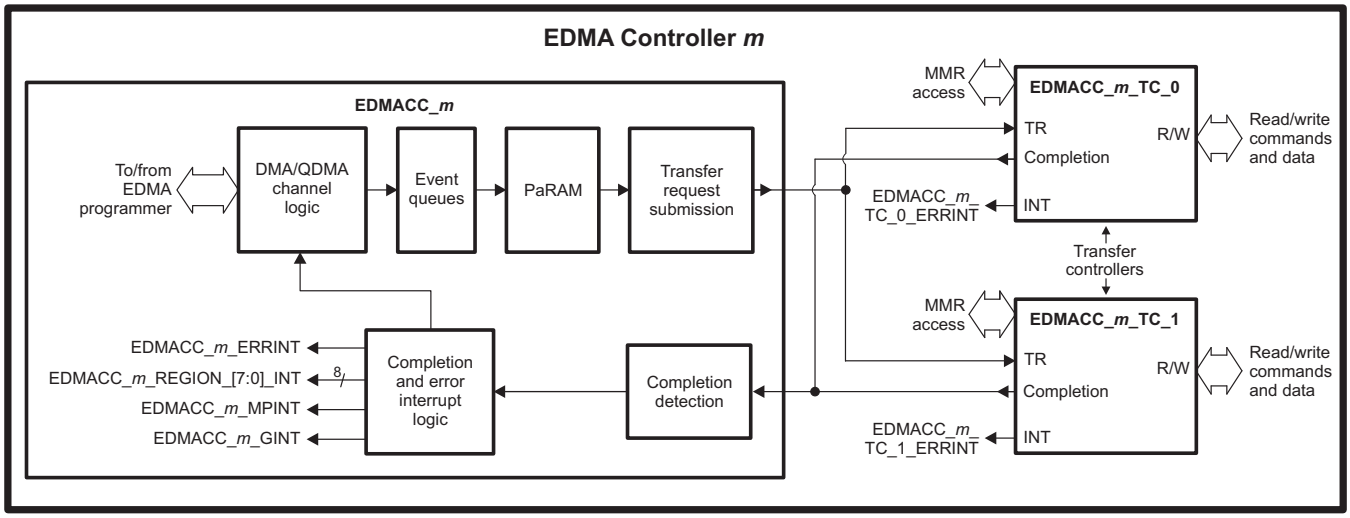
Event No.	Event Name	Description
60	GPIOMUX_INT44	GPIOMUX44 output event
61	GPIOMUX_INT45	GPIOMUX45 output event
62	GPIOMUX_INT46	GPIOMUX46 output event
63	GPIOMUX_INT47	GPIOMUX47 output event

10.3 EDMA Functional Description

10.3.1 EDMA Controller Block Diagram

Figure 10-4 shows the EDMA controller block diagram.

Figure 10-4. EDMA Controller Block Diagram



NOTE: 'm' indicates the EDMACC instance number.

10.3.2 EDMA Channel Controller (EDMACC)

The EDMACC is responsible for scheduling, arbitrating, and issuing user-programmed transfers to the two EDMATCs associated with it.

Figure 10-5 shows a functional block diagram of the EDMACC. The main blocks of the EDMACC are:

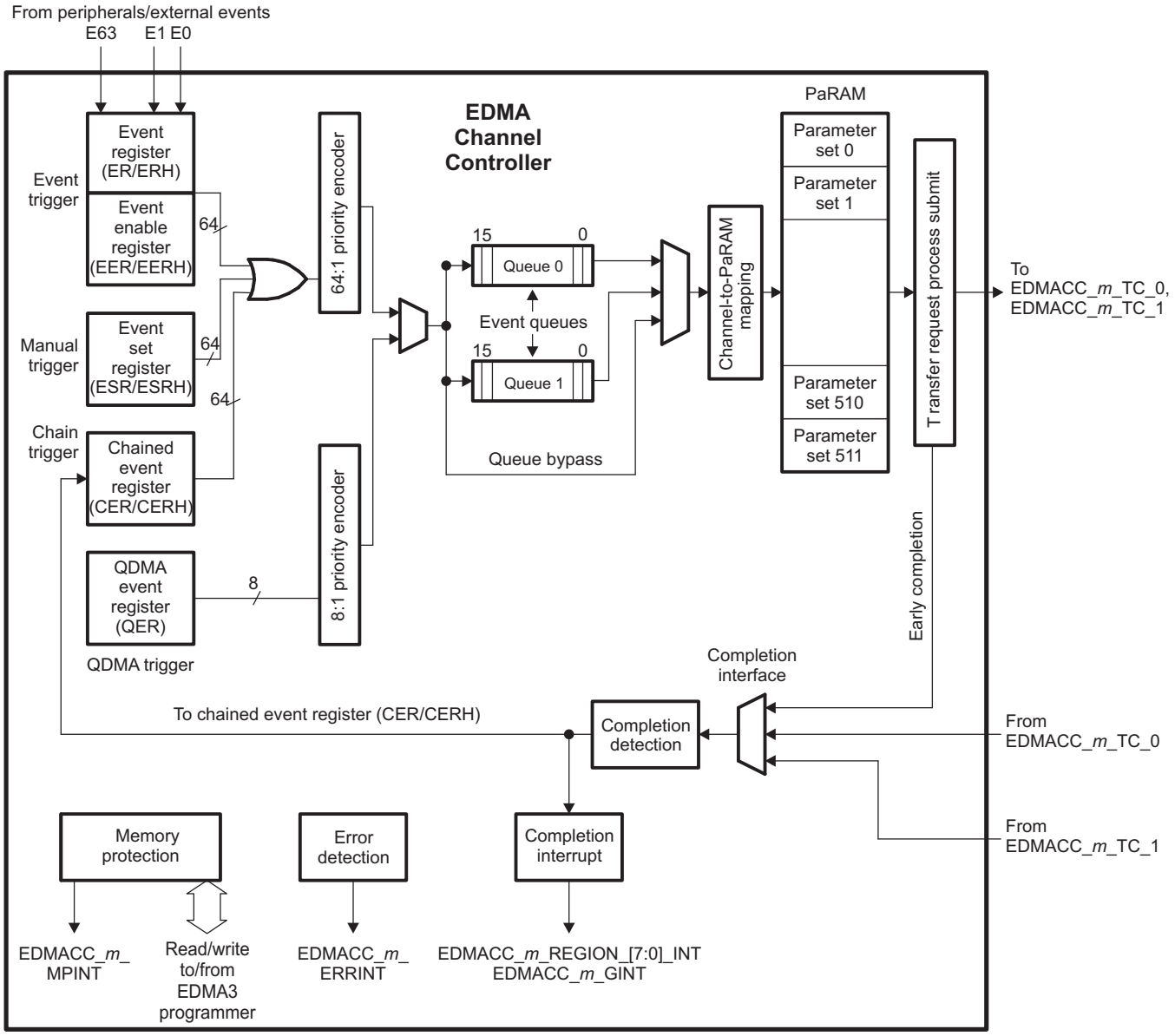
- **DMA/QDMA Channel Logic**: This block consists of logic that captures external system or peripheral events that can be used to initiate event triggered transfers. It also includes:
  - Registers that allow configuring the DMA/QDMA channels (queue mapping, PaRAM entry mapping).
  - All the registers for different trigger type (manual, external events, chained and auto-triggered) for enabling/disabling events, and monitor event status.
- **Parameter RAM (PaRAM)**: The PaRAM maintains parameter set entries for channel and reload parameter sets. The PaRAM must be written with the transfer context for the desired channels and link parameter sets.
- **Event Queues**: Event queues form the interface between the event detection logic and the transfer request submission logic.
- **Transfer Request Submission Logic**: This logic processes PaRAM sets based on a trigger event submitted to the event queue and submits a transfer request (TR) to the transfer controller associated with the event queue.
- **EDMA Event and Interrupt Processing Registers**: Allows mapping of events to parameter sets, enable/disable events, enable/disable interrupt conditions, and clearing interrupts.
- **Completion Detection**: This block detects completion of transfers by the EDMATC and/or slave peripherals. Completion of transfers can optionally be used to chain trigger new transfers or to assert interrupts. The logic includes the interrupt processing registers for enabling/disabling interrupt (to be sent to the CPU), interrupt status/clearing registers.
- **Memory Protection Registers**: Memory protection registers define the accesses (privilege level and requestor(s)) that are allowed to access the DMA channel shadow region view(s) and regions of

PaRAM.

Additional functions include the following:

- **Region Registers**: Region registers allow DMA resources (DMA channels and interrupts) to be assigned to unique regions, which can be owned by unique EDMA programmers.
- **Debug Registers**: Debug registers allow debug visibility by providing registers to read the queue status, channel controller status, and missed event status.

Figure 10-5. EDMA Channel Controller Block Diagram



NOTE: 'm' indicates the EDMACC instance number.

The EDMACC includes two channel types: DMA channels and QDMA channels.

Each channel is associated with a given event queue/transfer controller and with a given PaRAM set. The main difference between a DMA channel and QDMA channel is how the transfers are triggered by the system. See Section 10.3.6.

A trigger event is necessary to initiate a transfer. For DMA channels, a trigger event may be due to an external event, manual write to the event set register, or chained event. QDMA channels are auto-triggered when a write is performed to the user-programmed trigger word. All such trigger events are logged into appropriate registers upon recognition. See Section 10.6.1.5 and Section 10.6.1.7.

Once a trigger event is recognized, the event type/channel is queued in the appropriate EDMACC event queue. The assignment of each DMA/QDMA channel to event queue is programmable. Each queue is 16 deep, so up to 16 events may be queued (on a single queue) in the EDMACC at an instant in time. Additional pending events mapped to a full queue are queued when event queue space becomes available. See Section 10.3.13.

If events on different channels are detected simultaneously, the events are queued based on a fixed priority arbitration scheme with the DMA channels being higher priority events than the QDMA channels. Among the two groups of channels, the lowest-numbered channel is the highest priority.

Each event in the event queue is processed in FIFO order. On reaching the head of the queue, the PaRAM associated with that channel is read to determine the transfer details. The TR submission logic evaluates the validity of the TR and is responsible for submitting a valid transfer request (TR) to the appropriate EDMATC (based on the event queue to the EDMATC association, Q0 goes to EDMATC0, and Q1 goes to EDMATC1). For more information, see Section 10.3.5.

The EDMATC receives the request and is responsible for data movement as specified in the transfer request packet (TRP), and other necessary tasks like buffering, ensuring transfers are carried out in an optimal fashion wherever possible. For more information on EDMATC, see Section 10.3.3.

The user might choose to receive an interrupt or chain to another channel on completion of the current transfer, in which case the EDMATC signals completion to the EDMACC completion detection logic when the transfer is done. The user can alternately choose to trigger completion when a TR leaves the EDMACC boundary, rather than wait for all of the data transfers to complete. Based on the setting of the EDMACC interrupt registers, the completion interrupt generation logic is responsible for generating EDMACC completion interrupts to the CPU. For more information, see Section 10.3.7.

Additionally, the EDMACC also has an error detection logic that causes an error interrupt generation on various error conditions (like missed events, exceeding event queue thresholds, etc.). For more information on error interrupts, see Section 10.3.11.4.

10.3.3 EDMA Transfer Controller (EDMATC)

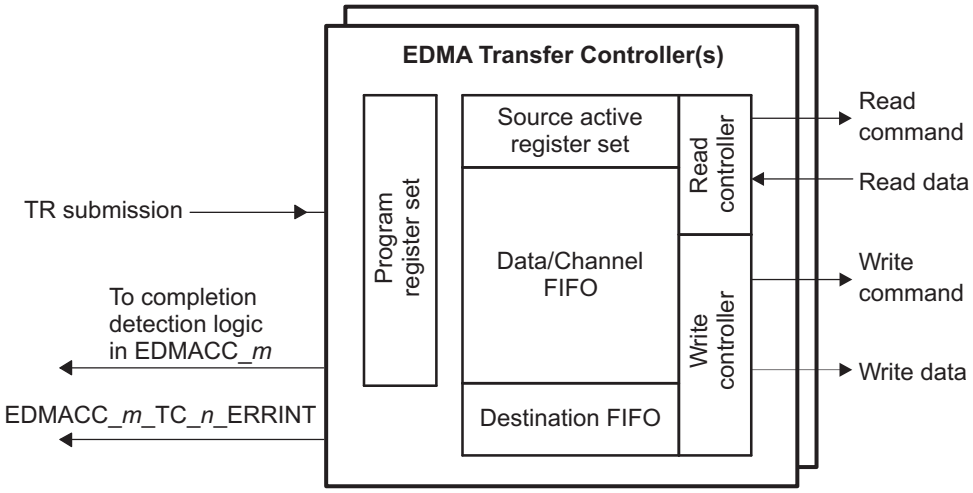
The EDMATC module is the EDMA transfer engine that generates transfers as programmed in dedicated working registers, using two dedicated master ports: a read-only port, and a write-only port.

Figure 10-6 shows a functional block diagram of the EDMA transfer controller (EDMATC). The main blocks of the EDMATC are:

- **DMA Program Register Set**: Stores the context for the DMA transfer that is loaded into the active register set when the current active register set completes. The CPU or EDMACC programs the program register set, not the active register set. For typical standalone operation, the CPU programs the program register while the EDMATC services the active register set. The program register set includes ownership control such that CPU software and the EDMA stay synchronized relative to one another.
- **DMA Source Active Register Set**: Stores the context (src/dst/cnt/etc) for the DMA Transfer Request (TR) in progress in the Read Controller. The active register set is split into independent Source and Destination, because the source interconnect controller and the distant interconnect controller operate independently of one another.
- **Destination FIFO Register Set**: Stores the context (src/dst/cnt/etc) for the DMA Transfer Request (TR) in progress, or pending, in the Write Controller. The pending register must allow the source controller to begin processing a new TR while the distant register set processes the previous TR.
- **Data/Channel FIFO**: Temporary holding buffer for in-flight data. The read return data of the source peripheral is stored in the Data FIFO, and then is written to the destination peripheral by the write command/data bus.
- **Read Controller/Read Interface**: The read interface issues optimally sized read commands to the source peripheral, based on a burst size of 64 bytes and available landing space in the channel FIFO.

- **Write Controller/Write Interface**: The write interface issues optimally sized write commands to the destination peripheral, based on a burst size of 64 bytes and available data in the channel FIFO.
- **Completion Interface**: Sends completion codes to the EDMACC when a transfer completes and generates interrupts and chained events in the EDMACC module (see Section 10.3.2 for more information on transfer completion reporting).
- **Configuration Port**: Slave interface that provides read/write access to program registers and read access to all memory-mapped EDMATC registers.

Figure 10-6. EDMA Transfer Controller Block Diagram



NOTE: 'm' indicates the EDMACC instance number, and 'n' indicates the EDMATC number.

When the EDMATC is idle and receives its first TR, the TR is received in the DMA program register set, where it transitions to the DMA source active set and the destination FIFO register set immediately. The source active register set tracks the commands for the source side of the transfers, and the destination FIFO register set tracks commands for the destination side of the transfer. The second TR (if pending from EDMACC) is loaded into the DMA program register set, ensuring it can start as soon as possible when the active transfer (the transfer in the source active set) is completed. As soon as the current active set is exhausted, the TR is loaded from the DMA program register set into the DMA source active register set as well as to the appropriate entry in the destination FIFO register set.

The read controller issues read commands governed by the rules of command fragmentation and optimization. These are issued only when the data FIFO has space available for the read data. The number of read commands issued depends on the TR transfer size. The TC write controller starts issuing write commands as soon as sufficient data is read in the data FIFO for the write controller to issue optimally sized write commands following the rules for command fragmentation and optimization. For details on command fragmentation and optimization, see Section 10.3.14.1.2.

The DSTREGDEPTH parameter (fixed in design for a given transfer controller) determines the number of entries in the destination FIFO register set. The number of entries determines the amount of TR pipelining possible for a given TC. The write controller can manage the write context for the number of entries in the destination FIFO register set. This allows the read controller to go ahead and issue read commands for the subsequent TRs while the destination FIFO register set manages the write commands and data for the previous TR.

In summary, since the DSTREGDEPTH is fixed to 4 in this device, the read controller is able to process up to 4 TRs ahead of the write controller. However, the overall TR pipelining is also subject to the amount of free space in the data FIFO.

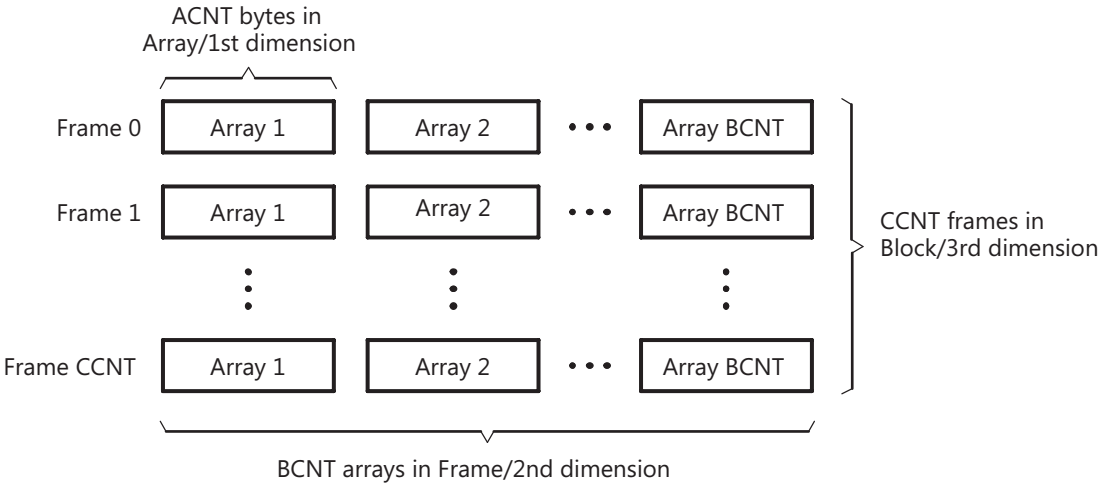
10.3.4 Types of EDMA Transfers

An EDMA transfer is always defined in terms of three dimensions. Figure 10-7 shows the three dimensions used by EDMA transfers. These three dimensions are defined as:

- 1st Dimension or Array (A): The 1st dimension in a transfer consists of ACNT contiguous bytes.
- 2nd Dimension or Frame (B): The 2nd dimension in a transfer consists of BCNT arrays of ACNT bytes. Each array transfer in the 2nd dimension is separated from each other by an index programmed using SRCBIDX or DSTBIDX.
- 3rd Dimension or Block (C): The 3rd dimension in a transfer consists of CCNT frames of BCNT arrays of ACNT bytes. Each transfer in the 3rd dimension is separated from the previous by an index programmed using SRCCIDX or DSTCIDX.

Note that the reference point for the index depends on the synchronization type. The amount of data transferred upon receipt of a trigger/synchronization event is controlled by the synchronization types (SYNCDIM bit in OPT). Of the three dimensions, only two synchronization types are supported: A-synchronized transfers and AB-synchronized transfers.

Figure 10-7. Definition of ACNT, BCNT, and CCNT



10.3.4.1 A-Synchronized Transfers

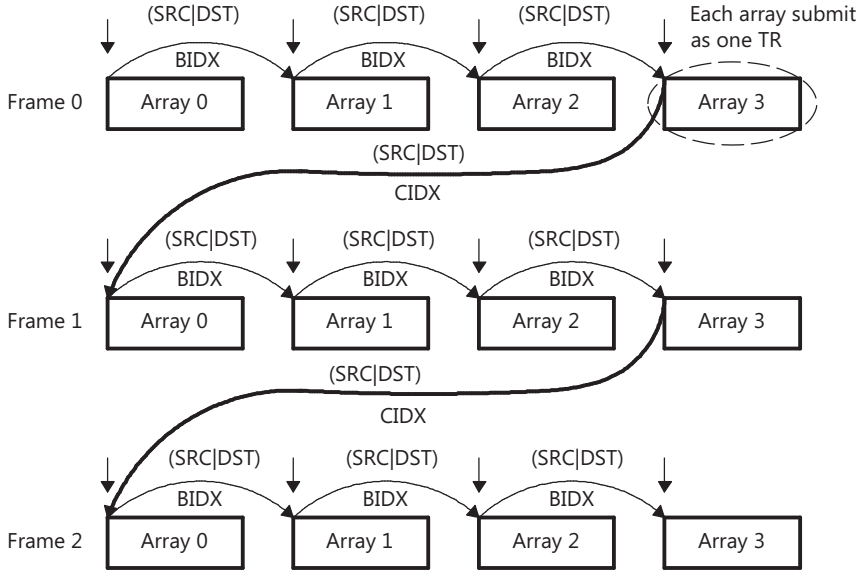
In an A-synchronized transfer, each EDMA sync event initiates the transfer of the 1st dimension of ACNT bytes, or one array of ACNT bytes. In other words, each event/TR packet conveys the transfer information for one array only. Thus, BCNT × CCNT events are needed to completely service a PaRAM set.

Arrays are always separated by SRCBIDX and DSTBIDX, as shown in Figure 10-8, where the start address of Array N is equal to the start address of Array N - 1 plus source (SRC) or destination (DST) BIDX.

Frames are always separated by SRCCIDX and DSTCIDX. For A-synchronized transfers, after the frame is exhausted, the address is updated by adding SRCCIDX/DSTCIDX to the beginning address of the last array in the frame. As in Figure 10-8, SRCCIDX/DSTCIDX is the difference between the start of Frame 0 Array 3 to the start of Frame 1 Array 0.

Figure 10-8 shows an A-synchronized transfer of 3 (CCNT) frames of 4 (BCNT) arrays of n (ACNT) bytes. In this example, a total of 12 sync events (BCNT × CCNT) exhaust a PaRAM set. See Section 10.3.5.6 for details on parameter set updates.

Figure 10-8. A-Synchronized Transfers (ACNT = n, BCNT = 4, CCNT = 3)



10.3.4.2 AB-Synchronized Transfers

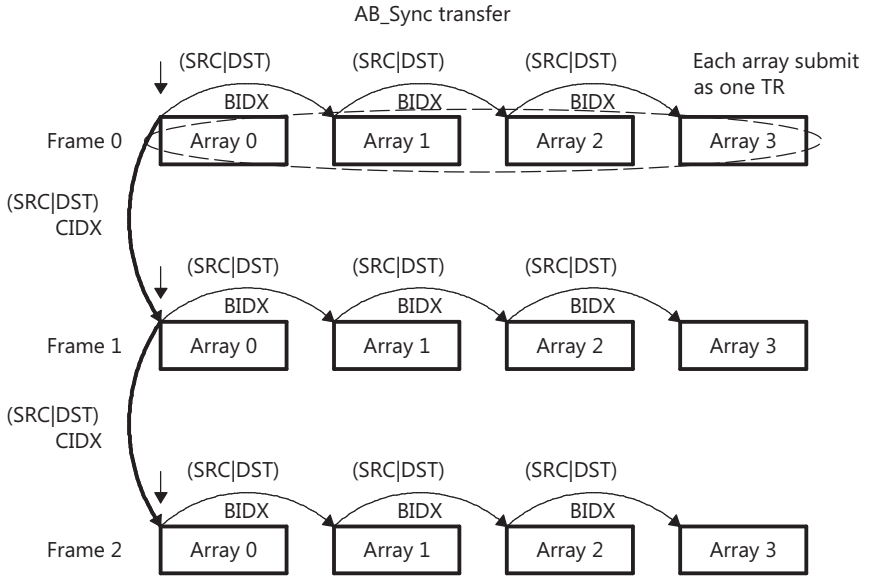
In a AB-synchronized transfer, each EDMA sync event initiates the transfer of 2 dimensions or one frame. In other words, each event/TR packet conveys information for one entire frame of BCNT arrays of ACNT bytes. Thus, CCNT events are needed to completely service a PaRAM set.

Arrays are always separated by SRCBIDX and DSTBIDX as shown in Figure 10-9. Frames are always separated by SRCCIDX and DSTCIDX.

Note that for AB-synchronized transfers, after a TR for the frame is submitted, the address update is to add SRCCIDX/DSTCIDX to the beginning address of the beginning array in the frame. This is different from A-synchronized transfers where the address is updated by adding SRCCIDX/DSTCIDX to the start address of the last array in the frame. See Section 10.3.5.6 for details on parameter set updates.

Figure 10-9 shows an AB-synchronized transfer of 3 (CCNT) frames of 4 (BCNT) arrays of n (ACNT) bytes. In this example, a total of 3 sync events (CCNT) exhaust a PaRAM set; that is, a total of 3 transfers of 4 arrays each completes the transfer.

Figure 10-9. AB-Synchronized Transfers (ACNT = n, BCNT = 4, CCNT = 3)



NOTE: ABC-synchronized transfers are not directly supported. It can be logically achieved by chaining between multiple AB-synchronized transfers.

10.3.5 Parameter RAM (PaRAM)

The EDMA controller is a RAM-based architecture. The transfer context (source/destination addresses, count, indexes, etc.) for DMA or QDMA channels is programmed in a parameter RAM (PaRAM) table within the EDMACC. The PaRAM table is segmented into multiple PaRAM sets. Each PaRAM set includes eight 4-byte PaRAM set entries (32 bytes total per PaRAM set), which includes typical DMA transfer parameters such as source address, destination address, transfer counts, indexes, options, etc.

The PaRAM structure supports flexible ping-pong, circular buffering, channel chaining, and autoreloading (linking). The first 64 PaRAM sets are directly mapped to the DMA channels. The remaining PaRAM sets can be used for link entries or associated with QDMA channels. Additionally, if the DMA channels are not used, the PaRAM sets associated with the unused DMA channels can also be used for link entries or QDMA channels.

The contents of the PaRAM include the following:

- 512 PaRAM sets
- Any PaRAM entry can be used for DMA, QDMA, or link sets
- By default, all channels are mapped to PaRAM set 0. They should be re-mapped before use. See (DCHMAP registers) and (QCHMAP registers) for more information.

Table 10-9. EDMA Parameter RAM Contents

PaRAM Set Number[(1)]	Address Offset
0	4000h to 401Fh
1	4020h to 403Fh
2	4040h to 405Fh
3	4060h to 407Fh
4	4080h to 409Fh
5	40A0h to 40BFh
6	40C0h to 40DFh

(1) A PaRAM set can be configured for use with either DMA channel, QDMA channel, or as a reload link set.

Table 10-9. EDMA Parameter RAM Contents (continued)

PaRAM Set Number[1]	Address Offset
7	40E0h to 40FFh
8	4100h to 411Fh
9	4120h to 413Fh
...	...
64	4800h to 481Fh
65	4820h to 483Fh
...	...
126	4FC0h to 4FDFh
127	4FE0h to 4FFFh
...	...
254	5FC0h to 5FDFh
255	5FE0h to 5FFFh
...	...
510	7FC0h to 7FDFh
511	7FE0h to 7FFFh

10.3.5.1 PaRAM Set

Each parameter set of PaRAM is organized into eight 32-bit words or 32 bytes, as shown in Figure 10-10 and described in Table 10-10. Each PaRAM set consists of 16-bit and 32-bit parameters.

Figure 10-10. PaRAM Set

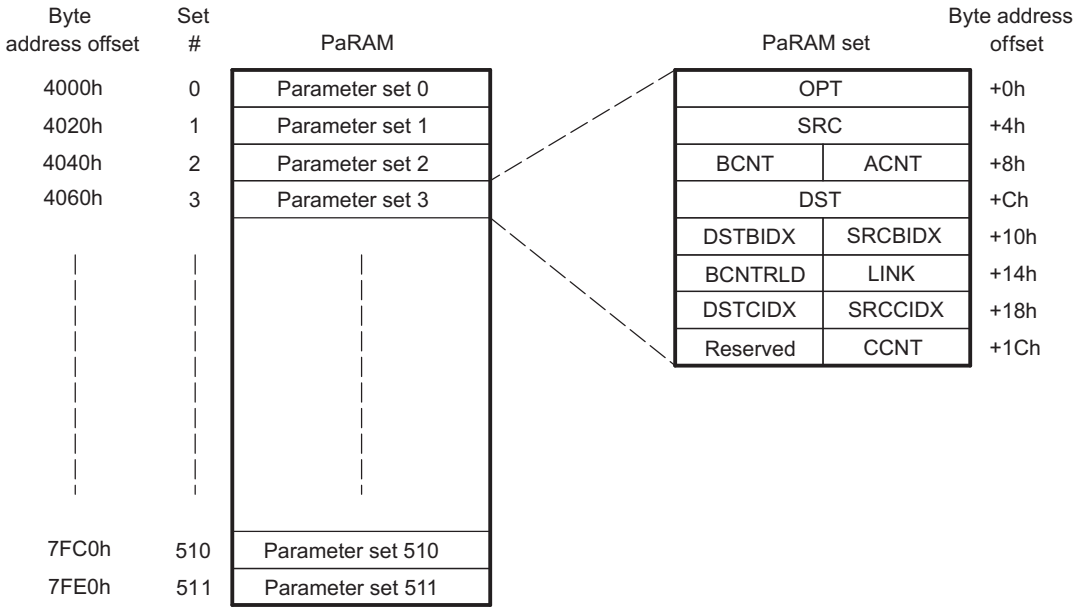


Table 10-10. EDMA Channel Parameter Description

Offset Address (bytes)	Acronym	Parameter	Description
0h	OPT	Channel Options	Transfer configuration options
4h	SRC	Channel Source Address	The byte address from which data is transferred

Table 10-10. EDMA Channel Parameter Description (continued)

Offset Address (bytes)	Acronym	Parameter	Description
8h [(1)]	ACNT	Count for 1st Dimension	Unsigned value specifying the number of contiguous bytes within an array (first dimension of the transfer). Valid values range from 1 to 65 535.
	BCNT	Count for 2nd Dimension	Unsigned value specifying the number of arrays in a frame, where an array is ACNT bytes. Valid values range from 1 to 65 535.
Ch	DST	Channel Destination Address	The byte address to which data is transferred
10h[(1)]	SRCBIDX	Source BCNT Index	Signed value specifying the byte address offset between source arrays within a frame (2nd dimension). Valid values range from -32 768 and 32 767.
	DSTBIDX	Destination BCNT Index	Signed value specifying the byte address offset between destination arrays within a frame (2nd dimension). Valid values range from -32 768 and 32 767.
14h[(1)]	LINK	Link Address	The PaRAM address containing the PaRAM set to be linked (copied from) when the current PaRAM set is exhausted. A value of FFFFh specifies a null link.
	BCNTRLD	BCNT Reload	The count value used to reload BCNT when BCNT decrements to 0 (TR is submitted for the last array in 2nd dimension). Only relevant in A-synchronized transfers.
18h[(1)]	SRCCIDX	Source CCNT Index	Signed value specifying the byte address offset between frames within a block (3rd dimension). Valid values range from -32 768 and 32 767. A-synchronized transfers: The byte address offset from the beginning of the last source array in a frame to the beginning of the first source array in the next frame. AB-synchronized transfers: The byte address offset from the beginning of the first source array in a frame to the beginning of the first source array in the next frame.
	DSTCIDX	Destination CCNT index	Signed value specifying the byte address offset between frames within a block (3rd dimension). Valid values range from -32 768 and 32 767. A-synchronized transfers: The byte address offset from the beginning of the last destination array in a frame to the beginning of the first destination array in the next frame. AB-synchronized transfers: The byte address offset from the beginning of the first destination array in a frame to the beginning of the first destination array in the next frame.
1Ch	CCNT	Count for 3rd Dimension	Unsigned value specifying the number of frames in a block, where a frame is BCNT arrays of ACNT bytes. Valid values range from 1 to 65 535.
	RSVD	Reserved	Reserved. Always write 0 to this bit; writing 1 to this bit is not supported and attempts to do so may result in undefined behavior.

(1) The parameter sets must be accessed as 32-bit words.

10.3.5.2 EDMA Channel PaRAM Set Entry Fields

10.3.5.2.1 Channel Options Parameter (OPT)

The 32-bit channel options parameter (OPT) specifies the transfer configuration options. The channel options parameter (OPT) is shown in Figure 10-11 and described in Table 10-11.

Figure 10-11. Channel Options Parameter (OPT)

31	30 28	27 24	23	22	21	20	19 18	17 16
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC
R-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

15 12	11	10 8	7 4	3	2	1	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM
R/W-0			R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -n = value after reset

Table 10-11. Channel Options Parameters (OPT) Field Descriptions

Bit	Field	Description
31	PRIV	Privilege level (supervisor versus user) for the host/DSP/DMA that programmed this PaRAM set. This value is set with the EDMA master's privilege value when any part of the PaRAM set is written. 0h = User level privilege 1h = Supervisor level privilege
30-28	Reserved	Reserved. Always write 0 to this bit; writes of 1 to this bit are not supported and attempts to do so may result in undefined behavior.
27-24	PRIVID	Privilege identification for the external host/DSP/DMA that programmed this PaRAM set. This value is set with the EDMA master's privilege identification value when any part of the PaRAM set is written. Value = 0-Fh
23	ITCCHEN	Intermediate transfer completion chaining enable. 0h = Intermediate transfer complete chaining is disabled 1h = Intermediate transfer complete chaining is enabled When enabled, the chained event register (EDMACC_CER/EDMACC_CERH) bit is set on every intermediate chained transfer completion (upon completion of every intermediate TR in the PaRAM set, except the final TR in the PaRAM set). The bit (position) set in EDMACC_CER or EDMACC_CERH is the TCC value specified.
22	TCCHEN	Transfer complete chaining enable. 0h = Transfer complete chaining is disabled 1h = Transfer complete chaining is enabled When enabled, the chained event register (EDMACC_CER/EDMACC_CERH) bit is set on final chained transfer completion (upon completion of the final TR in the PaRAM set). The bit (position) set in EDMACC_CER or EDMACC_CERH is the TCC value specified.
21	ITCINTEN	Intermediate transfer completion interrupt enable. 0h = Intermediate transfer complete interrupt is disabled 1h = Intermediate transfer complete interrupt is enabled When enabled, the interrupt pending register (EDMACC_IPR/EDMACC_IPRH) bit is set on every intermediate transfer completion (upon completion of every intermediate TR in the PaRAM set, except the final TR in the PaRAM set). The bit (position) set in EDMACC_IPR or EDMACC_IPRH is the TCC value specified. To generate a completion interrupt to the DSP, the corresponding EDMACC_IER [TCC] / EDMACC_IERH [TCC] bit must be set.
20	TCINTEN	Transfer complete interrupt enable. 0h = Transfer complete interrupt is disabled 1h = Transfer complete interrupt is enabled When enabled, the interrupt pending register (EDMACC_IPR / EDMACC_IPRH) bit is set on transfer completion (upon completion of the final TR in the PaRAM set). The bit (position) set in EDMACC_IPR or EDMACC_IPRH is the TCC value specified. To generate a completion interrupt to the DSP, the corresponding EDMACC_IER[TCC] / EDMACC_IERH [TCC] bit must be set.
19-18	Reserved	Reserved. Always write 0 to this bit; writes of 1 to this bit are not supported and attempts to do so may result in undefined behavior.
17-12	TCC	Transfer complete code. This 6-bit code sets the relevant bit in the chaining enable register (CER [TCC] /CERH [TCC]) for chaining or in the interrupt pending register (IPR [TCC] / IPRH [TCC]) for interrupts. Value = 0-3Fh
11	TCCMODE	Transfer complete code mode. Indicates the point at which a transfer is considered completed for chaining and interrupt generation. 0h = Normal completion: A transfer is considered completed after the data has been transferred. 1h = Early completion: A transfer is considered completed after the EDMACC submits a TR to the EDMATC. TC may still be transferring data when the interrupt/chain is triggered.

Table 10-11. Channel Options Parameters (OPT) Field Descriptions (continued)

Bit	Field	Description
10-8	FWID	FIFO Width. Applies if either SAM or DAM is set to constant addressing mode. 0h = FIFO width is 8-bit 1h = FIFO width is 16-bit 2h = FIFO width is 32-bit 3h = FIFO width is 64-bit 4h = FIFO width is 128-bit 5h = FIFO width is 256-bit 6h - 7h = Reserved
7-4	Reserved	Reserved. Always write 0 to this bit; writes of 1 to this bit are not supported and attempts to do so may result in undefined behavior.
3	STATIC	Static set. 0h = Set is not static. The PaRAM set is updated or linked after a TR is submitted. A value of 0 should be used for DMA channels and for non-final transfers in a linked list of QDMA transfers. 1h = Set is static. The PaRAM set is not updated or linked after a TR is submitted. A value of 1 should be used for isolated QDMA transfers or for the final transfer in a linked list of QDMA transfers.
2	SYNCDIM	Transfer synchronization dimension. 0h = A-synchronized. Each event triggers the transfer of a single array of ACNT bytes. 1h = AB-synchronized. Each event triggers the transfer of BCNT arrays of ACNT bytes.
1	DAM	Destination address mode. 0h = Increment (INCR) mode. Destination addressing within an array increments. Destination is not a FIFO. 1h = Constant addressing (CONST) mode. Destination addressing within an array wraps around upon reaching FIFO width.
0	SAM	Source address mode. 0h = Increment (INCR) mode. Source addressing within an array increments. Source is not a FIFO. 1h = Constant addressing (CONST) mode. Source addressing within an array wraps around upon reaching FIFO width.

10.3.5.2.2 Channel Source Address (SRC)

The 32-bit source address parameter specifies the starting byte address of the source. For SAM in increment mode, there are no alignment restrictions imposed by EDMA. For SAM in constant addressing mode, the user must program the source address to be aligned to a 256-bit aligned address (5 LSBs of address must be 0). The EDMATC will signal an error, if this rule is violated. See Section 10.3.14.3 for additional details.

10.3.5.2.3 Channel Destination Address (DST)

The 32-bit destination address parameter specifies the starting byte address of the destination. For DAM in increment mode, there are no alignment restrictions imposed by EDMA. For DAM in constant addressing mode, the user must program the destination address to be aligned to a 256-bit aligned address (5 LSBs of address must be 0). The EDMATC will signal an error, if this rule is violated. See Section 10.3.14.3 for additional details.

10.3.5.2.4 Count for 1st Dimension (ACNT)

ACNT represents the number of bytes within the 1st dimension of a transfer. ACNT is a 16-bit unsigned value with valid values between 0 and 65 535. Therefore, the maximum number of bytes in an array is 65535 bytes (64K - 1 bytes). ACNT must be greater than or equal to 1 for a TR to be submitted to EDMATC. A transfer with ACNT equal to 0 is considered either a null or dummy transfer.

See Section 10.3.5.5 and Section 10.3.7.3 for details on dummy/null completion conditions.

10.3.5.2.5 Count for 2nd Dimension (BCNT)

BCNT is a 16-bit unsigned value that specifies the number of arrays of length ACNT. For normal operation, valid values for BCNT are between 1 and 65 535. Therefore, the maximum number of arrays in a frame is 65 535 (64K - 1 arrays). A transfer with BCNT equal to 0 is considered either a null or dummy transfer.

See Section 10.3.5.5 and Section 10.3.7.3 for details on dummy/null completion conditions.

10.3.5.2.6 Count for 3rd Dimension (CCNT)

CCNT is a 16-bit unsigned value that specifies the number of frames in a block. Valid values for CCNT are between 1 and 65 535. Therefore, the maximum number of frames in a block is 65 535 (64K - 1 frames). A transfer with CCNT equal to 0 is considered either a null or dummy transfer.

See Section 10.3.5.5 and Section 10.3.7.3 for details on dummy/null completion conditions.

10.3.5.2.7 BCNT Reload (BCNTRLD)

BCNTRLD is a 16-bit unsigned value used to reload the BCNT field once the last array in the 2nd dimension is transferred. This field is only used for A-synchronized transfers. In this case, the EDMACC decrements the BCNT value by 1 on each TR submission. When BCNT reaches 0, the EDMACC decrements CCNT and uses the BCNTRLD value to re-initialize the BCNT value.

For AB-synchronized transfers, the EDMACC submits the BCNT in the TR and the EDMATC decrements BCNT appropriately. For AB-synchronized transfers, BCNTRLD is not used.

10.3.5.2.8 Source B Index (SRCBIDX)

SRCBIDX is a 16-bit signed value (2s complement) used for source address modification between each array in the 2nd dimension. Valid values for SRCBIDX are between -32 768 and 32 767. It provides a byte address offset from the beginning of the source array to the beginning of the next source array. It applies to both A-synchronized and AB-synchronized transfers. Some examples:

- SRCBIDX = 0000h (0): no address offset from the beginning of an array to the beginning of the next array. All arrays are fixed to the same beginning address.
- SRCBIDX = 0003h (+3): the address offset from the beginning of an array to the beginning of the next array in a frame is 3 bytes. For example, if the current array begins at address 1000h, the next array begins at 1003h.
- SRCBIDX = FFFFh (-1): the address offset from the beginning of an array to the beginning of the next array in a frame is -1 byte. For example, if the current array begins at address 5054h, the next array begins at 5053h.

10.3.5.2.9 Destination B Index (DSTBIDX)

DSTBIDX is a 16-bit signed value (2s complement) used for destination address modification between each array in the 2nd dimension. Valid values for DSTBIDX are between -32 768 and 32 767. It provides a byte address offset from the beginning of the destination array to the beginning of the next destination array within the current frame. It applies to both A-synchronized and AB-synchronized transfers. See Section 10.3.5.2.8 (SRCBIDX) for examples.

10.3.5.2.10 Source C Index (SRCCIDX)

SRCCIDX is a 16-bit signed value (2s complement) used for source address modification in the 3rd dimension. Valid values for SRCCIDX are between -32 768 and 32 767. It provides a byte address offset from the beginning of the current array (pointed to by SRC address) to the beginning of the first source array in the next frame. It applies to both A-synchronized and AB-synchronized transfers. Note that when SRCCIDX is applied, the current array in an A-synchronized transfer is the last array in the frame (Figure 10-8), while the current array in an AB-synchronized transfer is the first array in the frame (Figure 10-9).

10.3.5.2.11 Destination C Index (DSTCIDX)

DSTCIDX is a 16-bit signed value (2s complement) used for destination address modification in the 3rd dimension. Valid values are between -32 768 and 32 767. It provides a byte address offset from the beginning of the current array (pointed to by DST address) to the beginning of the first destination array TR in the next frame. It applies to both A-synchronized and AB-synchronized transfers. Note that when DSTCIDX is applied, the current array in an A-synchronized transfer is the last array in the frame (Figure 10-8), while the current array in a AB-synchronized transfer is the first array in the frame (Figure 10-9).

10.3.5.2.12 Link Address (LINK)

The EDMACC provides a mechanism, called linking, to reload the current PaRAM set upon its natural termination (that is, after the count fields are decremented to 0) with a new PaRAM set. The 16-bit parameter LINK specifies the byte address offset in the PaRAM from which the EDMACC loads/reloads the next PaRAM set during linking.

The user must program the link address to point to a valid aligned 32-byte PaRAM set. The 5 LSBs of the LINK field should be cleared to 0.

The EDMACC ignores the upper 2 bits of the LINK entry, allowing the programmer the flexibility of programming the link address as either an absolute/literal byte address or use the PaRAM-base-relative offset address. Therefore, if the user makes use of the literal address with a range from 4000h to 7FFFh, it will be treated as a PaRAM-base-relative value of 0000h to 3FFFh.

The user should make sure to program the LINK field correctly, so that link update is requested from a PaRAM address that falls in the range of the available PaRAM addresses on the device.

A LINK value of FFFFh is referred to as a NULL link that should cause the EDMACC to perform an internal write of 0 to all entries of the current PaRAM set, except for the LINK field that is set to FFFFh. Also, see Section 10.3.7 for details on terminating a transfer.

10.3.5.3 Null PaRAM Set

A null PaRAM set is defined as a PaRAM set where all count fields (ACNT, BCNT, and CCNT) are cleared to 0. If a PaRAM set associated with a channel is a NULL set, then when serviced by the EDMACC, the bit corresponding to the channel is set in the associated event missed register (EDMACC_EMR, EDMACC_EMRH, or EDMACC_QEMR). This bit remains set in the associated secondary event register (EDMACC_SER, EDMACC_SERH, or EDMACC_QSER). This implies that any future events on the same channel are ignored by the EDMACC and the user is required to clear the bit in EDMACC_SER, EDMACC_SERH, or EDMACC_QSER for the channel. This is considered an error condition, since events are not expected on a channel that is configured as a null transfer. See and for more information on the EDMACC_SER and EDMACC_EMR registers, respectively.

10.3.5.4 Dummy PaRAM Set

A dummy PaRAM set is defined as a PaRAM set where at least one of the count fields (ACNT, BCNT, or CCNT) is cleared to 0 and at least one of the count fields is nonzero.

If a PaRAM set associated with a channel is a dummy set, then when serviced by the EDMACC, it will not set the bit corresponding to the channel (DMA/QDMA) in the event missed register (EDMACC_EMR, EDMACC_EMRH, or EDMACC_QEMR) and the secondary event register (EDMACC_SER, EDMACC_SERH, or EDMACC_QSER) bit gets cleared similar to a normal transfer. Future events on that channel are serviced. A dummy transfer is a legal transfer of 0 bytes. See and for more information on the EDMACC_SER and EDMACC_EMR registers, respectively.

10.3.5.5 Dummy Versus Null Transfer Comparison

There are some differences in the way the EDMACC logic treats a dummy versus a null transfer request. A null transfer request is an error condition, but a dummy transfer is a legal transfer of 0 bytes. A null transfer causes an error bit (E *n*) in EDMACC_EMR to get set and the E *n* bit in EDMACC_SER remains set, essentially preventing any further transfers on that channel without clearing the associated error registers.

Table 10-12 summarizes the conditions and effects of null and dummy transfer requests.

Table 10-12. Dummy and Null Transfer Request

Feature	Null TR	Dummy TR
EDMACC_EMR/EDMACC_EMRH/EDMACC_QEMR is set	Yes	No
EDMACC_SER/EDMACC_SERH/EDMACC_QSER remains set	Yes	No
Link update (STATIC = 0 in OPT)	Yes	Yes
EDMACC_QER is set	Yes	Yes
EDMACC_IPR/EDMACC_IPRH EDMACC_CER/EDMACC_CERH is set using early completion	Yes	Yes

10.3.5.6 Parameter Set Updates

When a TR is submitted for a given DMA/QDMA channel and its corresponding PaRAM set, the EDMACC is responsible for updating the PaRAM set in anticipation of the next trigger event. For events that are not final, this includes address and count updates; for final events, this includes the link update.

The specific PaRAM set entries that are updated depend on the channel's synchronization type (A-synchronized or B-synchronized) and the current state of the PaRAM set. A B-update refers to the decrementing of BCNT in the case of A-synchronized transfers after the submission of successive TRs. A C-update refers to the decrementing of CCNT in the case of A-synchronized transfers after BCNT TRs for ACNT byte transfers have submitted. For AB-synchronized transfers, a C-update refers to the decrementing of CCNT after submission of every transfer request.

See Table 10-13 for details and conditions on the parameter updates. A link update occurs when the PaRAM set is exhausted, as described in Section 10.3.5.7.

After the TR is read from the PaRAM (and is in process of being submitted to EDMATC), the following fields are updated if needed:

- A-synchronized: BCNT, CCNT, SRC, DST
- AB-synchronized: CCNT, SRC, DST

The following fields are not updated (except for during linking, where all fields are overwritten by the link PaRAM set):

- A-synchronized: ACNT, BCNTRLD, SRCBIDX, DSTBIDX, SRCCIDX, DSTCIDX, OPT, LINK
- AB-synchronized: ACNT, BCNT, BCNTRLD, SRCBIDX, DSTBIDX, SRCCIDX, DSTCIDX, OPT, LINK

Note that PaRAM updates only pertain to the information that is needed to properly submit the next transfer request to the EDMATC. Updates that occur while data is moved within a transfer request are tracked within the transfer controller, and is detailed in Section 10.3.14. For A-synchronized transfers, the EDMACC always submits a TRP for ACNT bytes (BCNT = 1 and CCNT = 1). For AB-synchronized transfers, the EDMACC always submits a TRP for ACNT bytes of BCNT arrays (CCNT = 1). The EDMATC is responsible for updating source and destination addresses within the array based on ACNT and FWID (in OPT). For AB-synchronized transfers, the EDMATC is also responsible to update source and destination addresses between arrays based on SRCBIDX and DSTBIDX.

Table 10-13 shows the details of parameter updates that occur within EDMACC for A-synchronized and AB-synchronized transfers.

Table 10-13. Parameter Updates in EDMACC (for Non-Null, Non-Dummy PaRAM Set)

	A-Synchronized Transfer			AB-Synchronized Transfer		
	B-Update	C-Update	Link Update	B-Update	C-Update	Link Update
Condition:	**BCNT > 1**	**BCNT == 1 && CCNT > 1**	**BCNT == 1 && CCNT == 1**	**N/A**	**CCNT > 1**	**CCNT == 1**
SRC	+= SRCBIDX	+= SRCCIDX	= Link.SRC	in EDMATC	+= SRCCIDX	= Link.SRC
DST	+= DSTBIDX	+= DSTCIDX	= Link.DST	in EDMATC	+= DSTCIDX	= Link.DST

Table 10-13. Parameter Updates in EDMACC (for Non-Null, Non-Dummy PaRAM Set) (continued)

	A-Synchronized Transfer			AB-Synchronized Transfer		
	B-Update	C-Update	Link Update	B-Update	C-Update	Link Update
Condition:	BCNT > 1	BCNT == 1 && CCNT > 1	BCNT == 1 && CCNT == 1	N/A	CCNT > 1	CCNT == 1
ACNT	None	None	= Link.ACNT	None	None	= Link.ACNT
BCNT	-= 1	= BCNTRLD	= Link.BCNT	in EDMATC	N/A	= Link.BCNT
CCNT	None	-= 1	= Link.CCNT	in EDMATC	-=1	= Link.CCNT
SRCBIDX	None	None	= Link.SRCBIDX	in EDMATC	None	= Link.SRCBIDX
DSTBIDX	None	None	= Link.DSTBIDX	None	None	= Link.DSTBIDX
SRCCIDX	None	None	= Link.SRCBIDX	in EDMATC	None	= Link.SRCBIDX
DSTCIDX	None	None	= Link.DSTBIDX	None	None	= Link.DSTBIDX
LINK	None	None	= Link.LINK	None	None	= Link.LINK
BCNTRLD	None	None	= Link.BCNTRLD	None	None	= Link.BCNTRLD
OPT (1)	None	None	= LINK.OPT	None	None	= LINK.OPT

(1) In all cases, no updates occur if OPT.STATIC == 1 for the current PaRAM set.

NOTE: The EDMACC includes no special hardware to detect when an indexed address update calculation overflows/underflows. The address update will wrap across boundaries as programmed by the user. The user should ensure that no transfer is allowed to cross internal port boundaries between peripherals. A single TR must target a single source/destination slave endpoint.

10.3.5.7 Linking Transfers

The EDMACC provides a mechanism known as linking, which allows the entire PaRAM set to be reloaded from a location within the PaRAM memory map (for both DMA and QDMA channels). Linking is especially useful for maintaining ping-pong buffers, circular buffering, and repetitive/continuous transfers with no CPU intervention. Upon completion of a transfer, the current transfer parameters are reloaded with the parameter set pointed to by the 16-bit link address field (of the current parameter set). Linking only occurs when the STATIC bit in OPT is cleared to 0.

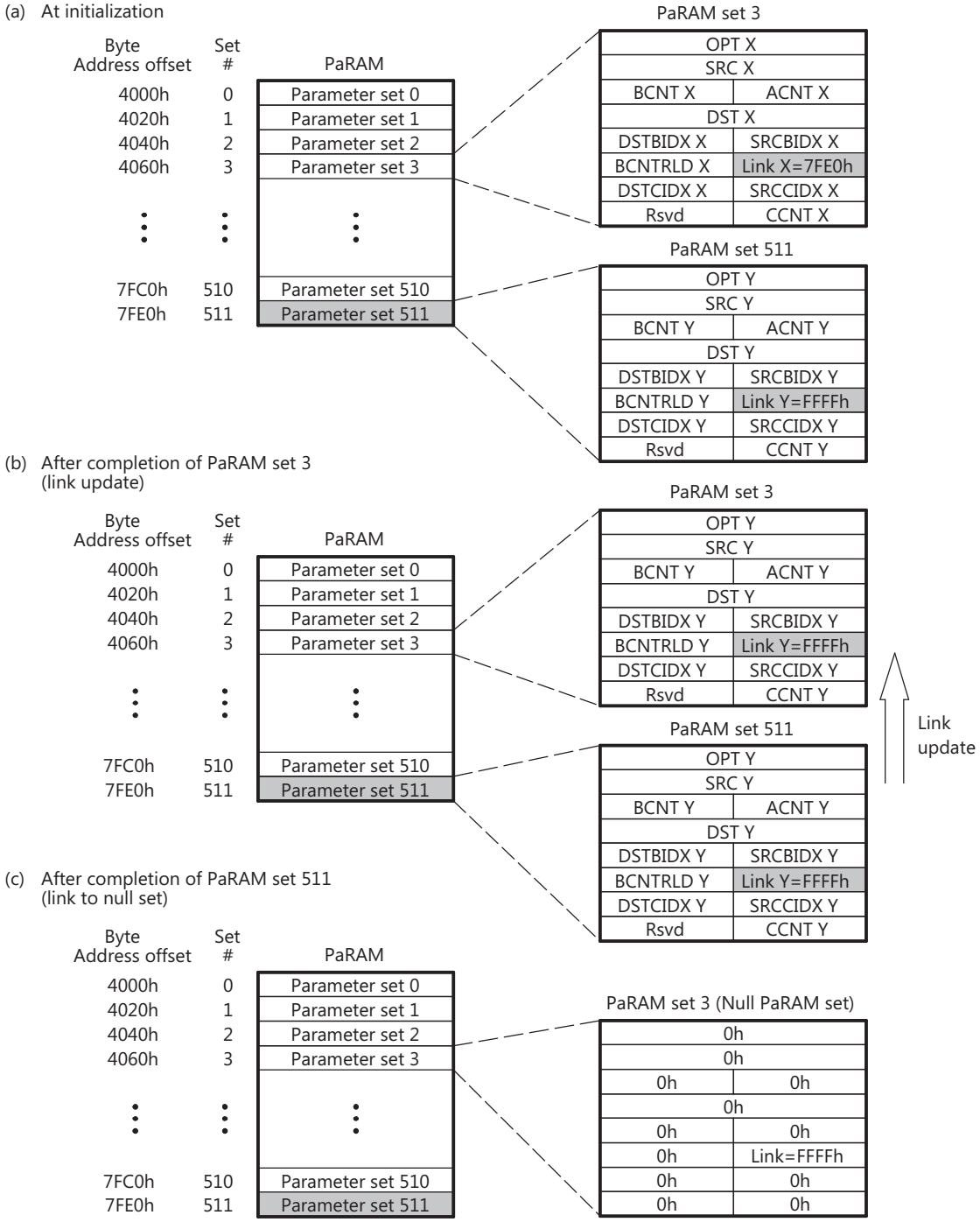
NOTE: A transfer (DMA or QDMA) should always be linked to another useful transfer. If it is required to terminate a transfer, the transfer should be linked to a NULL set. See Section 10.3.5.3.

The link update occurs after the current PaRAM set event parameters have been exhausted. An event's parameters are exhausted when the EDMA channel controller has submitted all the transfers associated with the PaRAM set.

A link update occurs for null and dummy transfers depending on the state of the STATIC bit in OPT and the LINK field. In both cases (null or dummy), if the value of LINK is FFFFh, then a null PaRAM set (with all 0s and LINK set to FFFFh) is written to the current PaRAM set. Similarly, if LINK is set to a value other than FFFFh, then the appropriate PaRAM location pointed to by LINK is copied to the current PaRAM set.

Once the channel completion conditions are met for an event, the transfer parameters located at the link address are loaded into the current DMA or QDMA channel's associated parameter set. The EDMACC reads the entire PaRAM set (8 words) from the PaRAM set specified by LINK and writes all 8 words to the PaRAM set associated with the current channel. Figure 10-12 shows an example of a linked transfer.

Figure 10-12. Linked Transfer Example

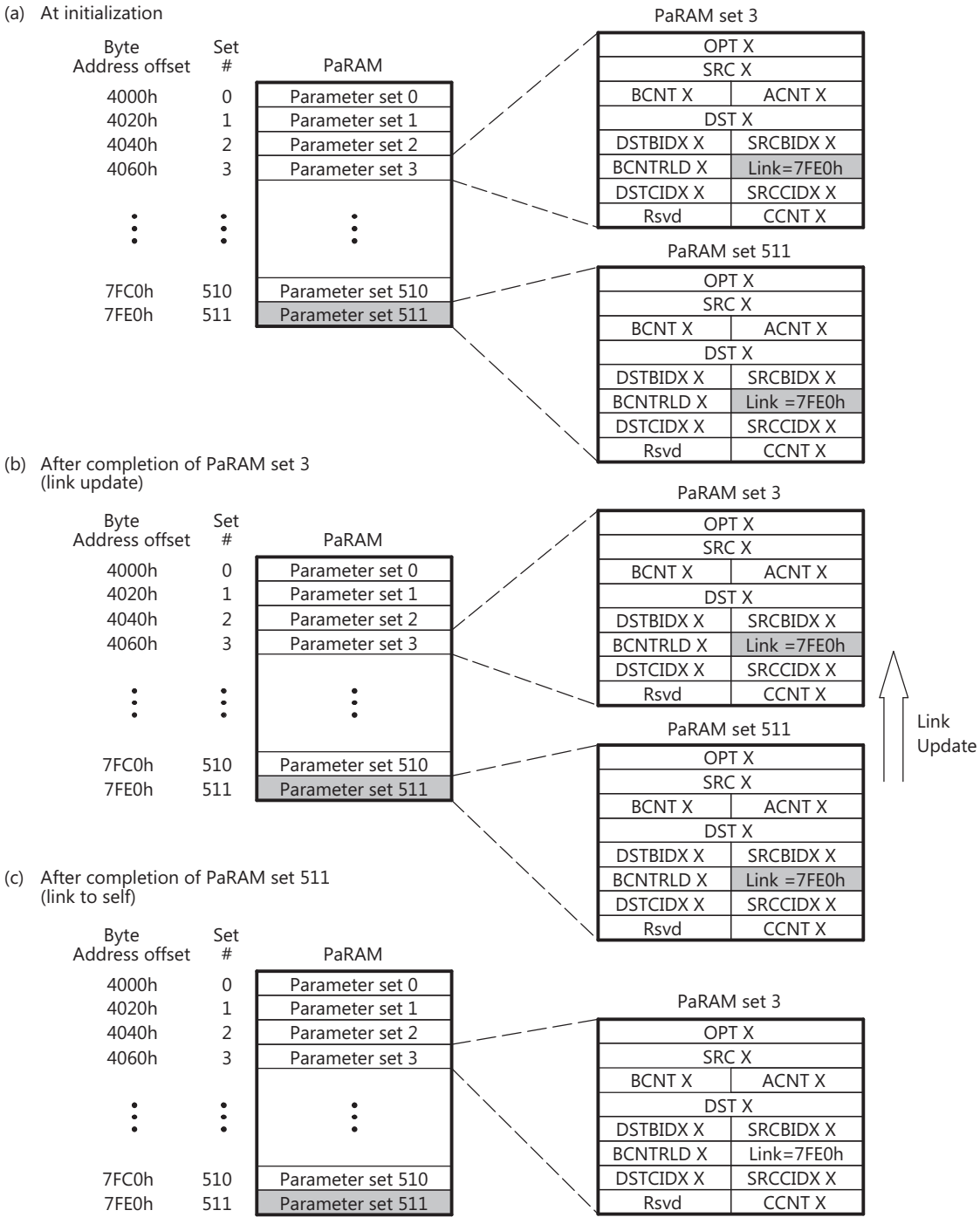


Any PaRAM set in the PaRAM can be used as a link/reload parameter set; however, it is recommended that the PaRAM sets associated with peripheral synchronization events (see Section 10.3.8) should only be used for linking if the synchronization event isolated with the channel mapped to that PaRAM set is disabled.

If a PaRAM set location is mapped to a QDMA channel (by QCHMAP *n*), then copying the link PaRAM set onto the current QDMA channel PaRAM set is recognized as a trigger event. It is latched in EDMACC_QER since a write to the trigger word was performed. This feature can be used to create a linked list of transfers using a single QDMA channel and multiple PaRAM sets. See Section 10.3.6.2.

Link-to-self transfers replicate the behavior of auto-initialization, which facilitates the use of circular buffering and repetitive transfers. After an EDMA channel exhausts its current PaRAM set, it reloads all the parameter set entries from another PaRAM set, which is initialized with values identical to the original PaRAM set. Figure 10-13 shows an example of a linked-to-self transfer. Here, the PaRAM set 511 has the link field pointing to the address of parameter set 511 (linked-to-self).

Figure 10-13. Link-to-Self Transfer Example



NOTE: If the STATIC bit in OPT is set for a PaRAM set, then link updates are not performed. The link updates performed internally by the EDMACC are atomic. This implies that when the EDMACC is updating a PaRAM set, accesses to PaRAM by other EDMA programmer's (for example, CPU configuration accesses) are not allowed. Also for QDMA, for example, if the first word of the PaRAM entry is defined as a trigger word, EDMACC logic assures that all 8 PaRAM words are updated before the new QDMA event can trigger the transfer for that PaRAM entry.

10.3.5.8 Constant Addressing Mode Transfers/Alignment Issues

If either SAM or DAM is set to 1 (constant addressing mode), then the source or destination address must be aligned to a 256-bit aligned address, respectively, and the corresponding BIDX should be an even multiple of 32 bytes (256 bit). The EDMACC does not recognize errors here, but the EDMATC asserts an error if this is not true. See Section 10.3.14.3.

NOTE: The constant addressing (CONST) mode has limited applicability. The EDMA should be configured for the constant addressing mode (SAM/DAM = 1) only if the transfer source or destination (on-chip memory, off-chip memory controllers, slave peripherals) support the constant addressing mode. If constant addressing mode is not supported, the similar logical transfer can be achieved using the increment (INCR) mode (SAM/DAM = 0) by appropriately programming the count and indices values.

10.3.5.9 Element Size

The EDMA controller does not use element-size and element-indexing. Instead, all transfers are defined in terms of all three dimensions: ACNT, BCNT, and CCNT. An element-indexed transfer is logically achieved by programming ACNT to the size of the element and BCNT to the number of elements that need to be transferred. For example, if there are 16-bit data and 256 samples that must be transferred to a serial port, the user can only do this by programming the ACNT = 2 (2 bytes) and BCNT = 256.

10.3.6 Initiating a DMA Transfer

There are multiple ways to initiate a programmed data transfer using the EDMA channel controller. Transfers on DMA channels are initiated by three sources:

- **Event-triggered transfer request** (this is the most typical usage of EDMA controller): A peripheral, system, or externally-generated event triggers a transfer request.
- **Manually-triggered transfer request**: The CPU manually triggers a transfer by writing 1 to the corresponding bit in the event set register (EDMACC_ESR/EDMACC_ESRH).
- **Chain-triggered transfer request**: A transfer is triggered on the completion of another transfer or sub-transfer.

Transfers on QDMA channels are initiated by two sources:

- **Auto-triggered transfer request**: Writing to the programmed trigger word triggers a transfer.
- **Link-triggered transfer requests**: Writing to the trigger word triggers the transfer when linking occurs.

10.3.6.1 DMA Channel

10.3.6.1.1 Event-Triggered Transfer Request

When an event is asserted from a peripheral or device pins, it gets latched in the corresponding bit of the event register (EDMACC_ER.E *n* = 1). If the corresponding event in the event enable register (EDMACC_EER) is enabled (EDMACC_EER.E *n* = 1), then the EDMACC prioritizes and queues the event in the appropriate event queue. When the event reaches the head of the queue, it is evaluated for submission as a transfer request to the transfer controller.

If the PaRAM set is valid (not a NULL set), then a transfer request packet (TRP) is submitted to the EDMATC and the E *n* bit in EDMACC_ER is cleared. At this point, a new event can be safely received by the EDMACC.

If the PaRAM set associated with the channel is a NULL set (see Section 10.3.5.3), then no transfer request (TR) is submitted and the corresponding E *n* bit in EDMACC_ER is cleared and simultaneously the corresponding channel bit is set in the event miss register (EDMACC_EMR.E *n* = 1) to indicate that the event was discarded due to a null TR being serviced. Good programming practices should include cleaning the event missed error before re-triggering the DMA channel.

When an event is received, the corresponding event bit in the event register is set (EDMACC_ER.E *n* = 1), regardless of the state of EDMACC_EER.E *n*. If the event is disabled when an external event is received (EDMACC_ER.E *n* = 1 and EDMACC_EER.E *n* = 0), the EDMACC_ER.E *n* bit remains set. If the event is subsequently enabled (EDMACC_EER.E *n* = 1), then the pending event is processed by the EDMACC and the TR is processed/submitted, after which the EDMACC_ER.E *n* bit is cleared.

If an event is being processed (prioritized or is in the event queue) and another sync event is received for the same channel prior to the original being cleared (EDMACC_ER.E *n* ! = 0), then the second event is registered as a missed event in the corresponding bit of the event missed register (EDMACC_EMR.E *n* = 1).

The mapping of the peripheral/external events to the EDMA controller inputs (and DMA channels, respectively) is presented in Section 10.2.1.

10.3.6.1.2 Manually-Triggered Transfer Request

A DMA transfer is initiated by writing to the event set register (EDMACC_ESR) by the CPU (or any EDMA programmer). Writing 1 to an event bit in the EDMACC_ESR results in the event being prioritized/queued in the appropriate event queue, regardless of the state of the EDMACC_EER.E *n* bit. When the event reaches the head of the queue, it is evaluated for submission as a transfer request to the transfer controller.

As in the event-triggered transfers, if the PaRAM set associated with the channel is valid (it is not a NULL set) then the TR is submitted to the associated EDMATC and the channel can be triggered again.

If the PaRAM set associated with the channel is a NULL set (see Section 10.3.5.3), then no transfer request (TR) is submitted and the corresponding E *n* bit in EDMACC_ER is cleared and simultaneously the corresponding channel bit is set in the event miss register (EDMACC_EMR.E *n* = 1) to indicate that the event was discarded due to a null TR being serviced. Good programming practices should include clearing the event missed error before re-triggering the DMA channel.

If an event is being processed (prioritized or is in the event queue) and the same channel is manually set by a write to the corresponding channel bit of the event set register (EDMACC_ESR.E *n* = 1) prior to the original being cleared (EDMACC_ESR.E *n* = 0), then the second event is registered as a missed event in the corresponding bit of the event missed register (EDMACC_EMR.E *n* = 1).

10.3.6.1.3 Chain-Triggered Transfer Request

Chaining is a mechanism by which the completion of one transfer automatically sets the event for another channel. When a chained completion code is detected, the value of which is dictated by the transfer completion code (TCC[5-0] in OPT of the PaRAM set associated with the channel), it results in the corresponding bit in the chained event register (EDMACC_CER) to be set (EDMACC_CER.E[TCC] = 1).

Once a bit is set in EDMACC_CER, the EDMACC prioritizes and queues the event in the appropriate event queue. When the event reaches the head of the queue, it is evaluated for submission as a transfer request to the transfer controller.

As in the event-triggered transfers, if the PaRAM set associated with the channel is valid (it is not a null set) then the TR is submitted to the associated EDMATC and the channel can be triggered again.

If the PaRAM set associated with the channel is a NULL set (see Section 10.3.5.3), then no transfer request (TR) is submitted and the corresponding En bit in EDMACC_CER is cleared and simultaneously the corresponding channel bit is set in the event miss register (EDMACC_EMR.E *n* = 1) to indicate that the event was discarded due to a null TR being serviced. In this case, the error condition must be cleared by software before the DMA channel can be re-triggered. Good programming practices might include clearing the event missed error before re-triggering the DMA channel.

If a chaining event is being processed (prioritized or queued) and another chained event is received for the same channel prior to the original being cleared (EDMACC_CER.E *n* != 0), then the second chained event is registered as a missed event in the corresponding channel bit of the event missed register (EDMACC_EMR.E *n* = 1).

NOTE: Chained event registers (EDMACC_CER/EDMACC_CERH), event registers (EDMACC_ER/EDMACC_ERH), and event set registers (EDMACC_ESR/EDMACC_ESRH) operate independently. An event (E *n*) can be triggered by any of the trigger sources (event-triggered, manually-triggered, or chain-triggered).

10.3.6.2 QDMA Channels

10.3.6.2.1 Auto-triggered and Link-Triggered Transfer Request

QDMA-based transfer requests are issued when a QDMA event gets latched in the QDMA event register (EDMACC_QER.E *n* = 1). A bit corresponding to a QDMA channel is set in the QDMA event register (EDMACC_QER) when the following occurs:

- A CPU (or any EDMA programmer) write occurs to a PaRAM address that is defined as a QDMA channel trigger word (programmed in the QDMA channel mapping register (QCHMAP *n*)) for the particular QDMA channel and the QDMA channel is enabled via the QDMA event enable register EDMACC_QEER.E *n* = 1).
- EDMACC performs a link update on a PaRAM set address that is configured as a QDMA channel matches QCHMAP *n* settings) and the corresponding channel is enabled via the QDMA event enable register (EDMACC_QEER.E *n* = 1).

Once a bit is set in EDMACC_QER, the EDMACC prioritizes and queues the event in the appropriate event queue. When the event reaches the head of the queue, it is evaluated for submission as a transfer request to the transfer controller.

As in the event-triggered transfers, if the PaRAM set associated with the channel is valid (it is not a null set) then the TR is submitted to the associated EDMATC and the channel can be triggered again.

If a bit is already set in EDMACC_QER (EDMACC_QER.E *n* = 1) and a second QDMA event for the same QDMA channel occurs prior to the original being cleared, the second QDMA event gets captured in the QDMA event miss register (EDMACC_QEMR.E *n* = 1).

10.3.6.3 Comparison Between DMA and QDMA Channels

The primary difference between DMA and QDMA channels is the event/channel synchronization. QDMA events are either auto-triggered or link triggered. Auto-triggering allows QDMA channels to be triggered by CPU(s) with a minimum number of linear writes to PaRAM. Link triggering allows a linked list of transfers to be executed, using a single QDMA PaRAM set and multiple link PaRAM sets.

A QDMA transfer is triggered when a CPU (or other EDMA programmer) writes to the trigger word of the QDMA channel parameter set (auto-triggered) or when the EDMACC performs a link update on a PaRAM set that has been mapped to a QDMA channel (link triggered). Note that for CPU-triggered (manually triggered) DMA channels, in addition to writing to the PaRAM set, it is required to write to the event set register (EDMACC_ESR) to kick-off the transfer.

QDMA channels are typically for cases where a single event will accomplish a complete transfer since the CPU (or other EDMA programmer) must reprogram some portion of the QDMA PaRAM set in order to re-trigger the channel. In other words, QDMA transfers are programmed with BCNT = CCNT = 1 for A-synchronized transfers, and CCNT = 1 for AB-synchronized transfers.

Additionally, since linking is also supported (if STATIC = 0 in OPT) for QDMA transfers, it allows the user to initiate a linked list of QDMAs, so when EDMACC copies over a link PaRAM set (including the write to the trigger word), the current PaRAM set mapped to the QDMA channel will automatically be recognized as a valid QDMA event and initiate another set of transfers as specified by the linked set.

10.3.7 Completion of a DMA Transfer

A parameter set for a given channel is complete when the required number of transfer requests is submitted (based on receiving the number of synchronization events). The expected number of TRs for a non-null/non-dummy transfer is shown in Table 10-14 for both synchronization types along with state of the PaRAM set prior to the final TR being submitted. When the counts (BCNT and/or CCNT) are this value, the next TR results in:

- Final chaining or interrupt codes to be sent by the transfer controllers (instead of intermediate).
- Link updates (linking to either null or another valid link set).

Table 10-14. Expected Number of Transfers for Non-Null Transfer

Sync Mode	Counts at Time 0	Total # Transfers	Counts Prior to Final TR
A-synchronized	ACNT BCNT CCNT	(BCNT × CCNT) TRs of ACNT bytes each	BCNT == 1 && CCNT == 1
AB-synchronized	ACNT BCNT CCNT	CCNT TRs for ACNT × BCNT bytes each	CCNT == 1

The user must program the PaRAM OPT field with a specific transfer completion code (TCC) along with the other OPT fields (TCCHEN, TCINTEN, ITCCHEN, and ITCINTEN bits) to indicate whether the completion code is to be used for generating a chained event or/and for generating an interrupt upon completion of a transfer.

The specific TCC value (6-bit binary value) programmed dictates which of the 64-bits in the chain event register (EDMACC_CER[TCC]) and/or interrupt pending register (EDMACC_IPR[TCC]) is set. See Section 10.3.11 for details on interrupts and Section 10.3.10 for details on chaining.

The user can also selectively program whether the transfer controller sends back completion codes on completion of the final transfer request (TR) of a parameter set (TCCHEN or TCINTEN), for all but the final transfer request (TR) of a parameter set (ITCCHEN or ITCINTEN), or for all TRs of a parameter set (both). See Section 10.3.10 for details on chaining (intermediate/final chaining) and Section 10.3.11 for details on intermediate/final interrupt completion.

A completion detection interface exists between the EDMA channel controller and transfer controller(s). This interface sends back information from the transfer controller to the channel controller to indicate that a specific transfer is completed.

All DMA/QDMA PaRAM sets must also specify a link address value. For repetitive transfers such as ping-pong buffers, the link address value should point to another predefined PaRAM set. Alternatively, a non-repetitive transfer should set the link address value to the null link value. The null link value is defined as FFFFh. See Section 10.3.5.7 for more details.

NOTE: Any incoming events that are mapped to a null PaRAM set results in an error condition. The error condition should be cleared before the corresponding channel is used again. See Section 10.3.5.5.

There are three ways the EDMACC gets updated/informed about a transfer completion: normal completion, early completion, and dummy/null completion. This applies to both chained events and completion interrupt generation.

10.3.7.1 Normal Completion

In normal completion mode (TCCMODE = 0 in OPT), the transfer or sub-transfer is considered to be complete when the EDMA channel controller receives the completion codes from the EDMA transfer controller. In this mode, the completion code to the channel controller is posted by the transfer controller after it receives a signal from the destination peripheral. Normal completion is typically used to generate an interrupt to inform the CPU that a set of data is ready for processing.

10.3.7.2 Early Completion

In early completion mode (TCCMODE = 1 in OPT), the transfer is considered to be complete when the EDMA channel controller submits the transfer request (TR) to the EDMA transfer controller. In this mode, the channel controller generates the completion code internally. Early completion is typically useful for chaining, as it allows subsequent transfers to be chained-triggered while the previous transfer is still in progress within the transfer controller, maximizing the overall throughput of the set of the transfers.

10.3.7.3 Dummy or Null Completion

This is a variation of early completion. Dummy or null completion is associated with a dummy set (Section 10.3.5.4) or null set (Section 10.3.5.3). In both cases, the EDMA channel controller does not submit the associated transfer request to the EDMA transfer controller(s). However, if the set (dummy/null) has the OPT field programmed to return completion code (intermediate/final interrupt/chaining completion), then it will set the appropriate bits in the interrupt pending registers (EDMACC_IPR/EDMACC_IPRH) or chained event register (EDMACC_CER/EDMACC_CERH). The internal early completion path is used by the channel controller to return the completion codes internally (that is, EDMACC generates the completion code).

10.3.8 Event, Channel, and PaRAM Mapping

Most of the DMA channels are tied to a specific hardware peripheral event, thus allowing transfers to be triggered by events from device peripherals or external hardware. A DMA channel typically requests a data transfer when it receives its event (apart from manually-triggered, chain-triggered, and other transfers). The amount of data transferred per synchronization event depends on the channel's configuration (ACNT, BCNT, CCNT, etc.) and the synchronization type (A-synchronized or AB-synchronized).

The association of an event to a channel is fixed. Each of the DMA channels has one specific event associated with it. For the synchronization events associated with each of the programmable DMA channels, see Section 10.2.1.

In an application, if a channel does not use the associated synchronization event or if it does not have an associated synchronization event (unused), that channel can be used for manually-triggered or chained-triggered transfers, for linking/reloading, or as a QDMA channel.

10.3.8.1 DMA Channel to PaRAM Mapping

The mapping between the DMA channel numbers and the PaRAM sets is programmable. The DMA channel *n* mapping register (DCHMAP *n*) in the EDMACC provide programmability that allows the user to map the DMA channels to any of the PaRAM sets in the PaRAM memory map. Figure 10-14 illustrates the use of DCHMAP. There is one DCHMAP register per channel.

10.3.8.2 QDMA Channel to PaRAM Mapping

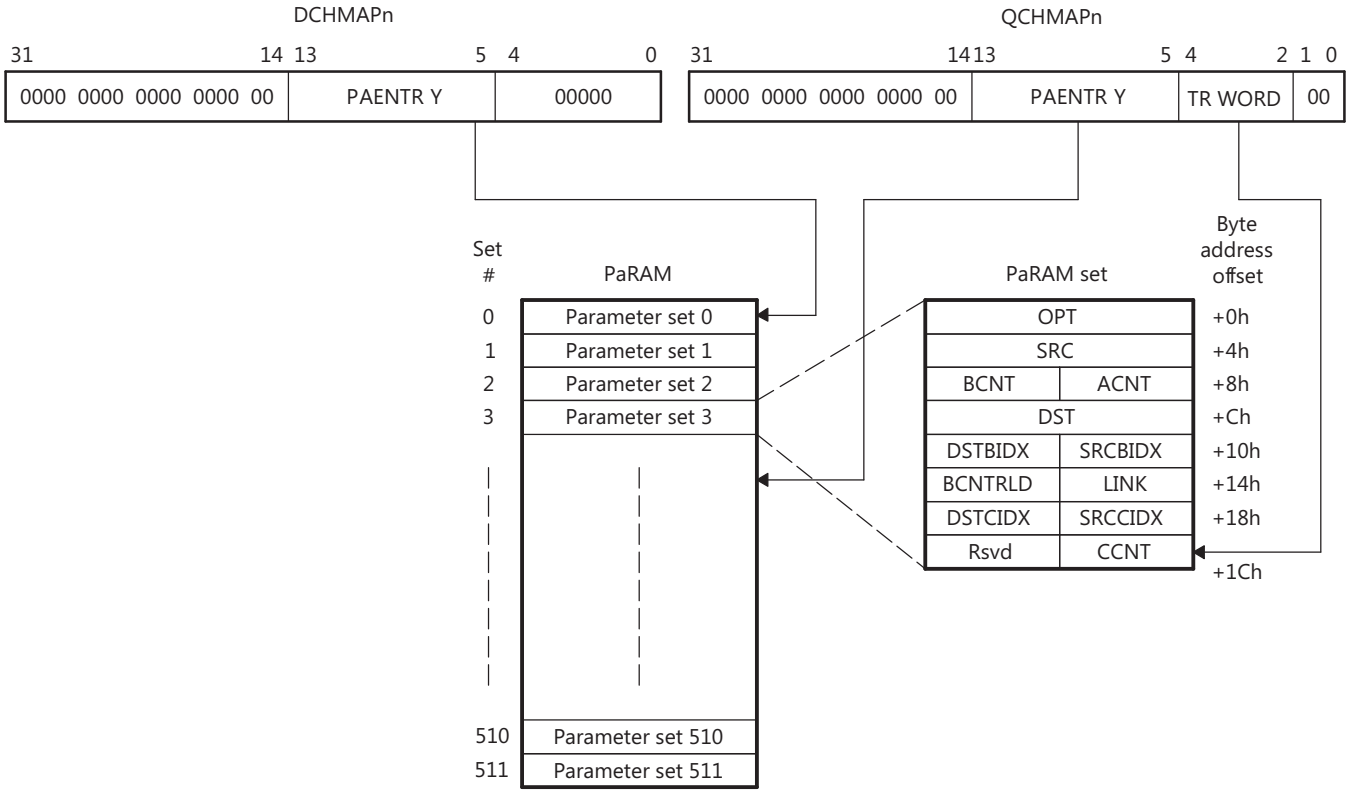
The mapping between the QDMA channels and the PaRAM sets is programmable. The QDMA channel *n* mapping register (QCHMAP *n*) in the EDMACC allows the user to map the QDMA channels to any of the PaRAM sets in the PaRAM memory map. Figure 10-14 illustrates the use of QCHMAP.

Additionally, QCHMAP allows the user to program the trigger word in the PaRAM set for the QDMA channel. A trigger word is one of the 8 words in the PaRAM set. For a QDMA transfer to occur, a valid TR synchronization event for EDMACC is a write to the trigger word in the PaRAM set pointed to by QCHMAP for a particular QDMA channel.

NOTE: By default, QDMA channels are mapped to PaRAM set 0. Care must be taken to appropriately remap PaRAM set 0 before it is used.

QDMA channel to PaRAM mapping is shown in Figure 10-14.

Figure 10-14. DMA/QDMA Channel to PaRAM Mapping



10.3.9 EDMA Channel Controller Regions

The EDMA channel controller (EDMACC) divides its address space into 8 regions. Individual channel resources are assigned to a specific region, where each region is typically assigned to a specific EDMA programmer.

The user can design the application software to use regions or to ignore them altogether. The user can use active memory protection in conjunction with regions so that only a specific EDMA programmer (for example, privilege identification) or privilege level (for example, user vs. supervisor) is allowed access to a given region, and thus to a given DMA or QDMA channel. This allows robust system-level DMA code where each EDMA programmer only modifies the state of the assigned resources. Memory protection is described in Section 10.3.12.

10.3.9.1 Region Overview

The EDMACC memory-mapped registers are divided in three main categories:

1. Global registers
2. Global region channel registers
3. Shadow region channel registers

The global registers are located at a single/fixed location in the EDMACC memory map. These registers control EDMA resource mapping and provide debug visibility and error tracking information.

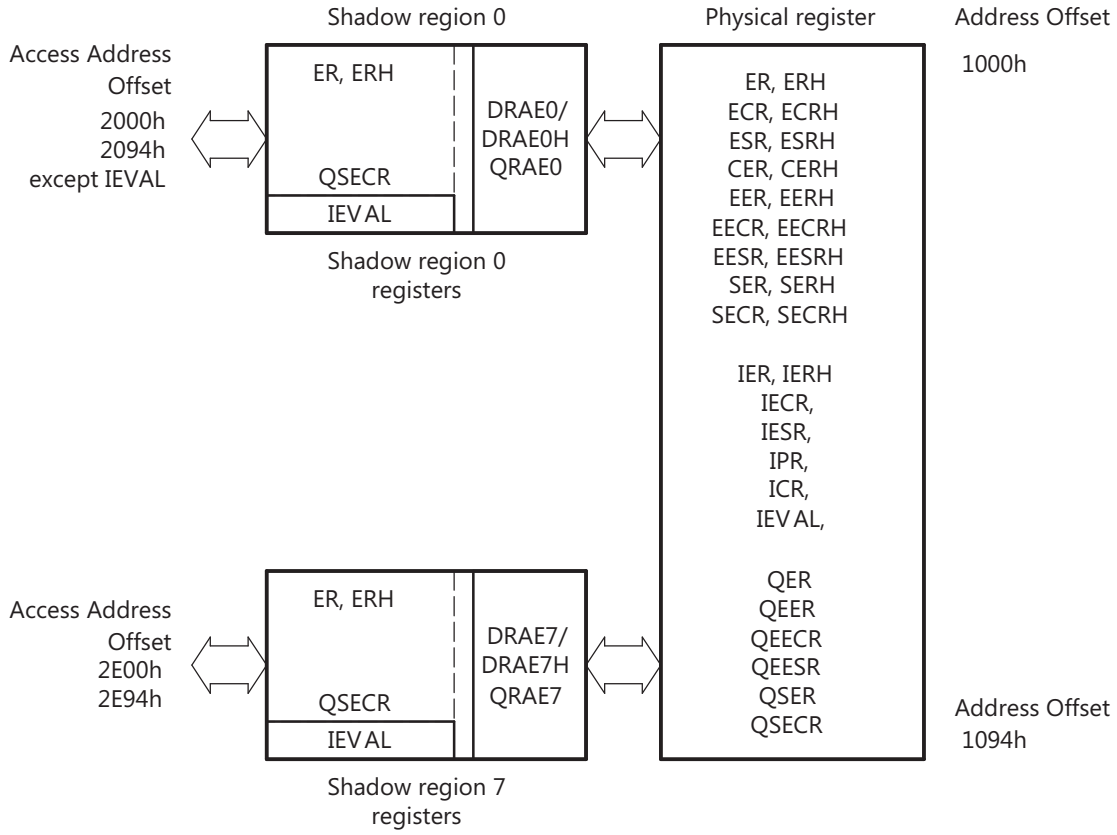
The channel registers (including DMA, QDMA, and interrupt registers) are accessible via the global channel region address range, or in the shadow *n* channel region address range(s). For example, the event enable register (EDMACC_EER) is visible in the global region register space at offset 1020h, or region addresses at offset 2020h for region 0, 2220h for region 1, ... offset 2E20h for region 7.

Figure 10-15 illustrates the conceptual view of the regions.

Table 10-15. Shadow Region Registers

DRAE *m*	DRAEH *m*	QRAE *n*
EDMACC_ER	EDMACC_ERH	EDMACC_QER
EDMACC_ECR	EDMACC_ECRH	EDMACC_QEER
EDMACC_ESR	EDMACC_ESRH	EDMACC_QEECR
EDMACC_CER	EDMACC_CERH	EDMACC_QEESR
EDMACC_EER	EDMACC_EERH	
EDMACC_EECR	EDMACC_EECRH	
EDMACC_EESR	EDMACC_EESRH	
EDMACC_SER	EDMACC_SERH	
EDMACC_SECR	EDMACC_SECRH	
EDMACC_IER	EDMACC_IERH	
EDMACC_IECR	EDMACC_IECRH	
EDMACC_IESR	EDMACC_IESRH	
EDMACC_IPR	EDMACC_IPRH	
EDMACC_ICR	EDMACC_ICRH	
Register not affected by DRAE/DRAEH		
EDMACC_IEVAL		

Figure 10-15. Shadow Region Registers



10.3.9.2 Channel Controller Shadow Regions

For each EDMA shadow region (and associated memory-maps) there is a set of registers associated with the shadow region that allows association of the DMA/QDMA channels and interrupt completion codes to the region. These registers are user-programmed per region to assign ownership of the DMA/QDMA channels and TCC values to a region.

- DRAE*m* and DRAEH*m*: One register pair exists for each of the shadow regions. The number of bits in each register pair matches the number of DMA channels. These registers need to be programmed to assign ownership of DMA channels and interrupt (or TCC codes) to the respective region. Accesses to DMA event registers and interrupt registers via the shadow region address map are filtered through the DRAE/DRAEH pair. A value of 1 in the corresponding DRAE(H) bit implies that the corresponding DMA/interrupt channel is accessible; a value of 0 in the corresponding DRAE(H) bit forces writes to be discarded and returns a value of 0 for reads.
- QRAE*n*: One register exists for every region. The number of bits in each register matches the number of QDMA channels. These registers must be programmed to assign ownership of QDMA channels to the respective region. To enable a channel in a shadow region using shadow region 0 EDMACC_QEER, the respective bit in QRAE must be set or writing into EDMACC_QEESR will not have the desired effect.
- MPPA*n* and EDMACC_MPPAG: One register exists for every region. This register defines the privilege level, requestor, and types of accesses allowed to a region's memory-mapped registers.

It is typical for an application to have a unique assignment of DMA/QDMA channels (and, therefore, a given bit position) to a given region.

The use of shadow regions allows for restricted access to EDMA resources (DMA channels, QDMA channels, TCC, interrupts) by tasks in a system by setting or clearing bits in the DRAE/QRAE registers. If exclusive access to any given channel/TCC code is required for a region, then only that region's DRAE/QRAE should have the associated bit set.

Example 10-1 illustrates a resource pool division across two regions, assuming region 0 must be allocated 16 DMA channels (0-15) and 1 QDMA channel (0) and 32 TCC codes (0-15 and 48-63). Region 1 needs to be allocated 16 DMA channels (16-32) and the remaining 3 QDMA channels (1-3) and TCC codes (16-47). DRAE should be equal to the OR of the bits that are required for the DMA channels and the TCC codes.

Example 10-1. Resource Pool Division Across Two Regions

```
Region 0:

DRAEH, DRAE = FFFF 0000h, 0000 FFFFh
QRAE = 000 0001h

Region 1:

DRAEH, DRAE = 0000 FFFFh, FFFF 0000h
QRAE = 000 000Eh
```

10.3.9.3 Region Interrupts

In addition to the EDMACC global completion interrupt (EDMACC_*m*_GINT, where *m* denotes the EDMACC instance number), there is an additional completion interrupt (EDMACC_*m*_REGION_*n*_INT, where *n* denotes the shadow region number) that is associated with every shadow region. The DRAE associated with each shadow region acts as a secondary interrupt enable along with the interrupt enable register (EDMACC_IER) for the respective shadow region interrupts. See Section 10.3.11 for more information on EDMA interrupts.

10.3.10 Chaining EDMA Channels

The channel chaining capability for the EDMA allows the completion of an EDMA channel transfer to trigger another EDMA channel transfer. The purpose is to allow the user the ability to chain several events through one event occurrence.

Chaining is different from linking (Section 10.3.5.7). The EDMA link feature reloads the current channel parameter set with the linked parameter set. The EDMA chaining feature does not modify or update any channel parameter set; it provides a synchronization event to the chained channel (see Section 10.3.6.1.3 for chain-triggered transfer requests).

Chaining is achieved at either final transfer completion or intermediate transfer completion, or both, of the current channel. Consider a channel *m* (DMA/QDMA) required to chain to channel *n*. Channel number *n* needs to be programmed into the TCC bit of channel *m* channel options parameter (OPT) set.

- If final transfer completion chaining (TCCHEN = 1 in OPT) is enabled, the chain-triggered event occurs after the *last* transfer request of channel *m* is either submitted (early completion) or completed (normal completion).
- If intermediate transfer completion chaining (ITCCHEN = 1 in OPT) is enabled, the chain-triggered event occurs after every *intermediate* transfer request of channel *m* is either submitted (early completion) or completed (normal completion).
- If both final and intermediate transfer completion chaining (TCCHEN = 1 and ITCCHEN = 1 in OPT) are enabled, then the chain-trigger event occurs after *every* transfer request of channel *m* is either submitted (early completion) or completed (normal completion).

Table 10-16 illustrates the number of chain event triggers occurring in different synchronized scenarios. Consider channel 31 programmed with ACNT = 3, BCNT = 4, CCNT = 5, and TCC = 30.

Table 10-16. Chain Event Triggers

Options	(Number of chained event triggers on channel 30)	
	A-Synchronized	AB-Synchronized
TCCHEN = 1, ITCCHEN = 0	1 (Owing to the last TR)	1 (Owing to the last TR)
TCCHEN = 0, ITCCHEN = 1	19 (Owing to all but the last TR)	4 (Owing to all but the last TR)
TCCHEN = 1, ITCCHEN = 1	20 (Owing to a total of 60 TRs)	5 (Owing to a total of 5 TRs)

10.3.11 EDMA Interrupts

The EDMA interrupts are divided into 2 categories:

- Transfer completion interrupts
- Error interrupts

The transfer completion interrupts are listed in Table 10-17. The error interrupts are listed in Table 10-18.

Table 10-17. EDMA Transfer Completion Interrupts

Name	Description
EDMACC_*m*_GINT	EDMACC Global Transfer Completion Interrupt
EDMACC_*m*_REGION_0_INT	EDMACC Transfer Completion Interrupt Shadow Region 0
EDMACC_*m*_REGION_1_INT	EDMACC Transfer Completion Interrupt Shadow Region 1
EDMACC_*m*_REGION_2_INT	EDMACC Transfer Completion Interrupt Shadow Region 2
EDMACC_*m*_REGION_3_INT	EDMACC Transfer Completion Interrupt Shadow Region 3
EDMACC_*m*_REGION_4_INT	EDMACC Transfer Completion Interrupt Shadow Region 4
EDMACC_*m*_REGION_5_INT	EDMACC Transfer Completion Interrupt Shadow Region 5
EDMACC_*m*_REGION_6_INT	EDMACC Transfer Completion Interrupt Shadow Region 6
EDMACC_*m*_REGION_7_INT	EDMACC Transfer Completion Interrupt Shadow Region 7
EDMACC_*m*_TC_AET_INT	EDMACC Advanced Event Triggering (AET) Event

Table 10-18. EDMA Error Interrupts

Name	Description
EDMACC_*m*_ERRINT	EDMACC Error Interrupt
EDMACC_*m*_MPINT	EDMACC Memory Protection Interrupt
EDMACC_*m*_TC_*n*_ERRINT	EDMATC Error Interrupt

10.3.11.1 Transfer Completion Interrupts

The EDMACC is responsible for generating transfer completion interrupts to the CPU. The EDMA generates a single completion interrupt per shadow region, as well as one for the global region on behalf of all DMA/QDMA channels. The various control registers and bit fields facilitate EDMA interrupt generation.

For a given DMA/QDMA channel, the software architecture should either use the global interrupt or the shadow interrupts, but not both.

The transfer completion code (TCC) value is directly mapped to the bits of the interrupt pending register (EDMACC_IPR/EDMACC_IPRH). For example, if TCC = 10 0001b, EDMACC_IPRH[1] is set after transfer completion, and results in interrupt generation to the CPU(s) if the completion interrupt is enabled for the CPU. See Section 10.3.11.1.1 for details on enabling EDMA transfer completion interrupts.

When a completion code is returned (as a result of early or normal completion), the corresponding bit in EDMACC_IPR/EDMACC_IPRH is set if transfer completion interrupt (final/intermediate) is enabled in the channel options parameter (OPT) for a PaRAM set associated with the transfer.

The transfer completion code (TCC) can be programmed to any value for a DMA/QDMA channel. A direct relation between the channel number and the transfer completion code value does not need to exist. This allows multiple channels having the same transfer completion code value to cause a CPU to execute the same interrupt service routine (ISR) for different channels.

NOTE: The TCC field in the channel options parameter (OPT) is a 6-bit field and can be programmed for any value between 0-64. For devices with 16/32 DMA channels, the TCC should have a value between 0 to 15/31 so that it sets the appropriate bits (0 to 15/31) in EDMACC_IPR (and can interrupt the CPU(s) on enabling the EDMACC_IER register bits (0-15/31)).

If the channel is used in the context of a shadow region and the user intends for the shadow region interrupt to be asserted, then ensure that the bit corresponding to the TCC code is enabled in EDMACC_IER/EDMACC_IERH and in the corresponding shadow region's DMA region access registers (DRAE/DRAEH).

The user can enable Interrupt generation at either final transfer completion or intermediate transfer completion, or both. Consider channel *m* as an example.

- If the final transfer interrupt (TCINTEN = 1 in OPT) is enabled, the interrupt occurs after the *last* transfer request of channel *m* is either submitted or completed (depending on early or normal completion).
- If the intermediate transfer interrupt (ITCINTEN = 1 in OPT) is enabled, the interrupt occurs after every *intermediate* transfer request of channel *m* is either submitted or completed (depending on early or normal completion).
- If both final and intermediate transfer completion interrupts (TCINTEN = 1, and ITCINTEN = 1 in OPT) are enabled, then the interrupt occurs after *every* transfer request of channel *m* is submitted or completed (depending on early or normal completion).

Table 10-19 shows the number of interrupts that occur in different synchronized scenarios. Consider channel 31, programmed with ACNT = 3, BCNT = 4, CCNT = 5, and TCC = 30.

Table 10-19. Number of Interrupts

Options	A-Synchronized	AB-Synchronized
TCINTEN = 1, ITCINTEN = 0	1 (Last TR)	1 (Last TR)
TCINTEN = 0, ITCINTEN = 1	19 (All but the last TR)	4 (All but the last TR)
TCINTEN = 1, ITCINTEN = 1	20 (All TRs)	5 (All TRs)

10.3.11.1.1 Enabling Transfer Completion Interrupts

For the EDMA channel controller to assert a transfer completion to the external world, the interrupts must be enabled in the EDMACC. This is in addition to setting up the TCINTEN and ITCINTEN bits in OPT of the associated PaRAM set.

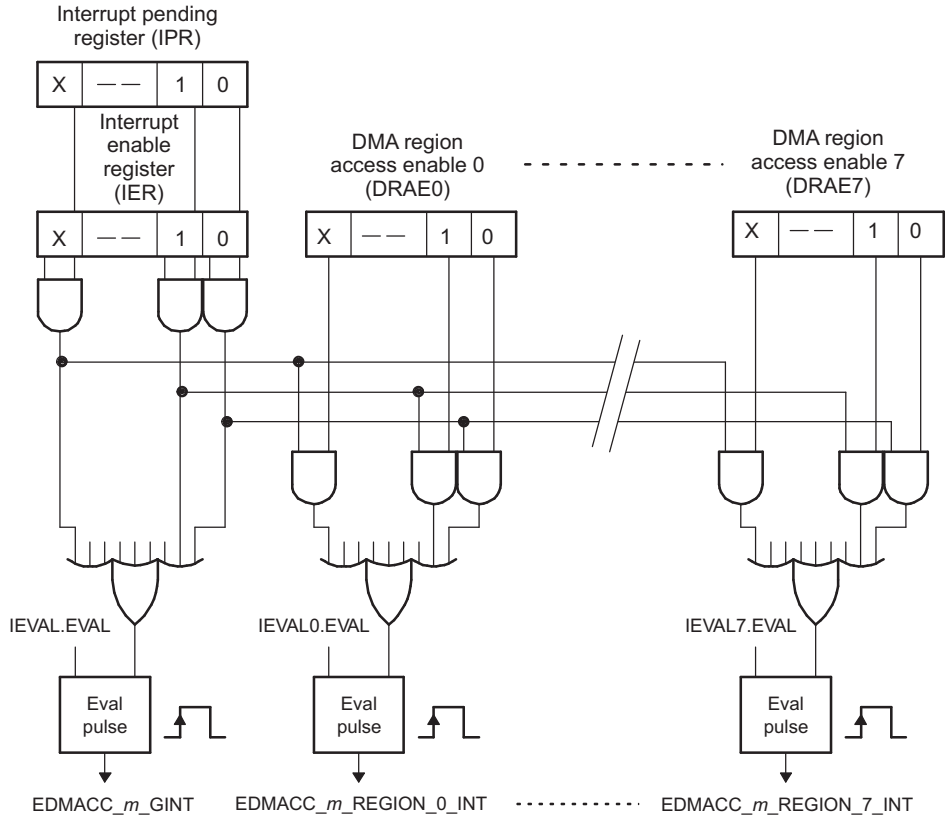
The EDMA channel controller has interrupt enable registers (EDMACC_IER/EDMACC_IERH) and each bit location in EDMACC_IER/EDMACC_IERH serves as a primary enable for the corresponding interrupt pending registers (EDMACC_IPR/EDMACC_IPRH).

All the interrupt registers (EDMACC_IER, EDMACC_IESR, EDMACC_IECR, and EDMACC_IPR) are either manipulated from the global DMA channel region, or by way of the DMA channel shadow regions. The shadow regions provide a view to the same set of physical registers that are in the global region.

The EDMA channel controller has a hierarchical completion interrupt scheme that uses a single set of interrupt pending registers (EDMACC_IPR/EDMACC_IPRH) and single set of interrupt enable registers (EDMACC_IER/EDMACC_IERH). The programmable DMA region access enable registers (DRAE/DRAEH) provides a second level of interrupt masking. The global region interrupt output is gated based on the enable mask that is provided by EDMACC_IER.

The region interrupt outputs are gated by EDMACC_IER and the specific DRAE/DRAEH associated with the region. See Figure 10-16.

Figure 10-16. Interrupt Diagram



For the EDMACC to generate the transfer completion interrupts that are associated with each shadow region, the following conditions must be true:

- EDMACC_m_REGION_0_INT: (EDMACC_IPR.E0 & EDMACC_IER.E0 & EDMACC_DRAE0.E0) | (EDMACC_IPR.E1 & EDMACC_IER.E1 & EDMACC_DRAE0.E1) | …|(EDMACC_IPR[H].E *n* & EDMACC_IER[H].E *n* & DRAE[H]0.E *n*)
- EDMACC_m_REGION_1_INT: (EDMACC_IPR.E0 & EDMACC_IER.E0 & EDMACC_DRAE1.E0) | (EDMACC_IPR.E1 & EDMACC_IER.E1 & EDMACC_DRAE1.E1) | …| (EDMACC_IPR[H].E *n* & EDMACC_IER[H].E *n* & DRAE[H]1.E *n*)
- EDMACC_m_REGION_2_INT: (EDMACC_IPR.E0 & EDMACC_IER.E0 & EDMACC_DRAE2.E0) | (EDMACC_IPR.E1 & EDMACC_IER.E1 & EDMACC_DRAE2.E1) | …|(EDMACC_IPR[H].E *n* & EDMACC_IER[H].E *n* & DRAE[H]2.E *n*)...
- Up to EDMACC_m_REGION_7_INT: (EDMACC_IPR.E0 & EDMACC_IER.E0 & EDMACC_DRAE7.E0) | (EDMACC_IPR.E1 & EDMACC_IER.E1 & EDMACC_DRAE7.E1) | …|(EDMACC_IPR[H].E *n* & EDMACC_IER[H].E *n* & DRAE[H]7.E *n*)

NOTE: The DRAE/DRAEH for all regions are expected to be set up at system initialization and to remain static for an extended period of time. The interrupt enable registers should be used for dynamic enable/disable of individual interrupts.
Because there is no relation between the TCC value and the DMA/QDMA channel, it is possible, for example, for DMA channel 0 to have the OPT.TCC = 63 in its associated PaRAM set. This would mean that if a transfer completion interrupt is enabled (OPT.TCINTEN or OPT.ITCINTEN is set), then based on the TCC value, EDMACC_IPRH.E63 is set up on completion. For proper channel operations and interrupt generation using the shadow region map, the user must program the DRAE/DRAEH that is associated with the shadow region to have read/write access to both bit 0 (corresponding to channel 0) and bit 63 (corresponding to EDMACC_IPRH bit that is set upon completion).

10.3.11.1.2 Clearing Transfer Completion Interrupts

Transfer completion interrupts that are latched to the interrupt pending registers (EDMACC_IPR/EDMACC_IPRH) are cleared by writing a 1 to the corresponding bit in the interrupt pending clear register (EDMACC_ICR/EDMACC_ICRH). For example, a write of 1 to EDMACC_ICR.E0 clears a pending interrupt in EDMACC_IPR.E0.

If an incoming transfer completion code (TCC) gets latched to a bit in EDMACC_IPR/EDMACC_IPRH, then additional bits that get set due to a subsequent transfer completion will not result in asserting the EDMACC completion interrupt. In order for the completion interrupt to be pulsed, the required transition is from a state where no enabled interrupts are set to a state where at least one enabled interrupt is set.

10.3.11.2 EDMA Interrupt Servicing

Upon completion of a transfer (early or normal completion), the EDMA channel controller sets the appropriate bit in the interrupt pending registers (EDMACC_IPR/EDMACC_IPRH) as specified by the transfer completion codes. If the completion interrupts are appropriately enabled, then the CPU enters the interrupt service routine (ISR) when the completion interrupt is asserted.

After servicing the interrupt, the ISR should clear the corresponding bit in EDMACC_IPR/EDMACC_IPRH, thereby enabling recognition of future interrupts. The EDMACC will only assert additional completion interrupts when all EDMACC_IPR/EDMACC_IPRH bits are cleared.

When one interrupt is serviced many other transfer completions may result in additional bits being set in EDMACC_IPR/EDMACC_IPRH, thereby resulting in additional interrupts. Each of the bits in EDMACC_IPR/EDMACC_IPRH may need different types of service; therefore, the ISR may check all pending interrupts and continue until all of the posted interrupts are serviced appropriately.

Examples of pseudo code for a CPU interrupt service routine for an EDMACC completion interrupt are shown in Section 10.3.11.2.1 and Section 10.3.11.2.2.

The ISR routine in Section 10.3.11.2.1 is more exhaustive and incurs a higher latency.

10.3.11.2.1 Interrupt Servicing Example 1

Step 1. Read the interrupt pending register (EDMACC_IPR/EDMACC_IPRH).

Step 2. Perform the operations needed.

Step 3. Writes to the interrupt pending clear register (EDMACC_ICR/EDMACC_ICRH) to clear the corresponding EDMACC_IPR/EDMACC_IPRH bit(s).

Step 4. Reads EDMACC_IPR/EDMACC_IPRH again:

a. If EDMACC_IPR/EDMACC_IPRH is not equal to 0, repeat from step 2 (implies occurrence of new event between step 2 to step 4).

b. If EDMACC_IPR/EDMACC_IPRH is equal to 0, this should assure the user that all enabled interrupts are inactive.

NOTE: An event may occur during step 4 while the EDMACC_IPR/EDMACC_IPRH bits are read as 0 and the application is still in the interrupt service routine. If this happens, a new interrupt is recorded in the device interrupt controller and a new interrupt is generated as soon as the application exits the interrupt service routine.

Section 10.3.11.2.2 is less rigorous, with less burden on the software in polling for set interrupt bits, but can occasionally cause a race condition as mentioned above.

If it is desired to leave any enabled and pending (possibly lower priority) interrupts, it is required to force the interrupt logic to reassert the interrupt pulse by setting the EVAL bit in the interrupt evaluation register (EDMACC_IEVAL).

10.3.11.2.2 Interrupt Servicing Example 2

Step 1. Enter ISR.

Step 2. Read EDMACC_IPR/EDMACC_IPRH.

Step 3. For the condition set in EDMACC_IPR/EDMACC_IPRH that the user desires to service, do the following:

a. Service interrupt as required by the application.

b. Clear the bit for serviced conditions (others may still be set, and other transfers may have resulted in returning the TCC to EDMACC after step 2).

Step 4. Read EDMACC_IPR/EDMACC_IPRH prior to exiting the ISR:

a. If EDMACC_IPR/EDMACC_IPRH is equal to 0, then exit the ISR.

b. If EDMACC_IPR/EDMACC_IPRH is not equal to 0, then set EDMACC_IEVAL so that upon exit of ISR, a new interrupt is triggered if any enabled interrupts are still pending.

NOTE: The EVAL bit must not be set when EDMACC_IPR/EDMACC_IPRH is read to be 0, to avoid generation of extra interrupt pulses.

10.3.11.3 Interrupt Evaluation Operations

The EDMACC has interrupt evaluate registers (EDMACC_IEVAL) that exist in the global region and in each shadow region. The registers in the shadow region are the only registers in the DMA channel shadow region memory map that are not affected by the settings for the DMA region access enable registers (DRAE/DRAEH). Writing 1 to the EVAL bit in the registers that are associated with a particular shadow region results in pulsing the associated region interrupt (global or shadow), if any enabled interrupt (via EDMACC_IER/EDMACC_IERH) is still pending (EDMACC_IPR/EDMACC_IPRH). This register assures that the CPU does not miss the interrupts (or the EDMA master associated with the shadow region) if the software architecture chooses not to use all interrupts. See Section 10.3.11.2.2 for the use of EDMACC_IEVAL in the EDMA interrupt service routine (ISR).

Similarly an error evaluation register (EDMACC_EEVAL) exists in the global region. Writing 1 to the EVAL bit in EDMACC_EEVAL causes the pulsing of the error interrupt if any pending errors are in EDMACC_EMR/EDMACC_EMRH, EDMACC_QEMR, or EDMACC_CCERR. See Section 10.3.11.4 for additional information regarding error interrupts.

NOTE: While using EDMACC_IEVAL for shadow region completion interrupts, the user should make sure that the EVAL operated upon is from that particular shadow region memory map.

10.3.11.4 Error Interrupts

The EDMACC error registers provide the capability to differentiate error conditions (event missed, threshold exceed, etc.). Additionally, setting the error bits in these registers results in asserting the EDMACC error interrupt. If the EDMACC error interrupt is enabled in the device interrupt controller(s), then it allows the CPU(s) to handle the error conditions.

The EDMACC has a single error interrupt (EDMA_CC_*m*_ERRINT) that is asserted for all EDMACC error conditions. There are four conditions that cause the error interrupt to be pulsed:

- DMA missed events: for all DMA channels. DMA missed events are latched in the event missed registers (EDMACC_EMR/EDMACC_EMRH).
- QDMA missed events: for all QDMA channels. QDMA missed events are latched in the QDMA event missed register (EDMACC_QEMR).
- Threshold exceed: for all event queues. These are latched in EDMACC error register (EDMACC_CCERR).
- TCC error: for outstanding transfer requests that are expected to return completion code (TCCHEN or TCINTEN bit in OPT is set to 1) exceeding the maximum limit of *n*-1, where *n* is the number of DMA channels supported by the EDMACC. This is also latched in the EDMACC error register (EDMACC_CCERR).

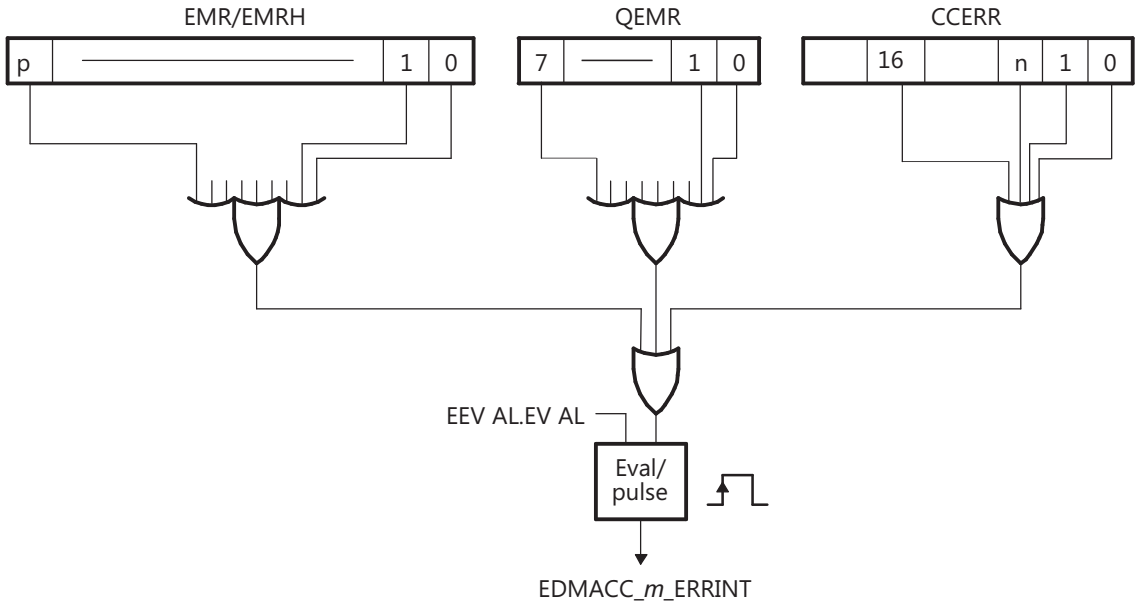
Figure 10-17 illustrates the EDMACC error interrupt generation operation.

If any of the bits are set in the error registers due to any error condition, the EDMA_CC *m*_ERRINT is always asserted, as there are no enables for masking these error events. Similar to the transfer completion interrupts, the error interrupt also is pulsed only when the error interrupt condition transitions from a state where no errors are set to a state where at least one error bit is set. If additional error events are latched prior to the original error bits being cleared, the EDMACC does not generate additional interrupt pulses.

To reduce the burden on the software, similar to the interrupt evaluate register (EDMACC_IEVAL), there is an error evaluate register (EDMACC_EEVAL) that allows re-evaluation of pending set error events/bits. This can be used so that the CPU(s) does not miss any error events.

NOTE: It is a good practice to enable the error interrupt in the device interrupt controller and to associate an interrupt service routine with it to address the various error conditions appropriately. Doing so puts less burden on the software (polling for error status). Additionally, it provides a good debug mechanism for unexpected error conditions.

Figure 10-17. Error Interrupt Operation



10.3.12 Memory Protection

The EDMA channel controller supports two kinds of memory protection: active and proxy.

10.3.12.1 Active Memory Protection

Active memory protection is a feature that allows or prevents read and write accesses (by any EDMA programmer) to the EDMACC registers (based on user-programmed permission characteristics). Active memory protection is achieved by a set of memory protection permissions attribute (MPPA) registers.

The EDMACC register map is divided into three categories:

- A global region
- A global channel region
- 8 shadow regions

Each shadow region consists of the respective shadow region registers and the associated PaRAM. For more detailed information regarding the contents of a shadow region, see Table 10-15.

Each of the eight shadow regions has an associated MPPA register (MPPA *n*) that defines the specific requestor(s) and types of requests that are allowed to the regions resources.

The global channel region is also protected with a memory-mapped register (EDMACC_MPPAG). The EDMACC_MPPAG applies to the global region and to the global channel region, except the other MPPA registers themselves. For more detailed information on the list of the registers in each region, see Table 10-21.

See for the bit field descriptions of MPPA *n*. The MPPA *n* have a certain set of access rules.

Table 10-20 shows the accesses that are allowed or not allowed to the EDMACC_MPPAG and MPPA *n*.

Table 10-20. Allowed Accesses

Access	Supervisor	User
Read	Yes	Yes
Write	Yes	No

Table 10-21 describes the MPPA register mapping for the shadow regions (which includes shadow region registers and PaRAM addresses).

The region-based MPPA registers are used to protect accesses to the DMA shadow regions and the associated region PaRAM. Because there are eight regions, there are eight MPPA region registers (MPPA[0-7]).

Table 10-21. MPPA Registers to Region Assignment

Register	Registers Protect	Address Range	PaRAM Protect	Address Range for 512 PaRAM sets
EDMACC_MPPAG	Global Range	0000h-1FFCh	N/A	N/A
EDMACC_MPPA0	DMA Shadow 0	2000h-21FCh	1st octant	4000h-47FCh
MPPA1	DMA Shadow 1	2200h-23FCh	2nd octant	4800h-4FFCh
MPPA2	DMA Shadow 2	2400h-25FCh	3rd octant	5000h-57FCh
MPPA3	DMA Shadow 3	2600h-27FCh	4th octant	5800h-5FFCh
MPPA4	DMA Shadow 4	2800h-29FCh	5th octant	6000h-67FCh
MPPA5	DMA Shadow 5	2A00h-2BFCh	6th octant	6800h-6FFCh
MPPA6	DMA Shadow 6	2C00h-2DFCh	7th octant	7000h-77FCh
EDMACC_MPPA7	DMA Shadow 7	2E00h-2FFCh	8th octant	7800h-7FFCh

Section 10.3.12.1.1 is an example of access denied to an EDMACC register. Write access to shadow region 7's event enable set register (EDMACC_EESR).

10.3.12.1.1 Access Denied to an EDMACC Register

Step 1. The original value of the event enable register (EDMACC_EER) at address offset 1020h is 0.

Step 2. The MPPA[7] is set to prevent user level accesses (UW = 0, UR = 0), but it allows supervisor level accesses (SW = 1, SR = 1) with a privilege ID of 0. (AID0 = 1).

Step 3. The C66x DSP, which has a privilege ID of 0, attempts to perform a user-level write access of a value of FF00 FF00h to shadow region 7's event enable set register (EDMACC_EESR) at address offset 2E30h. Note that the EDMACC_EER is a read-only register and the only way that the user can write to it is by writing to the EDMACC_EESR. Also remember that there is only one physical register for EDMACC_EER, EDMACC_EESR, etc. and that the shadow regions only provide to the same physical set.

Step 4. Since the MPPA[7] has UW = 0, though the privilege ID of the write access is set to 0, the access is not allowed and the EDMACC_EER is not written to.

Table 10-22 illustrates the example above.

Table 10-22. Example Access Denied

Register (Address Offset)	Value	Description
EDMACC_EER (1020h)	0000 0000h	Initial value in EDMACC_EER.
EDMACC_EESR (2E30h)	FF00 FF00h	Value attempted to be written to shadow region 7's EDMACC_EESR. This is done by an EDMA programmer with a privilege level of User and Privilege ID of 0.
MPPA[7] (082Ch)	0000 04B0h	Memory Protection Filter AID0 = 1, UW = 0, UR = 0, SW = 1, SR = 1.
	X	Access Denied
EDMACC_EER (1020h)	0000 0000h	Final value of EDMACC_EER

Section 10.3.12.1.2 is an example of access allowed to an EDMACC register.

10.3.12.1.2 Write access to shadow region 7's event enable set register (EDMACC_EESR)

Step 1. The original value of the event enable register (EDMACC_EER) at address offset 1020h is 0

Step 2. The MPPA[7] is set to allow user-level accesses (UW = 1, UR = 1) and supervisor-level accesses (SW = 1, SR = 1) with a privilege ID of 0. (AID0 = 1)

Step 3. The C66x DSP, which has a privilege ID of 0, attempts to perform a user-level write of a value of ABCD 0123h to shadow region 7's event enable set register (EDMACC_EESR) at address offset 2E30h. Note that the EDMACC_EER is a read-only register and the only way that the user can write to it is by writing to the EDMACC_EESR. Also remember that there is only one physical register for EDMACC_EER, EDMACC_EESR, etc. and that the shadow regions only provide to the same physical set.

Step 4. Since the MPPA[7] has UW = 1 and AID0 = 1, the user-level write access is allowed.

Step 5. Remember that accesses to shadow region registers are masked by their respective DRAE register. In this example, the DRAE[7] is set of 9F00 0F00h.

Step 6. The value finally written to EDMACC_EER is 9F00 0F00h.

Table 10-23 illustrates the example above.

Table 10-23. Example Access Allowed

Register (Address Offset)	Value	Description
EDMACC_EER (1020h)	0000 0000h	Initial value in EDMACC_EER.
EDMACC_EESR (2E30h)	FF00 FF00h	Value attempted to be written to shadow region 7's EDMACC_EESR. This is done by an EDMA programmer with a privilege level of User and Privilege ID of 0.
MPPA[7] (082Ch)	0000 04B3h	Memory Protection Filter AID0 = 1, UW = 1, UR = 1, SW = 1, SR = 1.
	3	Access allowed.
	↓	
DRAE[7] (0378h)	9FF0 0FC2h	DMA Region Access Enable Filter
	↓	
EDMACC_EESR (2E30h)	9F00 0F00h	Value written to shadow region 7's EDMACC_EESR. This is done by an EDMA programmer with a privilege level of User and a Privilege ID of 0.
	↓	
EDMACC_EER (1020h)	↓	Final value of EDMACC_EER.
	9F00 0F00h	

10.3.12.2 Proxy Memory Protection

Proxy memory protection allows an EDMA transfer programmed by a given EDMA programmer to have its permissions travel with the transfer through the EDMATC. The permissions travel along with the read transactions to the source and the write transactions to the destination endpoints. The PRIV bit and PRIVID bit in the channel options parameter (OPT) is set with the EDMA programmer's PRIV value and PRIVID values, respectively, when any part of the PaRAM set is written.

The PRIV is the privilege level (that is, user versus supervisor). The PRIVID refers to a privilege ID with a number that is associated with an EDMA programmer.

See the device Data Manual for the PRIVIDs that are associated with potential EDMA programmers.

These options are part of the TR that are submitted to the transfer controller. The transfer controller uses the above values on their respective read and write command bus so that the target endpoints can perform memory protection checks based on these values.

For example, consider a parameter set that is programmed by a DSP in user privilege level for a simple transfer with the source buffer on an L2 page and the destination buffer on an external memory page. The PRIV is 0 for user-level and the DSP has a PRIVID of 0.

The PaRAM set is shown in Figure 10-22.

Figure 10-18. PaRAM Set Content for Proxied Memory Protection Example

(a) EDMA Parameters

Parameter Contents	Parameter
0010 0007h	Channel Options Parameter (OPT)
1080 0000h	Channel Source Address (SRC)
0001h | 0004h	Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT)
8000 0000h	Channel Destination Address (DST)
0001h | 0001h	Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX)
0000h | FFFFh	BCNT Reload (BCNTRLD) | Link Address (LINK)
0001h | 1000h	Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX)
0000h | 0001h	Reserved | Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	0	1	1	1
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

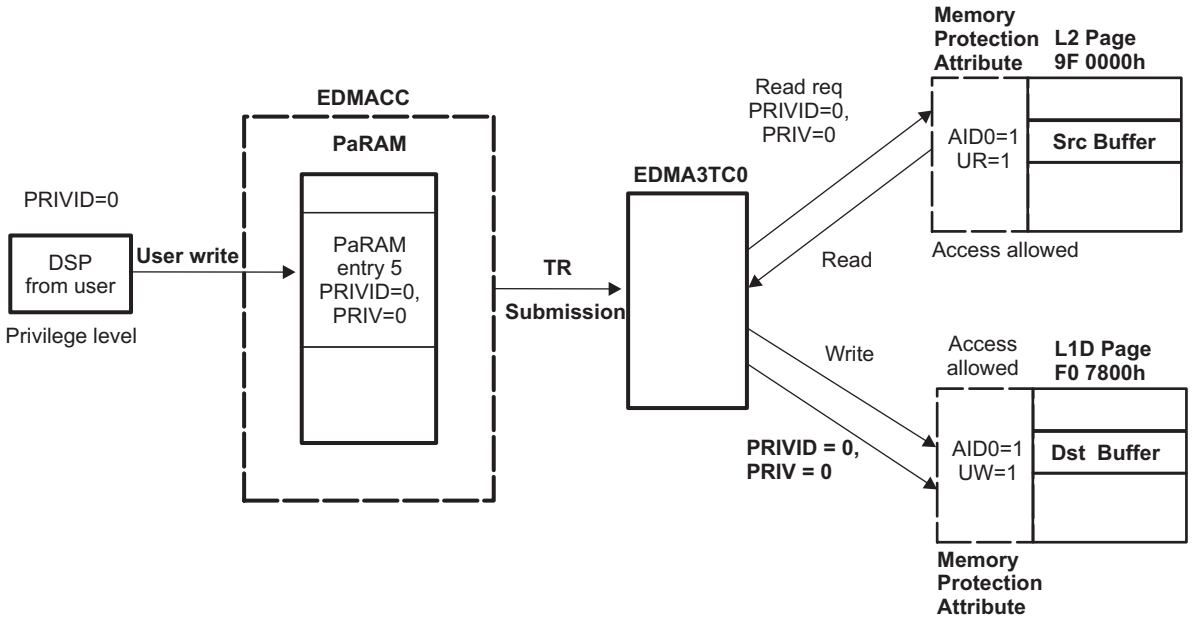
The PRIV and PRIVID information travels along with the read and write requests that are issued to the source and destination memories.

For example, if the access attributes that are associated with the L2 page with the source buffer only allow supervisor read, write accesses (SR, SW), the user-level read request above is refused. Similarly, if the access attributes that are associated with the L1D page with the destination buffer only allow supervisor read and write accesses (SR, SW), the user-level write request above is refused. For the transfer to succeed, the source and destination pages should have user-read and user-write permissions, respectively, along with allowing accesses with a PRIVID 0. For more information regarding how to set memory protection attributes for pages of memory in L2/L1D, see the TMS320C66x DSP CorePac User Guide (SPRUGW0).

Because the programmers privilege level and privilege identification travel with the read and write requests, EDMA acts as a proxy.

Figure 10-19 illustrates the propagation of PRIV and PRIVID at the boundaries (DSP, EDMACC, EDMATC, and slave memories).

Figure 10-19. Proxied Memory Protection Example



10.3.13 Event Queue(s)

Event queues are a part of the EDMA channel controller. Event queues form the interface between the event detection logic in the EDMACC and the transfer request (TR) submission logic of the EDMACC. Each queue is 16 entries deep; thus, each event queue can queue a maximum of 16 events. If there are more than 16 events, then the events that cannot find a place in the event queue remain set in the associated event register and the CPU does not stall.

There are two event queues for the device: Q0, and Q1. By default, there is a one-to-one mapping between the queues and transfer controllers. Therefore, the transfer requests (TRs) associated with events in Q0 get submitted to EDMATC0. Similarly, transfer requests associated with events in Q1 get submitted to EDMATC1.

An event that wins prioritization against other DMA and/or QDMA pending events is placed at the tail of the appropriate event queue. Each event queue is serviced in FIFO order. Once the event reaches the head of its queue and the corresponding transfer controller is ready to receive another TR, the event is de-queued and the PaRAM set corresponding to the de-queued event is processed and submitted as a transfer request packet (TRP) to the associated EDMA transfer controller.

A lower numbered queue has a higher de-queuing priority than a higher numbered queue. For example, Q0 has higher priority than Q1, if Q0 and Q1 both have at least one event entry and if both EDMATC0 and EDMATC1 can accept transfer requests, then the event in Q0 is de-queued first and its associated PaRAM set is processed and submitted as a transfer request (TR) to TC0.

See Section 10.3.13.4 for system-level performance considerations. All the event entries in all the event queues are software readable (not writable) by accessing the event queue entry registers (Q *x*E *y*). Each event entry register characterizes the queued event in terms of the type of event (manual, event, chained or auto-triggered) and the event number. See for a description of the bit fields in the queue event entry registers.

10.3.13.1 DMA/QDMA Channel to Event Queue Mapping

Each DMA channel and QDMA channel is independently programmed to map to a specific queue using the DMA queue number register (DMAQNUM *n*) and the QDMA queue number register (QDMANUM). The mapping of DMA/QDMA channels is critical to achieving the desired performance level for the EDMA and most importantly in meeting real-time deadlines. See Section 10.3.13.4.

NOTE: If an event is ready to be queued and both the event queue and the EDMA transfer controller associated to the event queue are empty, then the event bypasses the event queue, and goes to the PaRAM processing logic and eventually to the transfer request submission logic for submission to the EDMATC. In this case, the event is not logged in the event queue status registers.

10.3.13.2 Queue RAM Debug Visibility

Each event queue has 16 entries. These 16 entries are managed in a circular FIFO manner. All event queue entries for all event queues are software readable by the event queue entry register (Q *x*E *y*). Additionally, for each queue there is a queue status register (QSTAT *n*).

These registers provide user visibility and may be helpful while debugging real-time issues (typically post-mortem), involving multiple events and event sources.

Each of the 16 entries in the event queue are read using the EDMACC memory-mapped register. By reading the event queue, the user can see the history of the last 16 TRs that have been processed by the EDMA controller on a given queue. This provides user/software visibility and is helpful for debugging real-time issues (typically post-mortem), involving multiple events and event sources.

The event queue entry register (Q *x*E *y*) uniquely identifies the specific event type (event-triggered, manually-triggered, chain-triggered, and QDMA events) along with the event number (for DMA/QDMA channels) that are in the queue or have been de-queued (passed through the queue).

The queue status register (QSTAT *n*) includes fields for the start pointer (STRTPTR) that provides the offset to the head entry of an event. It also includes a NUMVAL field that provides the total number of valid entries residing in the event queue at a given instance of time. The STRTPTR field may be used to index appropriately into the 16 event entries. The NUMVAL number of entries starting from STRTPTR are indicative of events still queued in the respective queue. The remaining entries may be read to determine which events have already been de-queued and submitted to the associated transfer controller.

10.3.13.3 Queue Resource Tracking

The EDMACC event queue includes watermarking/threshold logic that allows the user to keep track of maximum usage of all event queues. This is useful for debugging real-time deadline violations that may result from head-of-line blocking on a given EDMA event queue.

The user can program the maximum number of events that can queue up in an event queue by programming the threshold value (between 0 to 15) in the queue watermark threshold A register (EDMACC_QWMTHRA). The maximum queue usage is recorded actively in the watermark (WM) field of the queue status register (QSTAT *n*) that keeps getting updated based on a comparison of number of valid entries, which is also visible in the NUMVAL bit in QSTAT *n* and the maximum number of entries (WM bit in QSTAT *n*).

If the queue usage is exceeded, this status is visible in the EDMACC registers: the QTHRXCD *n* bit in the channel controller error register (EDMACC_CCERR) and the THRXCD bit in QSTAT *n*, where *n* stands for the event queue number. Any bits that are set in EDMACC_CCERR also generate an EDMACC error interrupt.

10.3.13.4 Performance Considerations

The main TeraNet switch fabric arbitrates bus requests from all of the masters (like DSP core, Arm Cortex-A15, etc.) to the shared slave resources (peripherals and memories).

The priorities of transfer requests (read and write commands) from the EDMA transfer controllers with respect to other masters within the system crossbar are programmed using the queue priority register (EDMACC_QUEPRI). EDMACC_QUEPRI programs the priority of the event queues (or indirectly, TC *n*, because Q *n* transfer requests are submitted to TC *n*).

Therefore, the priority of unloading queues has a secondary affect compared to the priority of the transfers as they are executed by the EDMATC (dictated by the priority set using EDMACC_QUEPRI).

10.3.14 EDMA Transfer Controller Operation

The EDMA channel controller (EDMACC) is the user-interface of the EDMA module and the EDMA transfer controller (EDMATC) is the data movement engine of the EDMA module. The EDMACC submits transfer requests (TR) to the EDMATC and the EDMATC performs the data transfers dictated by the TR; thus, the EDMATC is a slave to the EDMACC.

10.3.14.1 Architecture Details

10.3.14.1.1 EDMATC Configuration

The parameters that determine the EDMATC configuration are:

- FIFOSIZE: Determines the size in bytes for the Data FIFO that is the temporary buffer for the in-flight data. The data FIFO is where the read return data read by the EDMATC read controller from the source endpoint is stored and subsequently written out to the destination endpoint by the EDMATC write controller.
- BUSBYTE: The width of the read and write data buses (in bytes) for the EDMATC read and write controller, respectively.
- Default Burst Size (DBS): The DBS is the maximum number of bytes per read/write command issued by a transfer controller.
- DSTREGDEPTH: This determines the number of Destination FIFO register set. The number of Destination FIFO register set for a transfer controller, determines the maximum number of outstanding transfer requests (TR pipelining).

All the above four parameters are fixed in design for the device (see section Section 10.1.3).

10.3.14.1.2 Command Fragmentation

The EDMATC read and write controllers in conjunction with the source and destination register sets are responsible for issuing optimally-sized reads and writes to the slave endpoints. An optimally-sized command is defined by the transfer controller default burst size (DBS), which is tied-off to be 64 bytes for each EDMATC instance present on this device.

The EDMATC attempts to issue the largest possible command size as limited by the DBS value or the ACNT/BCNT value of the TR. EDMATC obeys the following rules:

- The read/write controllers always issue commands less than or equal to the DBS tie-off value.
- The first command of a 1D transfer is always issued so that subsequent commands align to the DBS tie-off value.

Example 10-2 shows the command fragmentation for a DBS of 64 bytes. In summary, if the ACNT value is larger than the DBS value, then the EDMATC breaks the ACNT array into DBS-sized commands to the source/destination addresses. Each BCNT number of arrays are then serviced in succession.

For BCNT arrays of ACNT bytes (that is, a 2D transfer), if the ACNT value is less than or equal to the DBS value, then the TR may be optimized into a 1D-transfer in order to maximize efficiency. The optimization takes place if the EDMATC recognizes that the 2D-transfer is organized as a single dimension (ACNT == BIDX) and the ACNT value is a power of 2. Table 10-24 summarizes conditions in which the optimizations are performed.

Example 10-2. Command Fragmentation (DBS = 64)

The pseudo code:

1. ACNT = 8, BCNT = 8, SRCBIDX = 8, DSTBIDX = 10, SRCADDR = 64, DSTADDR = 191

   Read Controller: This is optimized from a 2D-transfer to a 1D-transfer such that the read side is equivalent to ACNT = 64, BCNT = 1. Cmd0 = 64 byte
   Write Controller: Because DSTBIDX != ACNT, it is not optimized.
   Cmd0 = 8 byte, Cmd1 = 8 byte, Cmd2 = 8 byte, Cmd3 = 8 byte, Cmd4 = 8 byte, Cmd5 = 8 byte, Cmd6 = 8 byte, Cmd7 = 8 byte.

2. ACNT=128, BCNT = 1,SRCADDR = 63, DSTADDR = 513

   Read Controller: Read address is not aligned.
   Cmd0 = 1 byte, (now the SRCADDR is aligned to 64 for the next command) Cmd1 = 64 bytes Cmd2 = 63 bytes
   Write Controller: The write address is also not aligned.
   Cmd0 = 63 bytes, (now the DSTADDR is aligned to 64 for the next command) Cmd1 = 64 bytes Cmd2 = 1 byte

Table 10-24. Read/Write Command Optimization Rules

ACNT DBS	ACNT is power of 2	BIDX = ACNT	BCNT 1023	SAM/DAM = Increment	Description
Yes	Yes	Yes	Yes	Yes	Optimized
No	x	x	x	x	Not Optimized
x	No	x	x	x	Not Optimized
x	x	No	x	x	Not Optimized
x	x	x	No	x	Not Optimized
x	x	x	x	No	Not Optimized

10.3.14.1.3 TR Pipelining and Data Ordering

The transfer controller(s) can issue back-to-back transfer requests (TR). The number of outstanding TRs for a EDMATC is limited by the number of destination FIFO register entries that is controlled by the DSTREGDEPTH parameter (fixed in design to a value of 4 entries for this device). TR pipelining refers to the ability of the EDMATC read controller to issue read commands for a subsequent TR, while the EDMATC write controller is still performing writes for the previous TR. Consider the case of 2 TRs (TR0 followed by TR1), because of TR pipelining, the EDMATC read controller can start issuing the read commands for TR1 as soon as the last read command for TR0 has been issued, meanwhile the write commands and write data for TR0 are tracked by the destination FIFO registers. In summary, the EDMATC read controller is able to process 4 TRs ahead of the write controller.

TR pipelining is useful for maintaining throughput on back-to-back small TRs. It minimizes the startup overhead because reads start in the background of a previous TR writes.

It should be noted that back-to-back TRs are targeted to different end points even though the read return data for the two TRs might get returned out of order (that is, read data for TR1 might come in before read data for TR0), the transfer controller issues the write commands in order (that is, write commands for TR0 will be issued before write commands for TR1).

10.3.14.1.4 Performance Tuning

By default, reads are as issued as fast as possible. In some cases, the reads issued by the EDMATC could fill the available command buffering for a slave, delaying other (potentially higher priority) masters from successfully submitting commands to that slave. The rate at which read commands are issued by the EDMATC is controlled by the EDMATC_RDRATE register. The EDMATC_RDRATE register defines the number of cycles that the EDMATC read controller waits before issuing subsequent commands for a given TR, thus minimizing the chance of the EDMATC consuming all available slave resources. The EDMATC_RDRATE value should be set to a relatively small value if the transfer controller is targeted for high priority transfers and to a higher value if the transfer controller is targeted for low priority transfers.

In contrast, the Write Interface does not have any performance turning knobs because writes always have an interval between commands as write commands are submitted along with the associated write data.

10.3.14.2 Memory Protection

The transfer controller plays an important role in handling proxy memory protection. There are two access properties associated with a transfer: for instance, the privilege id (system-wide identification assigned to a master) of the master initiating the transfer, and the privilege level (user versus supervisor) used to program the transfer. This information is maintained in the PaRAM set when it is programmed in the channel controller. When a TR is submitted to the transfer controller, this information is made available to the EDMATC and used by the EDMATC while issuing read and write commands. The read or write commands have the same privilege identification, and privilege level as that programmed in the EDMA transfer in the channel controller.

10.3.14.3 Error Generation

Similar to the channel controller, the transfer controllers are capable of detecting and reporting several error conditions. The EDMATC errors are generated, under three main conditions:

- BUSERR: The EDMATC read or write controllers detect an error signaled by the source or destination address. The additional details on the type of error is also recorded in the EDMATC_ERRDET register, which indicates whether it is a read error (source address errors) or write error (destination address error).
- MMRAERR: Attempt to read or write to an invalid/reserved addresses in the EDMACC/EDMATC memory map.
- TRERR: A transfer request packet is detected to be violating the constant addressing mode transfer rules (the source/destination addresses and source/destination indexes must be aligned to 32 bytes).

The user can poll for the errors, as the status of the errors can be read from the EDMATC_ERRSTAT registers. Additionally, if the error bits are enabled in the EDMATC_ERREN register, a bit set in the EDMATC_ERRSTAT will cause the error condition to interrupt the CPU(s). The user can decide to enable/disable either or all error types.

10.3.14.4 Debug Features

The DMA program register set, DMA source active register set, and the destination FIFO register set are used to derive a brief history of TRs serviced through the transfer controller.

Additionally, the EDMATC status register (EDMATC_TCSTAT) has dedicated bit fields to indicate the ongoing activity within different parts of the transfer controller:

- The SRCACTV bit indicates whether the source active set is active.
- The DSTACTV bit indicates the number of TRs resident in the destination register active set at a given instance.
- The PROGBUSY bit indicates whether a valid TR is present in the DMA program set.

If the TRs are in progression, caution must be used and the user must realize that there is a chance that the values read from the EDMATC status registers will be inconsistent since the EDMATC may change the values of these registers due to ongoing activities.

It is recommended that the user ensures no additional submission of TRs to the EDMATC in order to facilitate ease of debug.

10.3.14.4.1 Destination FIFO Register Pointer

The destination FIFO register pointer is implemented as a circular buffer with the start pointer being DFSTRTPTR and a buffer depth of 4 entries. The EDMATC maintains two important status details in EDMATC_TCSTAT that may be used during advanced debugging, if necessary. The DFSTRTPTR is a start pointer, that is, the index to the head of the destination FIFO register. The DSTACTV is a counter for the number of valid (occupied) entries. These registers may be used to get a brief history of transfers.

Examples of some register field values and their interpretation:

- DFSTRTPTR = 0 and DSTACTV = 0 implies that no TRs are stored in the destination FIFO register.
- DFSTRTPTR = 1 and DSTACTV = 2h implies that two TRs are present. The first pending TR is read from the destination FIFO register entry 1 and the second pending TR is read from the destination FIFO register entry 2.
- DFSTRTPTR = 3h and DSTACTV = 2h implies that two TRs are present. The first pending TR is read from the destination FIFO register entry 3 and the second pending TR is read from the destination FIFO register entry 0.

10.3.15 Event Dataflow

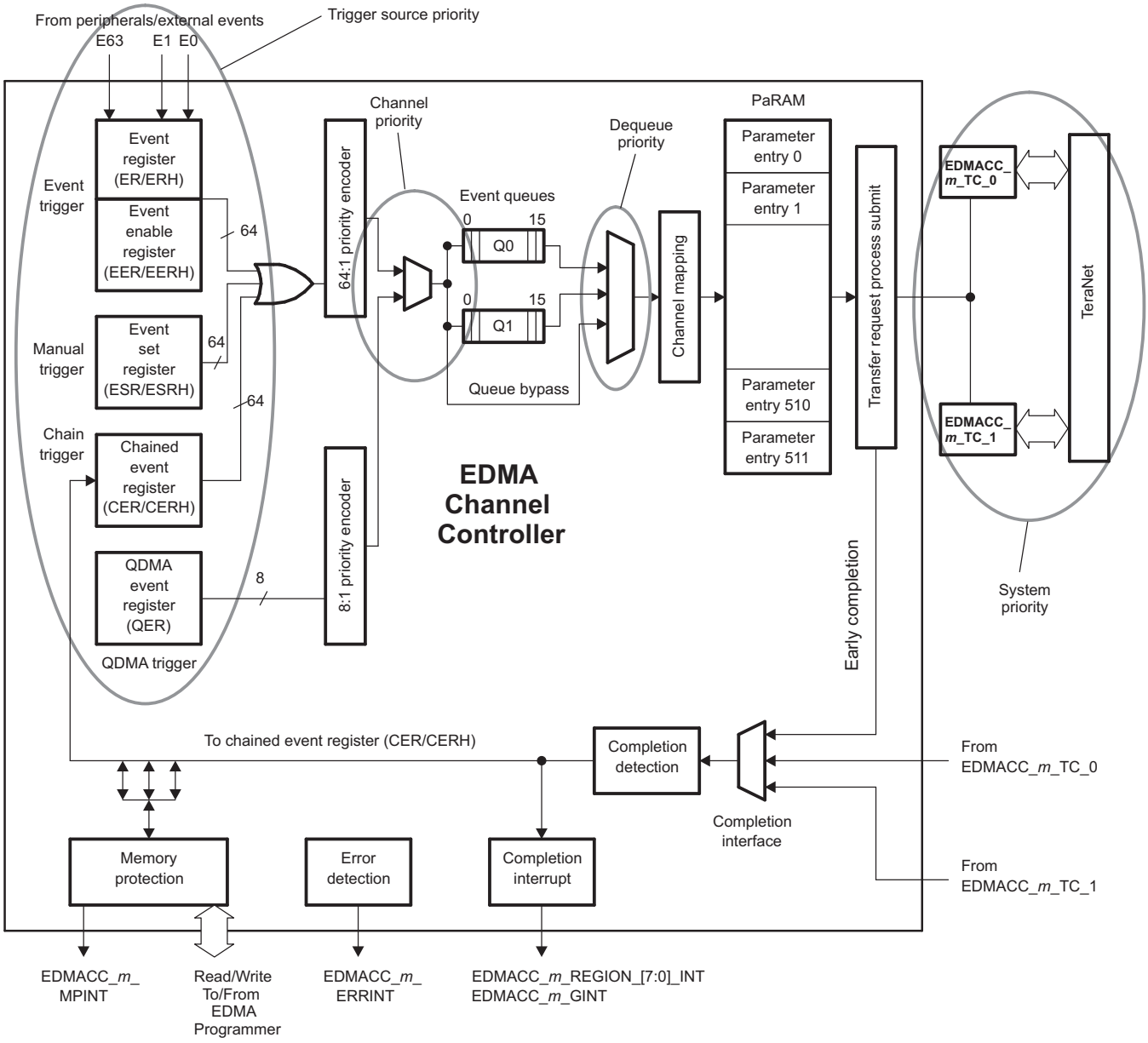
This section summarizes the data flow of a single event, from the time the event is latched to the channel controller to the time the transfer completion code is returned. The following steps list the sequence of EDMACC activity:

1. Event is asserted from an external source (peripheral or external interrupt). This also is similar for a manually-triggered, chained-triggered, or QDMA-triggered event. The event is latched into the EDMACC_ER.E *n*/EDMACC_ERH.E *n* (or EDMACC_CER.E *n*/EDMACC_CERH.E *n*, EDMACC_ESR.E *n* /EDMACC_ESRH.E *n*, EDMACC_QER.E *n*) bit.
2. Once an event is prioritized and queued into the appropriate event queue, the EDMACC_SER.E *n*/EDMACC_SERH.E *n* (or EDMACC_QSER.E *n*) bit is set to inform the event prioritization/processing logic to disregard this event since it is already in the queue. Alternatively, if the transfer controller and the event queue are empty, then the event bypasses the queue.
3. The EDMACC processing and the submission logic evaluates the appropriate PaRAM set and determines whether it is a non-null and non-dummy transfer request (TR).
4. The EDMACC clears the EDMACC_ER.E *n*/EDMACC_ERH.E *n* (or EDMACC_CER.E *n*/EDMACC_CERH.E *n*, EDMACC_ESR.E *n*/EDMACC_ESRH.E *n*, EDMACC_QER.E *n*) bit and the EDMACC_SER.E *n*/EDMACC_SERH.E *n* bit as soon as it determines the TR is non-null. In the case of a null set, the EDMACC_SER.E *n*/EDMACC_SERH.E *n* bit remains set. It submits the non-null/non-dummy TR to the associated transfer controller. If the TR was programmed for early completion, the EDMACC immediately sets the interrupt pending register (EDMACC_IPR.I[TCC]/EDMACC_IPRH.I[TCC]-32).
5. If the TR was programmed for normal completion, the EDMACC sets the interrupt pending register (EDMACC_IPR.I[TCC]/EDMACC_IPRH.I[TCC]) when the EDMATC informs the EDMACC about completion of the transfer (returns transfer completion codes).
6. The EDMACC programs the associated EDMATC *n*'s Program Register Set with the TR.
7. The TR is then passed to the Source Active set and the Dst FIFO Register Set, if both the register sets are available.
8. The Read Controller processes the TR by issuing read commands to the source slave endpoint. The Read Data lands in the Data FIFO of the EDMATC *n*.
9. As soon as sufficient data is available, the Write Controller begins processing the TR by issuing write commands to the destination slave endpoint.
10. This continues until the TR completes and the EDMATC *n* then signals completion status to the EDMACC.

10.3.16 EDMA Prioritization

The EDMA controller has many implementation rules to deal with concurrent events/channels, transfers, etc. The following subsections detail various arbitration details whenever there might be occurrence of concurrent activity. Figure 10-20 shows the different places EDMA priorities come into play.

Figure 10-20. EDMA Prioritization



NOTE: 'm' indicates the EDMACC instance number.

10.3.16.1 Trigger Source Priority

If a DMA channel is associated with more than one trigger source (event trigger, manual trigger, and chain trigger), and if multiple events are set simultaneously for the same channel (EDMACC_ER.E *n* = 1, EDMACC_ESR.E *n* = 1, EDMACC_CER.E *n* = 1), then the EDMACC always services these events in the following priority order: event trigger (via EDMACC_ER) is higher priority than chain trigger (via EDMACC_CER) and chain trigger is higher priority than manual trigger (via EDMACC_ESR).

This implies that if for channel 0, both EDMACC_ER.E0 = 1 and EDMACC_CER.E0 = 1 at the same time, then the EDMACC_ER.E0 event is always queued before the EDMACC_CER.E0 event.

10.3.16.2 Channel Priority

The DMA event registers (EDMACC_ER and EDMACC_ERH) captures all external/peripheral events connected to the EDMACC; likewise, the QDMA event register (EDMACC_QER) captures QDMA events for all QDMA channels; therefore, it is possible for events to occur simultaneously on the DMA/QDMA event inputs. For events arriving simultaneously, the event associated with the lowest channel number is prioritized for submission to the event queues (for DMA events, channel 0 has the highest priority and channel 63 has the lowest priority; similarly, for QDMA events, channel 0 has the highest priority and channel 7 has the lowest priority). This mechanism only sorts simultaneous events for submission to the event queues.

If a DMA and QDMA event occurs simultaneously, the DMA event always has prioritization against the QDMA event for submission to the event queues.

10.3.16.3 Dequeue Priority

The priority of the associated transfer request (TR) is further mitigated by which event queue is being used for event submission (dictated by DMAQNUM *n* and EDMACC_QDMAQNUM). For submission of a TR to the transfer controller, events need to be dequeued from the event queues. The lower numbered queue (Q0) has a higher dequeuing priority then the higher numbered queue (Q1). For example, if there are events in Q0 and Q1 and the respective transfer controllers (EDMATC0 and EDMATC1) are ready to receive the next TR from the EDMACC, then the transfer requests associated with events in Q0 will get submitted to EDMATC0 prior to any transfer requests associated with events in Q1 getting submitted to EDMATC1.

NOTE: At any given time, if there are outstanding events in both queues, when the transfer controller associated with Q0 (higher priority) queue is busy processing earlier transfer requests and the transfer controller associated with the higher numbered (lower priority) queue is idle, then the event in Q1 (lower priority) queue will de-queue first.

10.3.16.4 System (Transfer Controller) Priority

Every transfer controller has a programmable system priority (programmed via EDMACC_QUEPRI) (Figure 10-20) that is implemented when multiple masters in the system are vying for the same endpoint. The priority of the associated transfer request (TR) is further mitigated by the system priority setting of the transfer controller. This priority comes into play at the TeraNet when several masters are submitting requests to the main TeraNet.

The default priority for all TCs is the same (0 or highest priority to other masters). If an application requires the EDMA to service both real-time (urgent) and non-real-time transfers, it is recommended that this priority be changed.

10.3.17 EDMA Reset Considerations

A hardware reset initializes all internal logic of the EDMA controller (EDMACC and EDMATC) and the EDMA configuration registers. Note that the PaRAM memory contents are undefined after device reset and the user should not rely on parameters to be reset to a known state. The PaRAM set must be initialized to a desired value before it is used.

10.3.18 EDMA Emulation Considerations

During debug when using the emulator, the CPU(s) may be halted on an execute packet boundary for single-stepping, benchmarking, profiling, or other debug purposes. During an emulation halt, the EDMA channel controller and transfer controller operations continue. Events continue to be latched and processed and transfer requests continue to be submitted and serviced.

Since EDMA is involved in servicing multiple master and slave peripherals, it is not feasible to have an independent behavior of the EDMA for emulation halts. EDMA functionality would be coupled with the peripherals it is servicing, which might have different behavior during emulation halts. For example, if a serial port interface (SPI) is halted during an emulation access, the SPI stops generating the receive (REVT) or transmit (XEVT) events to the EDMA. From SPI point of view, the EDMA is suspended but other peripherals (for example, a timer) still assert events and will be serviced by the EDMA.

10.4 EDMA Transfer Examples

The EDMACC performs a variety of transfers depending on the parameter configuration. The following sections provide a description and PaRAM configuration for some typical use case scenarios.

10.4.1 Block Move Example

The most basic transfer performed by the EDMA is a block move. During device operation it is often necessary to transfer a block of data from one location to another, usually between on-chip and off-chip memory.

In this example, a section of data is to be copied from external memory to internal DSP L2 SRAM. A data block of 256 words residing at address 8000 0000h (external memory) needs to be transferred to internal address 0080 0000h (DSP L2 SRAM), as shown in Figure 10-21. Figure 10-22 shows the parameters for this transfer.

The source address for the transfer is set to the start of the data block in external memory, and the destination address is set to the start of the data block in DSP L2 SRAM. If the data block is less than 64K bytes, the PaRAM configuration shown in Figure 10-22 holds true with the synchronization type set to A-synchronized and indexes cleared to 0. If the amount of data is greater than 64K bytes, BCNT and the B-indexes need to be set appropriately with the synchronization type set to AB-synchronized. The STATIC bit in OPT is set to prevent linking.

This transfer example may also be set up using QDMA. For successive transfer submissions, of a similar nature, the number of cycles used to submit the transfer are fewer depending on the number of changing transfer parameters. The user may program the QDMA trigger word to be the highest numbered offset in the PaRAM set that undergoes change.

Figure 10-21. Block Move Example

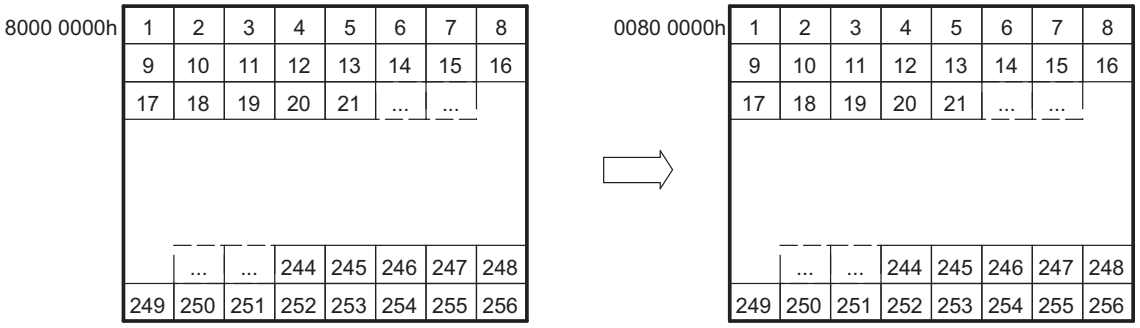


Figure 10-22. Block Move Example PaRAM Configuration

(a) EDMA Parameters

Parameter Contents	Parameter
0010 0008h	Channel Options Parameter (OPT)
80000000h	Channel Source Address (SRC)
0001h | 0100h	Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT)
1080 0000h	Channel Destination Address (DST)
0000h | 0000h	Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX)
0000h | FFFFh	BCNT Reload (BCNTRLD) | Link Address (LINK)
0000h | 0000h	Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX)
0000h | 0001h	Reserved | Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	1	0	0	1
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

10.4.2 Subframe Extraction Example

The EDMA can efficiently extract a small frame of data from a larger frame of data. By performing a 2D-to-1D transfer, the EDMA retrieves a portion of data for the DSP to process. In this example, a 640×480-pixel frame of video data is stored in external memory. Each pixel is represented by a 16-bit halfword. The DSP extracts a 16×12-pixel subframe of the image for processing. To facilitate more efficient processing time by the DSP, the EDMA places the subframe in internal L2 SRAM. Figure 10-23 shows the transfer of a subframe from external memory to L2. Figure 10-24 shows the parameters for this transfer.

The same PaRAM entry options are used for QDMA channels, as well as DMA channels. The STATIC bit in OPT is set to prevent linking. For successive transfers, only changed parameters need to be programmed before triggering the channel.

Figure 10-23. Subframe Extraction Example

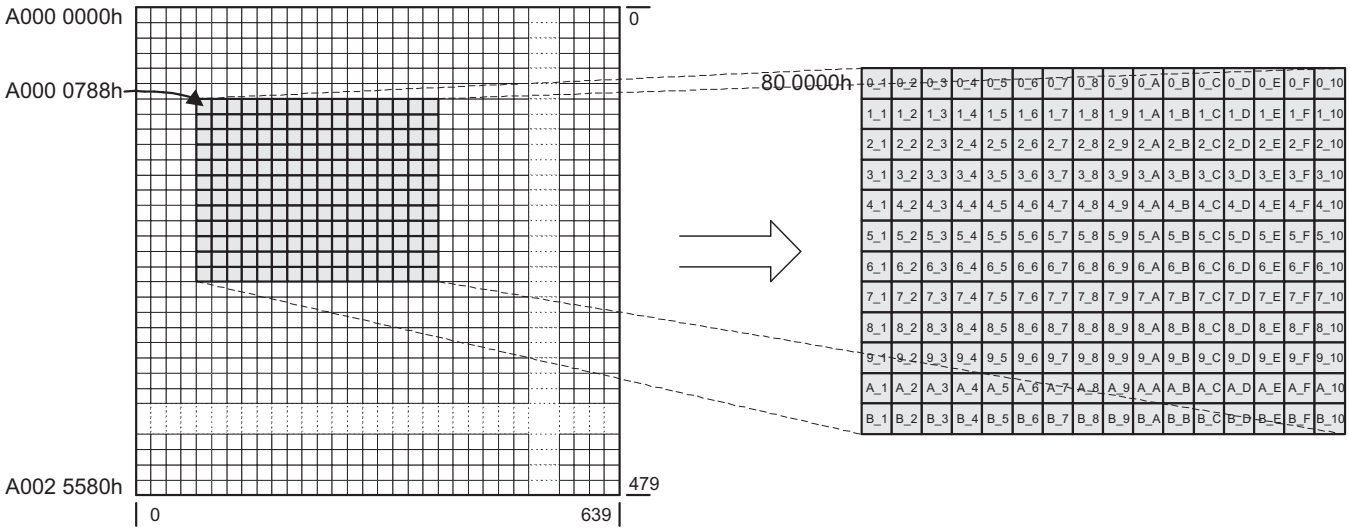


Figure 10-24. Subframe Extraction Example PaRAM Configuration

(a) EDMA Parameters

Parameter Contents	Parameter
0010 000Ch	Channel Options Parameter (OPT)
A000 0788h	Channel Source Address (SRC)
000Ch | 0020h	Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT)
1080 0000h	Channel Destination Address (DST)
0020h | 0500h	Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX)
0000h | FFFFh	BCNT Reload (BCNTRLD) | Link Address (LINK)
0000h | 0000h	Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX)
0000h | 0001h	Reserved | Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	1	1	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

10.4.3 Data Sorting Example

Many applications require the use of multiple data arrays; it is often desirable to have the arrays arranged such that the first elements of each array are adjacent, the second elements are adjacent, and so on.

Often this is not how the data is presented to the device. Either data is transferred via a peripheral with the data arrays arriving one after the other or the arrays are located in memory with each array occupying a portion of contiguous memory spaces. For these instances, the EDMA can reorganize the data into the desired format. Figure 10-25 shows the data sorting.

To determine the parameter set values, the following need to be considered:

- ACNT – The user should program this to be the size in bytes of an element.
- BCNT – The user should program this to be the number of elements in a frame.
- CCNT – The user should program this to be the number of frames.
- SRCBIDX – The user should program this to be the size of the element or ACNT.
- DSTBIDX = CCNT × ACNT
- SRCCDX = ACNT × BCNT
- DSTCIDX = ACNT

The synchronization type needs to be AB-synchronized and the STATIC bit is 0 to allow updates to the parameter set. It is advised to use normal EDMA channels for sorting.

It is not possible to sort this with a single trigger event. Instead, the channel can be programmed to be chained to itself. After BCNT elements get sorted, intermediate chaining could be used to trigger the channel again causing the transfer of the next BCNT elements and so on. Figure 10-26 shows the parameter set programming for this transfer, assuming channel 0 and an element size of 4 bytes.

Figure 10-25. Data Sorting Example

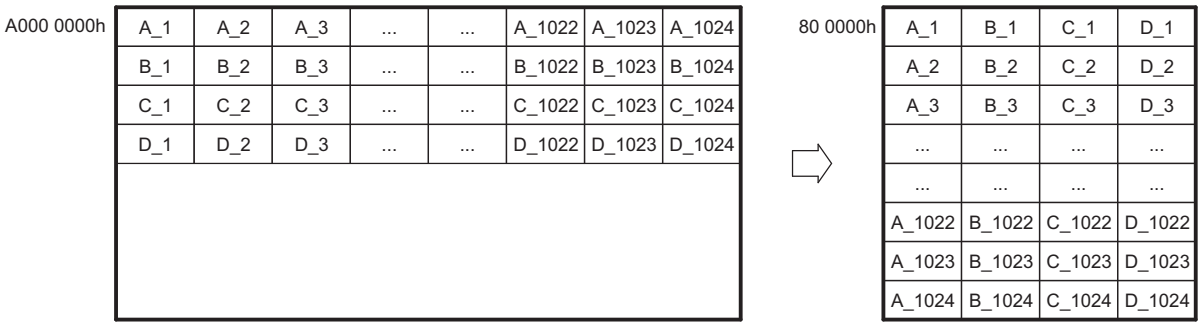


Figure 10-26. Data Sorting Example PaRAM Configuration

(a) EDMA Parameters

Parameter Contents	Parameter
0090 0004h	Channel Options Parameter (OPT)
A000 0000h	Channel Source Address (SRC)
0400h | 0004h	Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT)
1080 0000h	Channel Destination Address (DST)
0010h | 0001h	Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX)
0000h | FFFFh	BCNT Reload (BCNTRLD) | Link Address (LINK)
0001h | 1000h	Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX)
0000h | 0004h	Reserved | Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	1	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	0	1	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

10.4.4 Peripheral Servicing Example

The EDMA channel controller also services peripherals in the background of DSP operation, without requiring any DSP intervention. Through proper initialization of the EDMA channels, they can be configured to continuously service on-chip and off-chip peripherals throughout the device operation. Each event available to the EDMA has its own dedicated channel, and all channels operate simultaneously.

The only requirements are to use the proper channel for a particular transfer and to enable the channel event in the event enable register (EDMACC_EER). When programming an EDMA channel to service a peripheral, it is necessary to know how data is to be presented to the DSP. Data is always provided with some kind of synchronization event as either one element per event (non-bursting) or multiple elements per event (bursting).

10.4.4.1 Non-Bursting Peripherals

The non-bursting peripheral transmit and receive data streams are treated independently by the EDMA. The transmit and receive data streams can have completely different counts, data sizes, and formats. Figure 10-27 shows servicing incoming non-bursting peripheral data.

To transfer the incoming data stream to its proper location in L2 memory, the EDMA channel must be set up for a 1D-to-1D transfer with A-synchronization. Because a receive event (REVT) is generated for every word as it arrives, it is necessary to have the EDMA issue the transfer request for each element individually. Figure 10-28 shows the parameters for this transfer. The source address of the EDMA channel is set to the data receive register address of the non-bursting peripheral, and the destination address is set to the start of the data block in L2. Because the address of data receive register is fixed, the source B index is cleared to 0 (no modification) and the destination B index is set to 01b (increment).

Based on the premise that serial data is typically a high priority, the EDMA channel should be programmed to be on queue 0.

Figure 10-27. Servicing Incoming Non-Bursting Peripheral Data Example

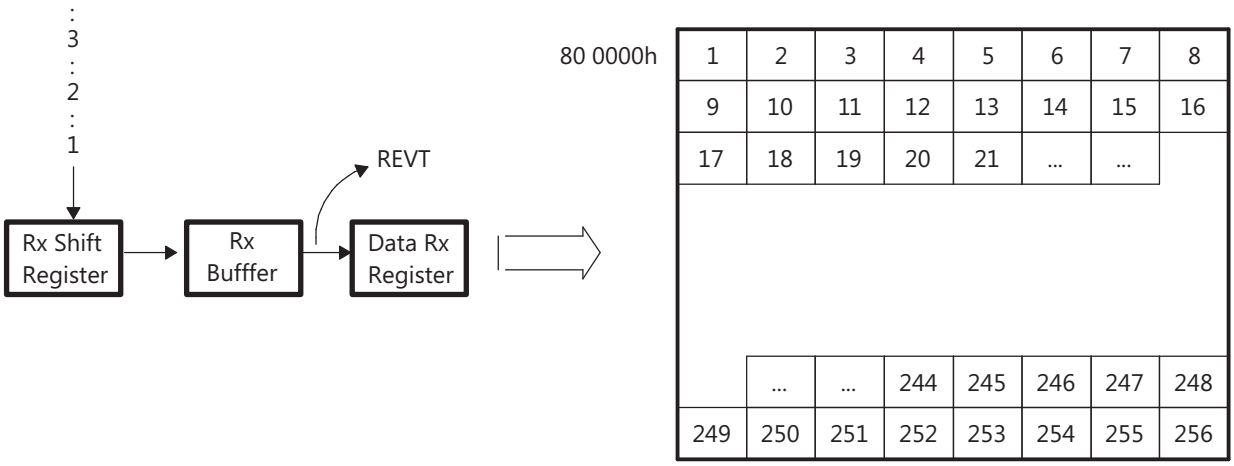


Figure 10-28. Servicing Incoming Non-Bursting Peripheral Data Example PaRAM Configuration

(a) EDMA Parameters

Parameter Contents		Parameter	
0010 0000h		Channel Options Parameter (OPT)	
Data receive register address		Channel Source Address (SRC)	
0100h	0001h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
1080 0000h		Channel Destination Address (DST)	
0001h	0000h	Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
0000h	FFFFh	BCNT Reload (BCNTRLD)	Link Address (LINK)
0000h	0000h	Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
0000h	0004h	Reserved	Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	0	0	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

10.4.4.2 Bursting Peripherals

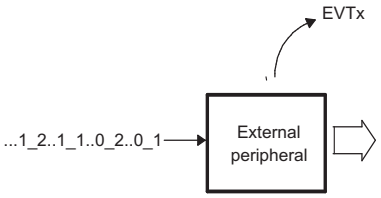
Higher bandwidth applications require that multiple data elements be presented to the DSP for every synchronization event. This frame of data can either be from multiple sources that are working simultaneously or from a single high-throughput peripheral that streams data to/from the DSP.

In this example, a port is receiving a video frame from a camera and presenting it to the DSP one array at a time. The video image is 640×480 pixels, with each pixel represented by a 16-bit element. The image is to be stored in external memory. Figure 10-29 shows this example.

To transfer data from an external peripheral to an external buffer one array at a time based on EVTn, channel n must be configured. Due to the nature of the data (a video frame made up of arrays of pixels) the destination is essentially a 2D entity. Figure 10-30 shows the parameters to service the incoming data with a 1D-to-2D transfer using AB-synchronization. The source address is set to the location of the video framer peripheral, and the destination address is set to the start of the data buffer. Because the input address is static, the SRCBIDX is 0 (no modification to the source address). The destination is made up of arrays of contiguous, linear elements; therefore, the DSTBIDX is set to pixel size, 2 bytes. ANCT is equal to the pixel size, 2 bytes. BCNT is set to the number of pixels in an array, 640. CCNT is equal to the total number of arrays in the block, 480. SRCCIDX is 0 because the source address undergoes no increment.

The DSTCIDX is equal to the difference between the starting addresses of each array. Because a pixel is 16 bits (2 bytes), DSTCIDX is equal to 640×2.

Figure 10-29. Servicing Peripheral Burst Example



A000 0000h	0_1	0_2	0_3	...	...	0_638	0_639	0_640
A000 0500h	1_1	1_2	...			...	1_639	1_640
A000 0A00h	2_1	...					...	2_640
...	...							...
...	...							...
A009 5100h	477_1	...					...	477_640
A009 5600h	478_1	478_2	...			...	478_639	478_640
A009 5B00h	479_1	479_2	479_3	...	...	479_638	479_639	479_640

Figure 10-30. Servicing Peripheral Burst Example PaRAM Configuration

(a) EDMA Parameters

Parameter Contents		Parameter	
0010 0004h		Channel Options Parameter (OPT)	
Channel Source Address		Channel Source Address (SRC)	
0280h	0002h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
E000 0000h		Channel Destination Address (DST)	
0002h	0000h	Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
0000h	FFFFh	BCNT Reload (BCNTRLD)	Link Address (LINK)
0500h	0000h	Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
0000h	01E0h	Reserved	Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	0	1	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

10.4.4.3 Continuous Operation

Configuring an EDMA channel to receive a single frame of data is useful, and is applicable to some systems. A majority of the time, however, data is going to be continuously transmitted and received throughout the entire operation of the DSP. In this case, it is necessary to implement some form of linking such that the EDMA channels continuously reload the necessary parameter sets. In this example, non-bursting peripheral is configured to transmit and receive data. To simplify the example, only two channels are active for both transmit and receive data streams. Each channel receives packets of 128 elements. The packets are transferred from the serial port to L2 memory and from L2 memory to the serial port, as shown Figure 10-31.

The non-bursting peripheral generates REVT for every element received and generates XEVT for every element transmitted. To service the data streams, EDMA channels 12 and 13 must be set up for 1D-to-1D transfers with A-synchronization.

Figure 10-32 shows the parameter entries for the channel for these transfers. To service the non-bursting peripheral continuously throughout DSP operation, the channels must be linked to a duplicate PaRAM set in the PaRAM. After all frames have been transferred, the EDMA channels reload and continue. Figure 10-33 shows the reload parameters for the channel.

10.4.4.3.1 Receive Channel

EDMA channel 13 services the incoming data stream of non-bursting peripheral. The source address is set to that of the data receive register of the non-bursting peripheral, and the destination address is set to the first element of the data block.

Because there are two data channels being serviced, A and B, they are to be located separately within the L2 SRAM.

To facilitate continuous operation, a copy of the PaRAM set for the channel is placed in PaRAM set 64. The LINK option is set and the link address is provided in the PaRAM set. Upon exhausting the channel 13 parameter set, the parameters located at the link address are loaded into the channel 13 parameter set and operation continues. This function continues throughout device operation until halted by the DSP.

10.4.4.3.2 Transmit Channel

EDMA channel 12 services the outgoing data stream of non-bursting peripheral. In this case the destination address needs no update, hence, the parameter set changes accordingly. Linking is also used to allow continuous operation by the EDMA channel, with duplicate PaRAM set entries at PaRAM set 65.

Figure 10-31. Servicing Continuous Non-Bursting Peripheral Data Example

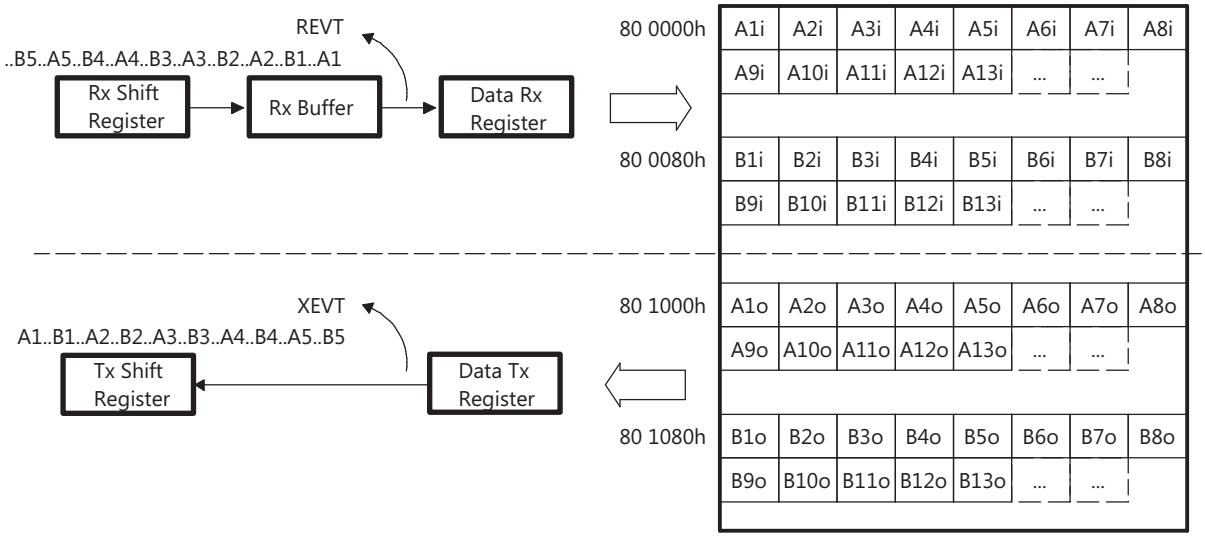


Figure 10-32. Servicing Continuous Non-Bursting Peripheral Data Example PaRAM Configuration

(a) EDMA Parameters for Receive Channel (PaRAM Set 13) being Linked to PaRAM Set 64

Parameter Contents	
0010 0000h	
Data receive register address	
0080h	0001h
1080 0000h	
0001h	0000h
0080h	4800h
0000h	0000h
0000h	FFFFh

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)	
Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
Reserved	Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content for Receive Channel (PaRAM Set 13)

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	0	0	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

(c) EDMA Parameters for Transmit Channel (PaRAM Set 12) being Linked to PaRAM Set 65

Parameter Contents	
0010 1000h	
0080 1000h	
0080h	0001h
Data transmit register address	
0000h	0001h
0080h	4820h
0000h	0000h
0000h	FFFFh

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)	
Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
Reserved	Count for 3rd Dimension (CCNT)

(d) Channel Options Parameter (OPT) Content for Transmit Channel (PaRAM Set 12)

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0001	0	000	0000	0	0	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

Figure 10-33. Servicing Continuous Non-Bursting Peripheral Data Example Reload PaRAM Configuration

(a) EDMA Reload Parameters (PaRAM Set 64) for Receive Channel

Parameter Contents	Parameter
0010 0000h	Channel Options Parameter (OPT)
Data receive register address	Channel Source Address (SRC)
0080h | 0001h	Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT)
0080 0000h	Channel Destination Address (DST)
0001h | 0000h	Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX)
0080h | 4800h	BCNT Reload (BCNTRLD) | Link Address (LINK)
0000h | 0000h	Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX)
0000h | FFFFh	Reserved | Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content for Receive Channel (PaRAM Set 64)

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	0	0	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

(c) EDMA Reload Parameters (PaRAM Set 65) for Transmit Channel

Parameter Contents	Parameter
0010 1000h	Channel Options Parameter (OPT)
0080 1000h	Channel Source Address (SRC)
0080h | 0001h	Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT)
Data transmit register address	Channel Destination Address (DST)
0001h | 0001h	Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX)
0080h | 4820h	BCNT Reload (BCNTRLD) | Link Address (LINK)
0000h | 0000h	Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX)
0000h | FFFFh	Reserved | Count for 3rd Dimension (CCNT)

(d) Channel Options Parameter (OPT) Content for Transmit Channel (PaRAM Set 65)

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
0001	0	000	0000	0	0	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

10.4.4.4 Ping-Pong Buffering

Although the previous configuration allows the EDMA to service a peripheral continuously, it presents a number of restrictions to the DSP. Because the input and output buffers are continuously being filled/emptied, the DSP must match the pace of the EDMA very closely to process the data. The EDMA receive data must always be placed in memory before the DSP accesses it, and the DSP must provide the output data before the EDMA transfers it. Though not impossible, this is an unnecessary challenge. It is particularly difficult in a 2-level cache scheme.

Ping-pong buffering is a simple technique that allows the DSP activity to be distanced from the EDMA activity. This means that there are multiple (usually two) sets of data buffers for all incoming and outgoing data streams. While the EDMA transfers the data into and out of the ping buffers, the DSP manipulates the data in the pong buffers. When both DSP and EDMA activity completes, they switch. The EDMA then writes over the old input data and transfers the new output data. Figure 10-34 shows the ping-pong scheme for this example.

To change the continuous operation example, such that a ping-pong buffering scheme is used, the EDMA channels need only a moderate change. Instead of one parameter set, there are two; one for transferring data to/from the ping buffers and one for transferring data to/from the pong buffers. As soon as one transfer completes, the channel loads the PaRAM set for the other and the data transfers continue. Figure 10-35 shows the EDMA channel configuration required.

Each channel has two parameter sets, ping and pong. The EDMA channel is initially loaded with the ping parameters (Figure 10-35). The link address for the ping set is set to the PaRAM offset of the pong parameter set (Figure 10-36). The link address for the pong set is set to the PaRAM offset of the ping parameter set (Figure 10-37). The channel options, count values, and index values are all identical between the ping and pong parameters for each channel. The only differences are the link address provided and the address of the data buffer.

10.4.4.4.1 Synchronization with the DSP

To utilize the ping-pong buffering technique, the system must signal the DSP when to begin to access the new data set. After the DSP finishes processing an input buffer (ping), it waits for the EDMA to complete before switching to the alternate (pong) buffer. In this example, both channels provide their channel numbers as their report word and set the TCINTEN bit to generate an interrupt after completion. When channel 13 fills an input buffer, the E13 bit in the interrupt pending register (EDMACC_IPR) is set; when channel 12 empties an output buffer, the E12 bit in EDMACC_IPR is set. The DSP must manually clear these bits. With the channel parameters set, the DSP polls EDMACC_IPR to determine when to switch. The EDMA and DSP could alternatively be configured such that the channel completion interrupts the DSP. By doing this, the DSP could service a background task while waiting for the EDMA to complete.

Figure 10-34. Ping-Pong Buffering for Non-Bursting Peripheral Data Example

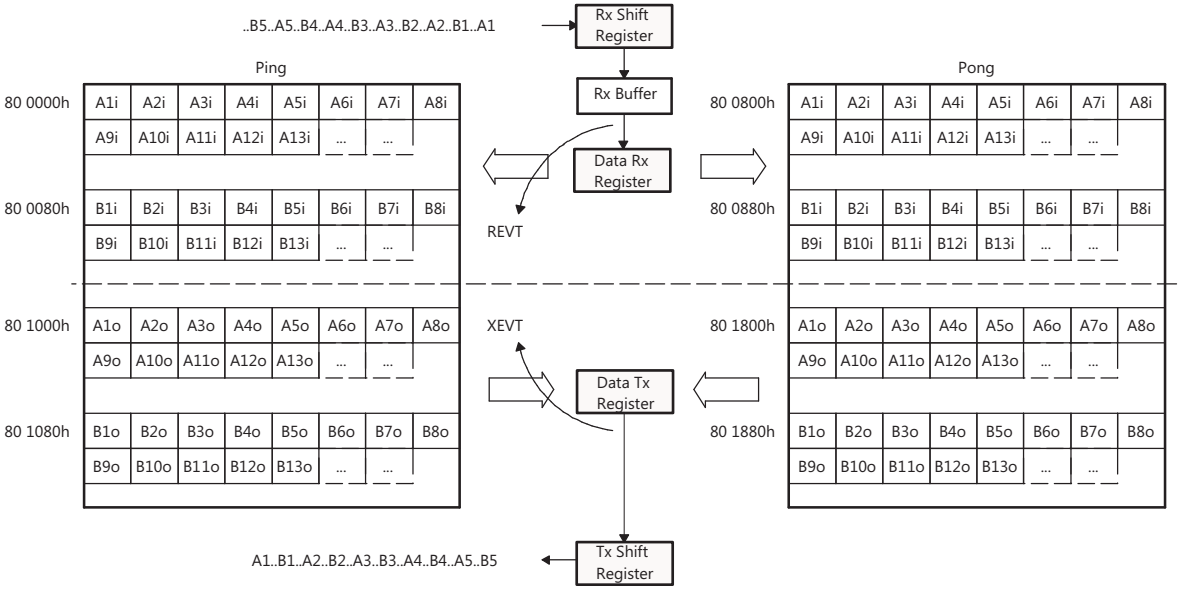


Figure 10-35. Ping-Pong Buffering for Non-Bursting Peripheral Example PaRAM Configuration

(a) EDMA Parameters for Channel 13 (Using PaRAM Set 13 Linked to Pong Set 64)

Parameter Contents	Parameter
0010 D000h	Channel Options Parameter (OPT)
Data receive register address	Channel Source Address (SRC)
0080h | 0001h	Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT)
0080 0000h	Channel Destination Address (DST)
0001h | 0000h	Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX)
0080h | 4800h	BCNT Reload (BCNTRLD) | Link Address (LINK)
0000h | 0000h	Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX)
0000h | FFFFh	Reserved | Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content for Channel 13

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
1101	0	000	0000	0	0	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

(c) EDMA Parameters for Channel 12 (Using PaRAM Set 12 Linked to Pong Set 65)

Parameter Contents	Parameter
0010 C000h	Channel Options Parameter (OPT)
0080 1000h	Channel Source Address (SRC)
0080h | 0001h	Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT)
Data transmit register address	Channel Destination Address (DST)
0001h | 0001h	Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX)
0080h | 4840h	BCNT Reload (BCNTRLD) | Link Address (LINK)
0000h | 0000h	Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX)
0000h | 0001h	Reserved | Count for 3rd Dimension (CCNT)

(d) Channel Options Parameter (OPT) Content for Channel 12

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	00	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15 12	11	10 8	7 4	3	2	1	0
1100	0	000	0000	0	0	0	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

Figure 10-36. Ping-Pong Buffering for Non-Bursting Peripheral Example Pong PaRAM Configuration

(a) EDMA Pong Parameters for Channel 13 at Set 64 Linked to Set 65

Parameter Contents		Parameter	
0010 D000h		Channel Options Parameter (OPT)	
Data receive register address		Channel Source Address (SRC)	
0080h	0001h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
0080 0800h		Channel Destination Address (DST)	
0001h	0000h	Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
0080h	4820h	BCNT Reload (BCNTRLD)	Link Address (LINK)
0000h	0000h	Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
0000h	0001h	Reserved	Count for 3rd Dimension (CCNT)

(b) EDMA Pong Parameters for Channel 12 at Set 66 Linked to Set 67

Parameter Contents		Parameter	
0010 C000h		Channel Options Parameter (OPT)	
0080 1800h		Channel Source Address (SRC)	
0080h	0001h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Data transmit register address		Channel Destination Address (DST)	
0000h	0001h	Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
0080h	4860h	BCNT Reload (BCNTRLD)	Link Address (LINK)
0000h	0000h	Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
0000h	0001h	Reserved	Count for 3rd Dimension (CCNT)

Figure 10-37. Ping-Pong Buffering for Non-Bursting Peripheral Example Ping PaRAM Configuration

(a) EDMA Ping Parameters for Channel 13 at Set 65 Linked to Set 64

Parameter Contents		Parameter	
0010 D000h		Channel Options Parameter (OPT)	
Data receive register address		Channel Source Address (SRC)	
0080h	0001h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
0080 0000h		Channel Destination Address (DST)	
0001h	0000h	Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
0080h	4800h	BCNT Reload (BCNTRLD)	Link Address (LINK)
0000h	0000h	Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
0000h	0001h	Reserved	Count for 3rd Dimension (CCNT)

(b) EDMA Ping Parameters for Channel 12 at Set 67 Linked to Set 66

Parameter Contents		Parameter	
0010 C000h		Channel Options Parameter (OPT)	
0080 1000h		Channel Source Address (SRC)	
0080h	0001h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Data transmit register address		Channel Destination Address (DST)	
0000h	0001h	Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
0080h	4840h	BCNT Reload (BCNTRLD)	Link Address (LINK)
0000h	0000h	Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
0000h	0001h	Reserved	Count for 3rd Dimension (CCNT)

10.4.4.5 Transfer Chaining Examples

The following examples explain the intermediate transfer complete chaining function.

10.4.4.5.1 Servicing Input/Output FIFOs with a Single Event

Many systems require the use of a pair of external FIFOs that must be serviced at the same rate. One FIFO buffers data input, and the other buffers data output. The EDMA channels that service these FIFOs can be set up for AB-synchronized transfers. While each FIFO is serviced with a different set of parameters, both can be signalled from a single event. For example, an external interrupt pin can be tied to the status flags of one of the FIFOs. When this event arrives, the EDMA needs to perform servicing for both the input and output streams. Without the intermediate transfer complete chaining feature this would require two events, and thus two external interrupt pins. The intermediate transfer complete chaining feature allows the use of a single external event (for example, a GPIO event). Figure 10-38 shows the EDMA setup and illustration for this example.

A GPIO event (in this example, mapped to EDMACC_0 input #48 via GPIOMUX48) triggers an array transfer. Upon completion of each intermediate array transfer of channel 48, intermediate transfer complete chaining sets the E8 bit (specified by TCC of 8) in the chained event register (EDMACC_CER) and provides a synchronization event to channel 8. Upon completion of the last array transfer of channel 48, transfer complete chaining – not intermediate transfer complete chaining – sets the E8 bit in EDMACC_CER (specified by TCCMODE:TCC) and provides a synchronization event to channel 8. The completion of channel 8 sets the I8 bit (specified by TCCMODE:TCC) in the interrupt pending register (EDMACC_IPR), which can generate an interrupt to the DSP, if the I8 bit in the interrupt enable register (EDMACC_IER) is set.

10.4.4.5.2 Breaking Up Large Transfers with Intermediate Chaining

Another feature of intermediate transfer chaining (ITCCHEN) is for breaking up large transfers. A large transfer may lock out other transfers of the same priority level for the duration of the transfer. For example, a large transfer on queue 0 from the internal memory to the external memory using the EMIF may starve other EDMA transfers on the same queue. In addition, this large high-priority transfer may prevent the EMIF for a long duration to service other lower priority transfers. When a large transfer is considered to be high priority, it should be split into multiple smaller transfers. Figure 10-39 shows the EDMA setup and illustration of an example single large block transfer.

The intermediate transfer chaining enable (ITCCHEN) provides a method to break up a large transfer into smaller transfers. For example, to move a single large block of memory (16K bytes), the EDMA performs an A-synchronized transfer. The element count is set to a reasonable value, where reasonable derives from the amount of time it would take to move this smaller amount of data. Assume 1 Kbyte is a reasonable small transfer in this example. The EDMA is set up to transfer 16 arrays of 1 Kbyte elements, for a total of 16K byte elements. The TCC field in the channel options parameter (OPT) is set to the same value as the channel number and ITCCHEN are set. In this example, EDMA channel 25 is used and TCC is also set to 25. The TCINTEN may also be set to trigger interrupt 25 when the last 1 Kbyte array is transferred. The DSP starts the EDMA transfer by writing to the appropriate bit of the event set register (EDMACC_ESR.E25). The EDMA transfers the first 1 Kbyte array. Upon completion of the first array, intermediate transfer complete code chaining generates a synchronization event to channel 25, a value specified by the TCC field. This intermediate transfer completion chaining event causes EDMA channel 25 to transfer the next 1 Kbyte array. This process continues until the transfer parameters are exhausted, at which point the EDMA has completed the 16K byte transfer. This method breaks up a large transfer into smaller packets, thus providing natural time slices in the transfer such that other events may be processed. Figure 10-40 shows the EDMA setup and illustration of the broken up smaller packet transfers.

Figure 10-38. Intermediate Transfer Completion Chaining Example

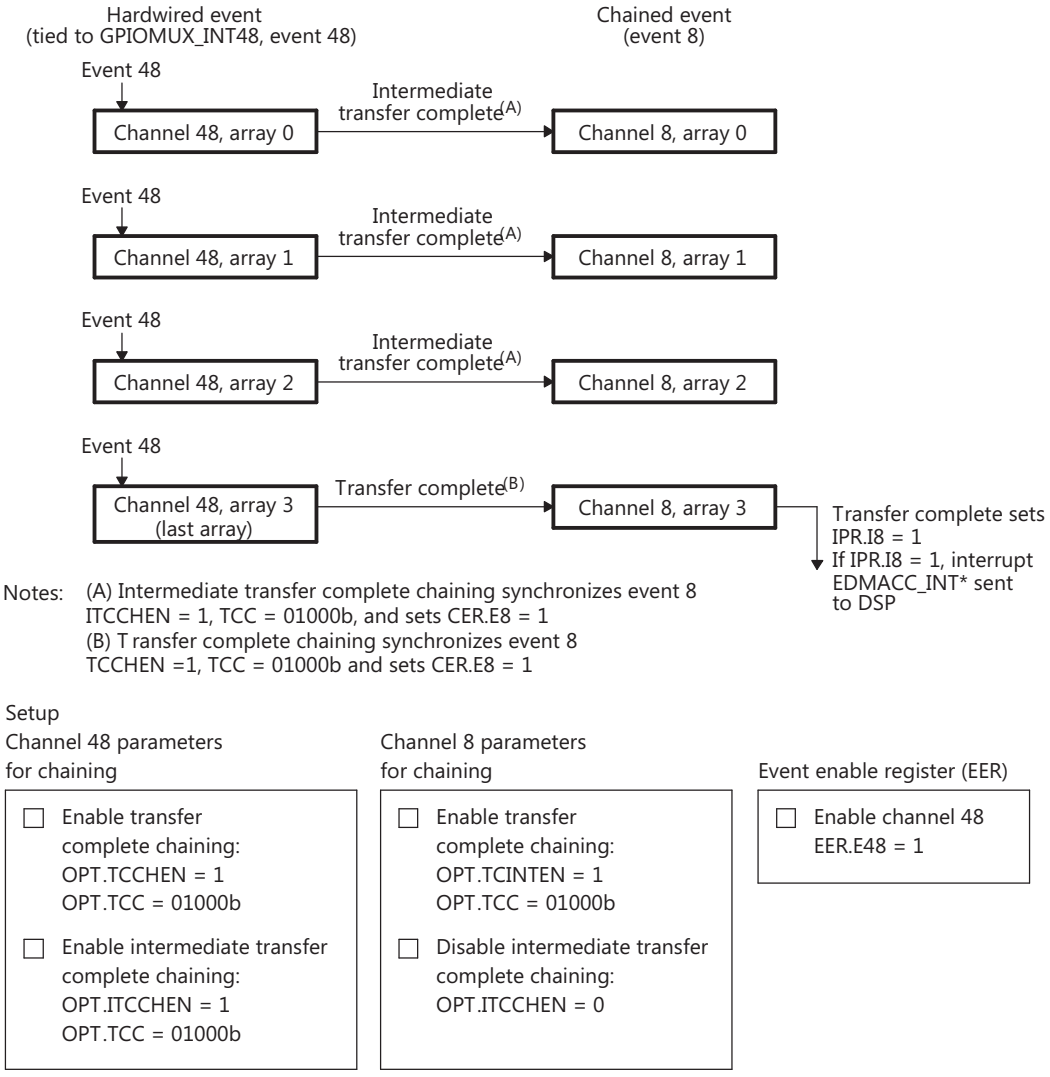


Figure 10-39. Single Large Block Transfer Example

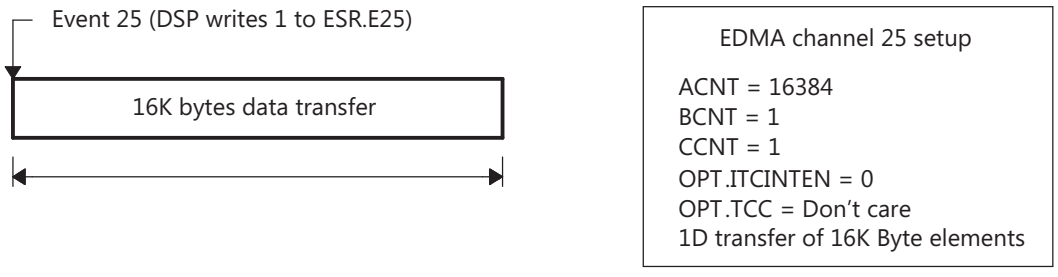
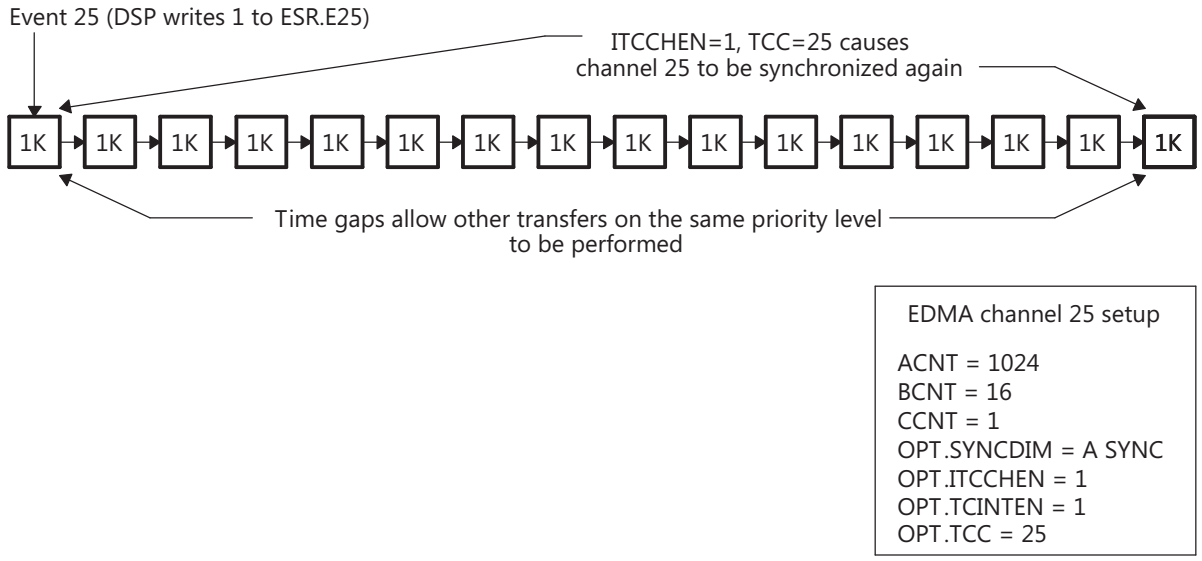


Figure 10-40. Smaller Packet Data Transfers Example



10.5 EDMA Debug/Programming Tips

10.5.1 Debug Checklist

This section lists some tips to keep in mind while debugging applications using the EDMA controller. Table 10-25 provides some common issues and their probable causes and resolutions.

Table 10-25. Debug Checklist

Issue	Description/Solution
The transfer associated with the channel does not happen. The channel does not get serviced.	The EDMACC may not service a transfer request, even though the associated PaRAM set is programmed appropriately. Check for the following: 1) Verify that events are enabled, i.e., if an external/peripheral event is latched in Event Verify that events are enabled, i.e., if an external/peripheral event is latched in Event Registers (EDMACC_ER/EDMACC_ERH), make sure that the event is enabled in the Event Enable Registers (EDMACC_EER/EDMACC_EERH). Similarly, for QDMA channels, make sure that QDMA events are appropriately enabled in the QDMA Event Enable Register (EDMACC_QEER). 2) Verify that the DMA or QDMA Secondary Event Register (EDMACC_SER/EDMACC_SERH/QSERH) bits corresponding to the particular event or channel are not set.
The Secondary Event Registers bits are set, not allowing additional transfers to occur on a channel.	It is possible that a trigger event was received when the parameter set associated with the channel/event was a NULL set for a previous transfer on the channel. This is typical in two cases: 1) QDMA channels: Typically if the parameter set is non-static and expected to be terminated by a NULL set (i.e., OPT.STATIC = 0, LINK = FFFFh), the parameter set is updated with a NULL set after submission of the last TR. Because QDMA channels are auto-triggered, this update caused the generation of an event. An event generated for a NULL set causes an error condition and results in setting the bits corresponding to the QDMA channel in the EDMACC_QEMR and EDMACC_QSER. This will disable further prioritization of the channel. 2) DMA channels used in a continuous mode: The peripheral may be set up to continuously generate infinite events (for instance, in case of Non-bursting peripheral, every time the data shifts out from the Data Transmit register, it generates an XEVT). The parameter set may be programmed to expect only a finite number of events and to be terminated by a NULL link. After the expected number of events, the parameter set is reloaded with a NULL parameter set. Because the peripheral will generate additional events, an error condition is set in the EDMACC_SER.Ex and EDMACC_EMR.Ex set, preventing further event prioritization. The user must ensure that the number of events received is limited to the expected number of events for which the parameter set is programmed, or the user must ensure that bits corresponding to particular channel or event are not set in the Secondary event registers (EDMACC_SER/EDMACC_SERH/EDMACC_QSER) and Event Missed Registers (EDMACC_EMR/EDMACC_EMRH/EDMACC_QEMR) before trying to perform subsequent transfers for the event/channel.
Completion interrupts are not asserted, or no further interrupts are received after the first completion interrupt.	The user must ensure the following: 1) The interrupt generation is enabled in the OPT of the associated PaRAM set (TCINTEN = 1 and/or ITCINTEN = 1). 2) The interrupts are enabled in the EDMACC, via the Interrupt Enable Registers (EDMACC_IER/EDMACC_IERH). 3) The corresponding interrupts are enabled in the device interrupt controller. 4) The set interrupts are cleared in the interrupt pending registers (EDMACC_IPR/EDMACC_IPRH) before exiting the transfer completion interrupt service routine (ISR). For details on writing EDMA ISRs, see Section 10.3.11.1.2. 5) If working with shadow region interrupts, make sure that the DMA Region Access registers (DRAE/DRAEH) are set up properly, because the DRAE/DRAEH registers act as secondary enables for shadow region completion interrupts, along with the EDMACC_IER/EDMACC_IERH registers. If working with shadow region interrupts, make sure that the bits corresponding to the transfer completion code (TCC) value are also enabled in the DRAE/DRAEH registers. For instance, if the PaRAM set associated with Channel 0 returns a completion code of 63 (OPT.TCC = 63), ensure that DRAEH.E63 is also set for a shadow region completion interrupt because the interrupt pending register bit set will be EDMACC_IPRH.I63 (not EDMACC_IPR.I0).

10.5.2 Miscellaneous Programming/Debug Tips

Setting and Clearing Bits

For several registers, the setting and clearing of bits needs to be done via separate dedicated registers. For example, the Event Register (EDMACC_ER/EDMACC_ERH) can only be cleared by writing a 1 to the corresponding bits in the Event Clear Registers (EDMACC_ECR/EDMACC_ECRH). Similarly, the Event Enable Register (EDMACC_EER/EDMACC_EERH) bits can only be set with writes of 1 to the Event Enable Set Registers (EDMACC_EESR/EDMACC_EESRH) and cleared with writes of 1 to the corresponding bits in the Event Enable Clear Register (EDMACC_EECR/EDMACC_EECRH).

Writing to Shadow Region Memory Maps

Writes to the shadow region memory maps are governed by region access registers (DRAE/DRAEH/QRAE). If the appropriate channels are not enabled in these registers, read/write access to the shadow region memory map is not enabled.

Shadow Region Completion Interrupts

When working with shadow region completion interrupts, ensure that the DMA Region Access Registers (DRAE/DRAEH) for every region are set in a mutually exclusive way (unless it is a requirement for an application). If there is an overlap in the allocated channels and transfer completion codes (setting of Interrupt Pending Register bits) in the region resource allocation, it results in multiple shadow region completion interrupts. For example, if EDMACC_DRAE0.E0 and EDMACC_DRAE1.E0 are both set, then on completion of a transfer that returns a TCC = 0, they will generate both shadow region 0 and 1 completion interrupts.

Programming a Non-Dummy Parameter Set

While programming a non-dummy parameter set, ensure the CCNT is not left to zero.

Ensuring Error Conditions are Not Missed

Enable the EDMACC error interrupt in the device controller and attach an interrupt service routine (ISR) to ensure that error conditions are not missed in an application and are appropriately addressed with the ISR.

Breaking Up Transfers

Depending on the application, the user may want to break large transfers into smaller transfers and use self-chaining to prevent starvation of other events in an event queue.

Chaining

In applications where a large transfer is broken into sets of small transfers using chaining or other methods, the user might choose to use the early chaining option to reduce the time between the sets of transfers and increase the throughput. However, keep in mind that with early completion, all data might have not been received at the end point when completion is reported because the EDMACC internally signals completion when the TR is submitted to the EDMATC, potentially before any data has been transferred.

View Event Queue Entries

The event queue entries can be observed to determine the last few events if there is a system failure (provided the entries were not bypassed).

10.5.3 Setting Up a Transfer

The following procedure provides a quick guide for the typical steps involved in setting up a transfer:

Step 1. Initiating a DMA/QDMA channel

a. Determine the type of channel (QDMA or DMA) to be used.

b. Channel mapping

i. If using a QDMA Channel, program the QCHMAP with the parameter set number to which the channel maps and the trigger word.

ii. If using a DMA channel, program the DCHMAP with the parameter set number to which the channel maps.

c. If the channel is being used in the context of a shadow region, ensure the DRAE/DRAEH for the region is properly set up to allow read write accesses to bits in the event registers and interrupt registers in the Shadow region memory map. The subsequent steps in this process should be done using the respective shadow region registers. (Shadow region descriptions and usage are provided in Table 10-15)

d. Determine the type of triggering used.

   i. If external events are used for triggering (DMA channels), enable the respective event in EDMACC_EER/EDMACC_EERH by writing into EDMACC_EESR/EDMACC_EESRH.

   ii. If QDMA Channel is used, enable the channel in EDMACC_QEER by writing into EDMACC_QEESR.

e. Queue setup

   i. If a QDMA channel is used, set up the EDMACC_QDMAQNUM to map the channel to the respective event queue.

   ii. If a DMA channel is used, set up the DMAQNUM to map the event to the respective event queue.

Step 2. Parameter set setup
Program the PaRAM set number associated with the channel. Note that if it is a QDMA channel, the PaRAM entry that is configured as trigger word is written to last. Alternatively, enable the QDMA channel (step 1-b-ii above) just before the write to the trigger word. See Section 10.4 for parameter set field setups for different types of transfers. See the sections on chaining () and interrupt completion (Section 10.3.11) on how to set up final/intermediate completion chaining and/or interrupts.

Step 3. Interrupt setup

a. Enable the interrupt in the EDMACC_IER/EDMACC_IERH by writing into EDMACC_IESR/EDMACC_IESRH.

b. Ensure the EDMACC completion interrupt (this refers to either the global interrupt or the shadow region interrupt) is enabled properly in the device interrupt controller.

c. Set up the interrupt controller properly to receive the expected EDMA interrupt.

Step 4. Initiate transfer
This step is highly dependent on the event trigger source:

a. If the source is an external event coming from a peripheral, the peripheral will be enabled to start generating relevant EDMA events that can be latched to the EDMACC_ER transfer.

b. For QDMA events, writes to the trigger word (step 2-a above) will initiate the transfer.

c. Manually triggered transfers will be initiated by writes to the Event Set Registers (EDMACC_ESR/EDMACC_ESRH).

d. Chained-trigger events initiate when a previous transfer returns a transfer completion code equal to the chained channel number.

Step 5. Wait for completion

a. If the interrupts are enabled as mentioned in step 3 above, then the EDMACC will generate a completion interrupt to the DSP whenever transfer completion results in setting the corresponding bits in the interrupt pending register (EDMACC_IPR/EDMACC_IPRH). The set bits must be cleared in the EDMACC_IPR/EDMACC_IPRH by writing to corresponding bit in EDMACC_ICR/EDMACC_ICRH.

b. If polling for completion (interrupts not enabled in the device controller), then the application code can wait on the expected bits to be set in the EDMACC_IPR/EDMACC_IPRH. Again, the set bits in the EDMACC_IPR/EDMACC_IPRH must be manually cleared via EDMACC_ICR/EDMACC_ICRH before the next set of transfers is performed for the same transfer completion code values.

10.6 EDMA Registers

This section describes the memory-mapped registers associated with the EDMA controller.

10.6.1 EDMA Channel Controller (EDMACC) Registers

Table 10-26 lists the base address for each of the EDMACC module instances.

Table 10-27 lists the memory-mapped registers for the EDMA channel controller (EDMACC). All register offset addresses not listed in Table 10-27 should be considered as reserved locations and the register contents should not be modified.

Table 10-26. EDMACC Instances

Instance	Base Address
EDMACC_0	0270 0000h
EDMACC_1	0272 8000h

Table 10-27. EDMACC Registers

Offset	Acronym	Register Name	EDMACC_0 Physical Address	EDMACC_1 Physical Address	Section
0h	EDMACC_PID	Peripheral Identification Register	0270 0000h	0272 8000h	Section 10.6.1.1.1
4h	EDMACC_CCCFG	EDMACC Configuration Register	0270 0004h	0272 8004h	Section 10.6.1.1.2
100h to 1FCh	EDMACC_DCHMAP0 to EDMACC_DCHMAP63	DMA Channel 0-63 Mapping Registers	0270 0100h to 0270 01FCh	0272 8100h to 0272 81FCh	Section 10.6.1.1.3
200h to 21Ch	EDMACC_QCHMAP0 to EDMACC_QCHMAP7	QDMA Channel 0-7 Mapping Register	0270 0200h to 0270 021Ch	0272 8200h to 0272 821Ch	Section 10.6.1.1.4
240h to 25Ch	EDMACC_DMAQNUM0 to EDMACC_DMAQNUM7	DMA Queue Number Register 0-7	0270 0240h to 0270 025Ch	0272 8240h to 0272 825Ch	Section 10.6.1.1.5
260h	EDMACC_QDMAQNUM	QDMA Queue Number Register	0270 0260h	0272 8260h	Section 10.6.1.1.6
280h	EDMACC_QUETCMAP	Queue-to-TC Mapping Register	0270 0280h	0272 8280h	Section 10.6.1.1.7
284h	EDMACC_QUEPRI	Queue Priority Register	0270 0284h	0272 8284h	Section 10.6.1.1.8
300h	EDMACC_EMR	Event Missed Register	0270 0300h	0272 8300h	Section 10.6.1.1.9
304h	EDMACC_EMRH	Event Missed Register High	0270 0304h	0272 8304h	Section 10.6.1.1.10
308h	EDMACC_EMCR	Event Missed Clear Register	0270 0308h	0272 8308h	Section 10.6.1.1.11
30Ch	EDMACC_EMCRH	Event Missed Clear Register High	0270 030Ch	0272 830Ch	Section 10.6.1.1.12
310h	EDMACC_QEMR	QDMA Event Missed Register	0270 0310h	0272 8310h	Section 10.6.1.1.13
314h	EDMACC_QEMCR	QDMA Event Missed Clear Register	0270 0314h	0272 8314h	Section 10.6.1.1.14
318h	EDMACC_CCERR	EDMACC Error Register	0270 0318h	0272 8318h	Section 10.6.1.1.15
31Ch	EDMACC_CCERRCLR	EDMACC Error Clear Register	0270 031Ch	0272 831Ch	Section 10.6.1.1.16
320h	EDMACC_EEVAL	Error Evaluate Register	0270 0320h	0272 8320h	Section 10.6.1.1.17
340h	EDMACC_DRAE0	DMA Region Access Enable Register for Region 0	0270 0340h	0272 8340h	Section 10.6.1.2.1

Table 10-27. EDMACC Registers (continued)

Offset	Acronym	Register Name	EDMACC_0 Physical Address	EDMACC_1 Physical Address	Section
344h	EDMACC_DRAEH0	DMA Region Access Enable Register High for Region 0	0270 0344h	0272 8344h	Section 10.6.1.2.2
348h	EDMACC_DRAE1	DMA Region Access Enable Register for Region 1	0270 0348h	0272 8348h	Section 10.6.1.2.3
34Ch	EDMACC_DRAEH1	DMA Region Access Enable Register High for Region 1	0270 034Ch	0272 834Ch	Section 10.6.1.2.4
350h	EDMACC_DRAE2	DMA Region Access Enable Register for Region 2	0270 0350h	0272 8350h	Section 10.6.1.2.5
354h	EDMACC_DRAEH2	DMA Region Access Enable Register High for Region 2	0270 0354h	0272 8354h	Section 10.6.1.2.6
358h	EDMACC_DRAE3	DMA Region Access Enable Register for Region 3	0270 0358h	0272 8358h	Section 10.6.1.2.7
35Ch	EDMACC_DRAEH3	DMA Region Access Enable Register High for Region 3	0270 035Ch	0272 835Ch	Section 10.6.1.2.8
360h	EDMACC_DRAE4	DMA Region Access Enable Register for Region 4	0270 0360h	0272 8360h	Section 10.6.1.2.9
364h	EDMACC_DRAEH4	DMA Region Access Enable Register High for Region 4	0270 0364h	0272 8364h	Section 10.6.1.2.10
368h	EDMACC_DRAE5	DMA Region Access Enable Register for Region 5	0270 0368h	0272 8368h	Section 10.6.1.2.11
36Ch	EDMACC_DRAEH5	DMA Region Access Enable Register High for Region 5	0270 036Ch	0272 836Ch	Section 10.6.1.2.12
370h	EDMACC_DRAE6	DMA Region Access Enable Register for Region 6	0270 0370h	0272 8370h	Section 10.6.1.2.13
374h	EDMACC_DRAEH6	DMA Region Access Enable Register High for Region 6	0270 0374h	0272 8374h	Section 10.6.1.2.14
378h	EDMACC_DRAE7	DMA Region Access Enable Register for Region 7	0270 0378h	0272 8378h	Section 10.6.1.2.15
37Ch	EDMACC_DRAEH7	DMA Region Access Enable Register High for Region 7	0270 037Ch	0272 837Ch	Section 10.6.1.2.16
380h to 39Ch	EDMACC_QRAE0 to EDMACC_QRAE7	QDMA Region Access Enable Registers for Region 0-7	0270 0380h to 0270 039Ch	0272 8380h to 0272 839Ch	Section 10.6.1.2.17
400h to 4FCh	EDMACC_Q0E0 to EDMACC_Q3E15	Event Queue Entry Registers	0270 0400h to 0270 04FCh	0272 8400h to 0272 84FCh	Section 10.6.1.3.1
600h to 60Ch	EDMACC_QSTAT0 to EDMACC_QSTAT3	Queue Status Registers 0-3	0270 0600h to 0270 060Ch	0272 8600h to 0272 860Ch	Section 10.6.1.3.2
620h	EDMACC_QWMTHRA	Queue Watermark Threshold A Register	0270 0620h	0272 8620h	Section 10.6.1.3.3
640h	EDMACC_CCSTAT	EDMACC Status Register	0270 0640h	0272 8640h	Section 10.6.1.3.4
800h	EDMACC_MPFAR	Memory Protection Fault Address Register	0270 0800h	0272 8800h	Section 10.6.1.4.1
804h	EDMACC_MPFSR	Memory Protection Fault Status Register	0270 0804h	0272 8804h	Section 10.6.1.4.2
808h	EDMACC_MPFCR	Memory Protection Fault Command Register	0270 0808h	0272 8808h	Section 10.6.1.4.3
80Ch	EDMACC_MPPAG	Memory Protection Page Attribute Global Register	0270 080Ch	0272 880Ch	Section 10.6.1.4.4
810h to 82Ch	EDMACC_MPPA0 to EDMACC_MPPA7	Memory Protection Page Attribute Registers 0-7	0270 0810h to 0270 082Ch	0272 8810h to 0272 882Ch	Section 10.6.1.4.5
		Global Channel Registers			
1000h	EDMACC_ER	Event Register	0270 1000h	0272 9000h	Section 10.6.1.5.1

Table 10-27. EDMACC Registers (continued)

Offset	Acronym	Register Name	EDMACC_0 Physical Address	EDMACC_1 Physical Address	Section
1004h	EDMACC_ERH	Event Register High	0270 1004h	0272 9004h	Section 10.6.1.5.2
1008h	EDMACC_ECR	Event Clear Register	0270 1008h	0272 9008h	Section 10.6.1.5.3
100Ch	EDMACC_ECRH	Event Clear Register High	0270 100Ch	0272 900Ch	Section 10.6.1.5.4
1010h	EDMACC_ESR	Event Set Register	0270 1010h	0272 9010h	Section 10.6.1.5.5
1014h	EDMACC_ESRH	Event Set Register High	0270 1014h	0272 9014h	Section 10.6.1.5.6
1018h	EDMACC_CER	Chained Event Register	0270 1018h	0272 9018h	Section 10.6.1.5.7
101Ch	EDMACC_CERH	Chained Event Register High	0270 101Ch	0272 901Ch	Section 10.6.1.5.8
1020h	EDMACC_EER	Event Enable Register	0270 1020h	0272 9020h	Section 10.6.1.5.9
1024h	EDMACC_EERH	Event Enable Register High	0270 1024h	0272 9024h	Section 10.6.1.5.10
1028h	EDMACC_EECR	Event Enable Clear Register	0270 1028h	0272 9028h	Section 10.6.1.5.11
102Ch	EDMACC_EECRH	Event Enable Clear Register High	0270 102Ch	0272 902Ch	Section 10.6.1.5.12
1030h	EDMACC_EESR	Event Enable Set Register	0270 1030h	0272 9030h	Section 10.6.1.5.13
1034h	EDMACC_EESRH	Event Enable Set Register High	0270 1034h	0272 9034h	Section 10.6.1.5.14
1038h	EDMACC_SER	Secondary Event Register	0270 1038h	0272 9038h	Section 10.6.1.5.15
103Ch	EDMACC_SERH	Secondary Event Register High	0270 103Ch	0272 903Ch	Section 10.6.1.5.16
1040h	EDMACC_SECR	Secondary Event Clear Register	0270 1040h	0272 9040h	Section 10.6.1.5.17
1044h	EDMACC_SECRH	Secondary Event Clear Register High	0270 1044h	0272 9044h	Section 10.6.1.5.18
1050h	EDMACC_IER	Interrupt Enable Register	0270 1050h	0272 9050h	Section 10.6.1.6.1
1054h	EDMACC_IERH	Interrupt Enable Register High	0270 1054h	0272 9054h	Section 10.6.1.6.2
1058h	EDMACC_IECR	Interrupt Enable Clear Register	0270 1058h	0272 9058h	Section 10.6.1.6.3
105Ch	EDMACC_IECRH	Interrupt Enable Clear Register High	0270 105Ch	0272 905Ch	Section 10.6.1.6.4
1060h	EDMACC_IESR	Interrupt Enable Set Register	0270 1060h	0272 9060h	Section 10.6.1.6.5
1064h	EDMACC_IESRH	Interrupt Enable Set Register High	0270 1064h	0272 9064h	Section 10.6.1.6.6
1068h	EDMACC_IPR	Interrupt Pending Register	0270 1068h	0272 9068h	Section 10.6.1.6.7
106Ch	EDMACC_IPRH	Interrupt Pending Register High	0270 106Ch	0272 906Ch	Section 10.6.1.6.8
1070h	EDMACC_ICR	Interrupt Clear Register	0270 1070h	0272 9070h	Section 10.6.1.6.9
1074h	EDMACC_ICRH	Interrupt Clear Register High	0270 1074h	0272 9074h	Section 10.6.1.6.10

Table 10-27. EDMACC Registers (continued)

Offset	Acronym	Register Name	EDMACC_0 Physical Address	EDMACC_1 Physical Address	Section
1078h	EDMACC_IEVAL	Interrupt Evaluate Register	0270 1078h	0272 9078h	Section 10.6.1.6.11
1080h	EDMACC_QER	QDMA Event Register	0270 1080h	0272 9080h	Section 10.6.1.7.1
1084h	EDMACC_QEER	QDMA Event Enable Register	0270 1084h	0272 9084h	Section 10.6.1.7.2
1088h	EDMACC_QEECR	QDMA Event Enable Clear Register	0270 1088h	0272 9088h	Section 10.6.1.7.3
108Ch	EDMACC_QEESR	QDMA Event Enable Set Register	0270 108Ch	0272 908Ch	Section 10.6.1.7.4
1090h	EDMACC_QSER	QDMA Secondary Event Register	0270 1090h	0272 9090h	Section 10.6.1.7.5
1094h	EDMACC_QSECR	QDMA Secondary Event Clear Register	0270 1094h	0272 9094h	Section 10.6.1.7.6
		Shadow Region 0 Channel Registers			
2000h	EDMACC_ER	Event Register	0270 2000h	0272 A000h	Section 10.6.1.5.1
2004h	EDMACC_ERH	Event Register High	0270 2004h	0272 A004h	Section 10.6.1.5.2
2008h	EDMACC_ECR	Event Clear Register	0270 2008h	0272 A008h	Section 10.6.1.5.3
200Ch	EDMACC_ECRH	Event Clear Register High	0270 200Ch	0272 A00Ch	Section 10.6.1.5.4
2010h	EDMACC_ESR	Event Set Register	0270 2010h	0272 A010h	Section 10.6.1.5.5
2014h	EDMACC_ESRH	Event Set Register High	0270 2014h	0272 A014h	Section 10.6.1.5.6
2018h	EDMACC_CER	Chained Event Register	0270 2018h	0272 A018h	Section 10.6.1.5.7
201Ch	EDMACC_CERH	Chained Event Register High	0270 201Ch	0272 A01Ch	Section 10.6.1.5.8
2020h	EDMACC_EER	Event Enable Register	0270 2020h	0272 A020h	Section 10.6.1.5.9
2024h	EDMACC_EERH	Event Enable Register High	0270 2024h	0272 A024h	Section 10.6.1.5.10
2028h	EDMACC_EECR	Event Enable Clear Register	0270 2028h	0272 A028h	Section 10.6.1.5.11
202Ch	EDMACC_EECRH	Event Enable Clear Register High	0270 202Ch	0272 A02Ch	Section 10.6.1.5.12
2030h	EDMACC_EESR	Event Enable Set Register	0270 2030h	0272 A030h	Section 10.6.1.5.13
2034h	EDMACC_EESRH	Event Enable Set Register High	0270 2034h	0272 A034h	Section 10.6.1.5.14
2038h	EDMACC_SER	Secondary Event Register	0270 2038h	0272 A038h	Section 10.6.1.5.15
203Ch	EDMACC_SERH	Secondary Event Register High	0270 203Ch	0272 A03Ch	Section 10.6.1.5.16
2040h	EDMACC_SECR	Secondary Event Clear Register	0270 2040h	0272 A040h	Section 10.6.1.5.17
2044h	EDMACC_SECRH	Secondary Event Clear Register High	0270 2044h	0272 A044h	Section 10.6.1.5.18
2050h	EDMACC_IER	Interrupt Enable Register	0270 2050h	0272 A050h	Section 10.6.1.6.1

Table 10-27. EDMACC Registers (continued)

Offset	Acronym	Register Name	EDMACC_0 Physical Address	EDMACC_1 Physical Address	Section
2054h	EDMACC_IERH	Interrupt Enable Register High	0270 2054h	0272 A054h	Section 10.6.1.6.2
2058h	EDMACC_IECR	Interrupt Enable Clear Register	0270 2058h	0272 A058h	Section 10.6.1.6.3
205Ch	EDMACC_IECRH	Interrupt Enable Clear Register High	0270 205Ch	0272 A05Ch	Section 10.6.1.6.4
2060h	EDMACC_IESR	Interrupt Enable Set Register	0270 2060h	0272 A060h	Section 10.6.1.6.5
2064h	EDMACC_IESRH	Interrupt Enable Set Register High	0270 2064h	0272 A064h	Section 10.6.1.6.6
2068h	EDMACC_IPR	Interrupt Pending Register	0270 2068h	0272 A068h	Section 10.6.1.6.7
206Ch	EDMACC_IPRH	Interrupt Pending Register High	0270 206Ch	0272 A06Ch	Section 10.6.1.6.8
2070h	EDMACC_ICR	Interrupt Clear Register	0270 2070h	0272 A070h	Section 10.6.1.6.9
2074h	EDMACC_ICRH	Interrupt Clear Register High	0270 2074h	0272 A074h	Section 10.6.1.6.10
2078h	EDMACC_IEVAL	Interrupt Evaluate Register	0270 2078h	0272 A078h	Section 10.6.1.6.11
2080h	EDMACC_QER	QDMA Event Register	0270 2080h	0272 A080h	Section 10.6.1.7.1
2084h	EDMACC_QEER	QDMA Event Enable Register	0270 2084h	0272 A084h	Section 10.6.1.7.2
2088h	EDMACC_QEECR	QDMA Event Enable Clear Register	0270 2088h	0272 A088h	Section 10.6.1.7.3
208Ch	EDMACC_QEESR	QDMA Event Enable Set Register	0270 208Ch	0272 A08Ch	Section 10.6.1.7.4
2090h	EDMACC_QSER	QDMA Secondary Event Register	0270 2090h	0272 A090h	Section 10.6.1.7.5
2094h	EDMACC_QSECR	QDMA Secondary Event Clear Register	0270 2094h	0272 A094h	Section 10.6.1.7.6
2200h to 2294h	...	Shadow Region 1 Channel Registers	0270 2200h to 0270 2294h	0272 A200h to 0272 A294h	
2400h to 2494h	...	Shadow Region 2 Channel Registers	0270 2400h to 0270 2494h	0272 A400h to 0272 A494h	
...	...	...	...	...	
2E00h to 2E94h	...	Shadow Region 7 Channel Registers	0270 2E00h to 0270 2E94h	0272 AE00h to 0272 AE94h	

10.6.1.1 Global Registers

10.6.1.1.1 EDMACC_PID Register (Offset = 0h) [reset = 4001AB00h]

The peripheral identification register (EDMACC_PID) uniquely identifies the EDMACC and the specific revision of the EDMACC.

The EDMACC_PID is shown in Figure 10-41 and described in Table 10-29.

Table 10-28. EDMACC_PID Instances

Instance	Physical Address
EDMACC_0	0270 0000h
EDMACC_1	0272 8000h

Figure 10-41. EDMACC_PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PID																															
R-4001AB00h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-29. EDMACC_PID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PID	R	4001AB00h	TI internal data. Identifies revision of peripheral.

Table 10-30. Register Call Summary for EDMACC_PID

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_PID Register (Offset = 0h) [reset = 4001AB00h]: [0][1]

10.6.1.1.2 EDMACC_CCCFG Register (Offset = 4h) [reset = 3305445h]

The EDMACC configuration register (EDMACC_CCCFG) provides the features/resources for the EDMACC in a particular device.

The EDMACC_CCCFG is shown in Figure 10-42 and described in Table 10-32.

Table 10-31. EDMACC_CCCFG Instances

Instance	Physical Address
EDMACC_0	0270 0004h
EDMACC_1	0272 8004h

Figure 10-42. EDMACC_CCCFG Register

31	30	29	28	27	26	25	24
RESERVED						MP_EXIST	CHMAP_EXIST
R-0h						R-1h	R-1h
23	22	21	20	19	18	17	16
RESERVED		NUM_REGN		RESERVED		NUM_EVQUE	
R-0h		R-3h		R-0h		R-0h	
15	14	13	12	11	10	9	8
RESERVED	NUM_PAENTRY			RESERVED		NUM_INTCH	
R-0h	R-5h			R-0h		R-4h	
7	6	5	4	3	2	1	0
RESERVED	NUM_QDMACH			RESERVED		NUM_DMACH	
R-0h	R-4h			R-0h		R-5h	

LEGEND: R = Read Only; *-n* = value after reset

Table 10-32. EDMACC_CCCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	MP_EXIST	R	1h	Memory protection existence. 0h = Reserved 1h = Memory protection logic included
24	CHMAP_EXIST	R	1h	Channel mapping existence. 0h = Reserved 1h = Channel mapping logic included
23-22	RESERVED	R	0h	Reserved
21-20	NUM_REGN	R	3h	Number of MP and shadow regions. 0h - 2h = Reserved 3h = 8 regions
19	RESERVED	R	0h	Reserved
18-16	NUM_EVQUE	R	0h	Number of queues/number of TCs. 0h = Reserved 1h = 2 EDMATCs/Event Queues 2h = Reserved 3h = 4 EDMATCs/Event Queues 4h - 7h = Reserved
15	RESERVED	R	0h	Reserved

Table 10-32. EDMACC_CCCFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14-12	NUM_PAENTRY	R	5h	Number of PaRAM sets. 0h - 2h = Reserved 3h = 128 PaRAM sets 4h = Reserved 5h = 512 PaRAM sets 6h - 7h = Reserved
11	RESERVED	R	0h	Reserved
10-8	NUM_INTCH	R	4h	Number of interrupt channels. 0h - 1h = Reserved 2h = 16 interrupt channels 3h = Reserved 4h = 64 interrupt channels 5h - 7h = Reserved
7	RESERVED	R	0h	Reserved
6-4	NUM_QDMACH	R	4h	Number of QDMA channels. 0h - 3h = Reserved 4h = 8 QDMA channels 5h - 7h = Reserved
3	RESERVED	R	0h	Reserved
2-0	NUM_DMACH	R	5h	Number of DMA channels 0h - 2h = Reserved 3h = 16 DMA channels 4h = Reserved 5h = 64 DMA channels 6h - 7h = Reserved

Table 10-33. Register Call Summary for EDMACC_CCCFG

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_CCCFG Register (Offset = 4h) [reset = 3300000h]: [0][1]

10.6.1.1.3 EDMACC_DCHMAP0 to EDMACC_DCHMAP63 Register (Offset = 100h to 1FCh) [reset = 0h]

The DMA channel map *n* register (DCHMAP *n*) is shown in Figure 10-43 and described in Table 10-35.

Table 10-34. EDMACC_DCHMAP0 to EDMACC_DCHMAP63 Instances

Instance	Physical Address
EDMACC_0	0270 0100h to 0270 01FCh
EDMACC_1	0272 8100h to 0272 81FCh

Figure 10-43. EDMACC_DCHMAP0 to EDMACC_DCHMAP63 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			PAENTRY				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
PAENTRY			RESERVED				
R/W-0h			R-0h				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 10-35. EDMACC_DCHMAP0 to EDMACC_DCHMAP63 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13-5	PAENTRY	R/W	0h	Points to the PaRAM set number for DMA channel *n*.
4-0	RESERVED	R	0h	Reserved

Table 10-36. Register Call Summary for EDMACC_DCHMAP0

EDMA Registers
• EDMA Channel Controller (EDMACC) Registers: [0]

10.6.1.1.4 EDMACC_QCHMAP0 to EDMACC_QCHMAP7 Register (Offset = 200h to 21Ch) [reset = 0h]

Each QDMA channel in EDMACC can be associated with any PaRAM set available on the device. Furthermore, the specific trigger word (0-7) of the PaRAM set can be programmed. The PaRAM set association and trigger word for every QDMA channel register is configurable using the QDMA channel map n register (QCHMAP *n*).

The QCHMAP *n* is shown in Figure 10-44 and described in Table 10-38.

NOTE: At reset the QDMA channel map registers for all QDMA channels point to PaRAM set 0. If an application makes use of both a DMA channel that points to PaRAM set 0 and any QDMA channels, ensure that QCHMAP *n* is programmed appropriately to point to a different PaRAM entry.

Table 10-37. EDMACC_QCHMAP0 to EDMACC_QCHMAP7 Instances

Instance	Physical Address
EDMACC_0	0270 0200h to 0270 021Ch
EDMACC_1	0272 8200h to 0272 821Ch

Figure 10-44. EDMACC_QCHMAP0 to EDMACC_QCHMAP7 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			PAENTRY				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
PAENTRY			TRWORD			RESERVED	
R/W-0h			R/W-0h			R-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 10-38. EDMACC_QCHMAP0 to EDMACC_QCHMAP7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13-5	PAENTRY	R/W	0h	Points to the PaRAM set number for QDMA channel *n*. Value = 0-1FFh
4-2	TRWORD	R/W	0h	Points to the specific trigger word of the PaRAM set defined by PAENTRY. A write to the trigger word results in a QDMA event being recognized. Value = 0-7h
1-0	RESERVED	R	0h	Reserved

Table 10-39. Register Call Summary for EDMACC_QCHMAP0

EDMA Registers
• EDMA Channel Controller (EDMACC) Registers: [0]

10.6.1.1.5 EDMACC_DMAQNUM0 to EDMACC_DMAQNUM7 Register (Offset = 240h to 25Ch) [reset = 0h]

The DMA channel queue number register (DMAQNUM *n*) allows programmability of the DMA channels in the EDMACC to submit its associated synchronization event to any event queue in the EDMACC. At reset, all channels point to event queue 0.

The DMAQNUM *n* is shown in Figure 10-45 and described in Table 10-41. shows the channels and their corresponding bits in DMAQNUM *n*.

NOTE: Because the event queues in EDMACC have a fixed association to the transfer controllers, that is, Q0 TRs are submitted to TC0, Q1 TRs are submitted to TC1, etc., by programming DMAQNUM *n* for a particular DMA channel *n* also dictates which transfer controller is utilized for the data movement (or which EDMATC receives the TR request).

N refer to the number of DMA Channels (0 to 7).

Table 10-40. EDMACC_DMAQNUM0 to EDMACC_DMAQNUM7 Instances

Instance	Physical Address
EDMACC_0	0270 0240h to 0270 025Ch
EDMACC_1	0272 8240h to 0272 825Ch

Figure 10-45. EDMACC_DMAQNUM0 to EDMACC_DMAQNUM7 Register

31	30	29	28	27	26	25	24
RESERVED	E(7+N*8)			RESERVED	E(6+N*8)		
R-0h	R/W-0h			R-0h	R/W-0h		
23	22	21	20	19	18	17	16
RESERVED	E(5+N*8)			RESERVED	E(4+N*8)		
R-0h	R/W-0h			R-0h	R/W-0h		
15	14	13	12	11	10	9	8
RESERVED	E(3+N*8)			RESERVED	E(2+N*8)		
R-0h	R/W-0h			R-0h	R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	E(1+N*8)			RESERVED	E(0+N*8)		
R-0h	R/W-0h			R-0h	R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 10-41. EDMACC_DMAQNUM0 to EDMACC_DMAQNUM7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30-28	E(7+N*8)	R/W	0h	DMA queue number. Contains the event queue number to be used for the corresponding DMA channel. Programming DMAQNUM *N* for an event queue number to a value more than the number of queues available in the EDMACC results in undefined behavior. 0h = Event *N* is queued on Q0 1h = Event *N* is queued on Q1 2h = Event *N* is queued on Q2 3h = Event *N* is queued on Q3 4h - 7h = Reserved
27	RESERVED	R	0h	Reserved

Table 10-41. EDMACC_DMAQNUM0 to EDMACC_DMAQNUM7 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26-24	E(6+N*8)	R/W	0h	DMA queue number. Contains the event queue number to be used for the corresponding DMA channel. Programming DMAQNUM *N* for an event queue number to a value more than the number of queues available in the EDMACC results in undefined behavior. 0h = Event *N* is queued on Q0 1h = Event *N* is queued on Q1 2h = Event *N* is queued on Q2 3h = Event *N* is queued on Q3 4h - 7h = Reserved
23	RESERVED	R	0h	Reserved
22-20	E(5+N*8)	R/W	0h	DMA queue number. Contains the event queue number to be used for the corresponding DMA channel. Programming DMAQNUM *N* for an event queue number to a value more than the number of queues available in the EDMACC results in undefined behavior. 0h = Event *N* is queued on Q0 1h = Event *N* is queued on Q1 2h = Event *N* is queued on Q2 3h = Event *N* is queued on Q3 4h - 7h = Reserved
19	RESERVED	R	0h	Reserved
18-16	E(4+N*8)	R/W	0h	DMA queue number. Contains the event queue number to be used for the corresponding DMA channel. Programming DMAQNUM *N* for an event queue number to a value more than the number of queues available in the EDMACC results in undefined behavior. 0h = Event *N* is queued on Q0 1h = Event *N* is queued on Q1 2h = Event *N* is queued on Q2 3h = Event *N* is queued on Q3 4h - 7h = Reserved
15	RESERVED	R	0h	Reserved
14-12	E(3+N*8)	R/W	0h	DMA queue number. Contains the event queue number to be used for the corresponding DMA channel. Programming DMAQNUM *N* for an event queue number to a value more than the number of queues available in the EDMACC results in undefined behavior. 0h = Event *N* is queued on Q0 1h = Event *N* is queued on Q1 2h = Event *N* is queued on Q2 3h = Event *N* is queued on Q3 4h - 7h = Reserved
11	RESERVED	R	0h	Reserved
10-8	E(2+N*8)	R/W	0h	DMA queue number. Contains the event queue number to be used for the corresponding DMA channel. Programming DMAQNUM *N* for an event queue number to a value more than the number of queues available in the EDMACC results in undefined behavior. 0h = Event *N* is queued on Q0 1h = Event *N* is queued on Q1 2h = Event *N* is queued on Q2 3h = Event *N* is queued on Q3 4h - 7h = Reserved
7	RESERVED	R	0h	Reserved

Table 10-41. EDMACC_DMAQNUM0 to EDMACC_DMAQNUM7 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-4	E(1+N*8)	R/W	0h	DMA queue number. Contains the event queue number to be used for the corresponding DMA channel. Programming DMAQNUM *N* for an event queue number to a value more than the number of queues available in the EDMACC results in undefined behavior. 0h = Event *N* is queued on Q0 1h = Event *N* is queued on Q1 2h = Event *N* is queued on Q2 3h = Event *N* is queued on Q3 4h - 7h = Reserved
3	RESERVED	R	0h	Reserved
2-0	E(0+N*8)	R/W	0h	DMA queue number. Contains the event queue number to be used for the corresponding DMA channel. Programming DMAQNUM *N* for an event queue number to a value more than the number of queues available in the EDMACC results in undefined behavior. 0h = Event *N* is queued on Q0 1h = Event *N* is queued on Q1 2h = Event *N* is queued on Q2 3h = Event *N* is queued on Q3 4h - 7h = Reserved

Table 10-42. Register Call Summary for EDMACC_DMAQNUM0

EDMA Registers
• EDMA Channel Controller (EDMACC) Registers: [0]

10.6.1.1.6 EDMACC_QDMAQNUM Register (Offset = 260h) [reset = 0h]

The QDMA channel queue number register (EDMACC_QDMAQNUM) is used to program all the QDMA channels in the EDMACC to submit the associated QDMA event to any of the event queues in the EDMACC.

The EDMACC_QDMAQNUM is shown in Figure 10-46 and described in Table 10-44.

Table 10-43. EDMACC_QDMAQNUM Instances

Instance	Physical Address
EDMACC_0	0270 0260h
EDMACC_1	0272 8260h

Figure 10-46. EDMACC_QDMAQNUM Register

31	30	29	28	27	26	25	24
RESERVED	E7			RESERVED	E6		
R-0h	R/W-0h			R-0h	R/W-0h		
23	22	21	20	19	18	17	16
RESERVED	E5			RESERVED	E4		
R-0h	R/W-0h			R-0h	R/W-0h		
15	14	13	12	11	10	9	8
RESERVED	E3			RESERVED	E2		
R-0h	R/W-0h			R-0h	R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	E1			RESERVED	E0		
R-0h	R/W-0h			R-0h	R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 10-44. EDMACC_QDMAQNUM Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30-28	E7	R/W	0h	QDMA queue number. Contains the event queue number to be used for the corresponding QDMA channel. 0h = Event *N* is queued on Q0 1h = Event *N* is queued on Q1 2h = Event *N* is queued on Q2 3h = Event *N* is queued on Q3 4h - 7h = Reserved
27	RESERVED	R	0h	Reserved
26-24	E6	R/W	0h	QDMA queue number. Contains the event queue number to be used for the corresponding QDMA channel. 0h = Event *N* is queued on Q0 1h = Event *N* is queued on Q1 2h = Event *N* is queued on Q2 3h = Event *N* is queued on Q3 4h - 7h = Reserved
23	RESERVED	R	0h	Reserved
22-20	E5	R/W	0h	QDMA queue number. Contains the event queue number to be used for the corresponding QDMA channel. 0h = Event *N* is queued on Q0 1h = Event *N* is queued on Q1 2h = Event *N* is queued on Q2 3h = Event *N* is queued on Q3 4h - 7h = Reserved

Table 10-44. EDMACC_QDMAQNUM Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	RESERVED	R	0h	Reserved
18-16	E4	R/W	0h	QDMA queue number. Contains the event queue number to be used for the corresponding QDMA channel. 0h = Event *N* is queued on Q0 1h = Event *N* is queued on Q1 2h = Event *N* is queued on Q2 3h = Event *N* is queued on Q3 4h - 7h = Reserved
15	RESERVED	R	0h	Reserved
14-12	E3	R/W	0h	QDMA queue number. Contains the event queue number to be used for the corresponding QDMA channel. 0h = Event *N* is queued on Q0 1h = Event *N* is queued on Q1 2h = Event *N* is queued on Q2 3h = Event *N* is queued on Q3 4h - 7h = Reserved
11	RESERVED	R	0h	Reserved
10-8	E2	R/W	0h	QDMA queue number. Contains the event queue number to be used for the corresponding QDMA channel. 0h = Event *N* is queued on Q0 1h = Event *N* is queued on Q1 2h = Event *N* is queued on Q2 3h = Event *N* is queued on Q3 4h - 7h = Reserved
7	RESERVED	R	0h	Reserved
6-4	E1	R/W	0h	QDMA queue number. Contains the event queue number to be used for the corresponding QDMA channel. 0h = Event *N* is queued on Q0 1h = Event *N* is queued on Q1 2h = Event *N* is queued on Q2 3h = Event *N* is queued on Q3 4h - 7h = Reserved
3	RESERVED	R	0h	Reserved
2-0	E0	R/W	0h	QDMA queue number. Contains the event queue number to be used for the corresponding QDMA channel. 0h = Event *N* is queued on Q0 1h = Event *N* is queued on Q1 2h = Event *N* is queued on Q2 3h = Event *N* is queued on Q3 4h - 7h = Reserved

Table 10-45. Register Call Summary for EDMACC_QDMAQNUM

EDMA Debug/Programming Tips • Setting Up a Transfer: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_QDMAQNUM Register (Offset = 260h) [reset = 0h]: [0][1]
EDMA Prioritization • Dequeue Priority: [0]

10.6.1.1.7 EDMACC_QUETCMAP Register (Offset = 280h) [reset = 3210h]

The Queue-to-TC Mapping Register (EDMACC_QUETCMAP) defines the mapping between Event queue n and the TC number. By default, there is a one-to-one mapping between the Event queues and the Transfer Controllers. Therefore, the Transfer requests (TRs) associated with events in Q0 get submitted to TC0, TRs associated with events in Q1 get submitted to TC1, and so on. This register enables a change to the default mapping of the TC number that Event queue n TRs are submitted to. It is only used if TCs are not identical (e.g., TC0 FIFO is 512 bytes while TC1 FIFO is 1024 bytes).

CAUTION

Care must be taken while changing the default mapping in this register. User must ensure that the same TC number should never be assigned to multiple queues (i.e., TCNUMQn != TCNUMQm).

The Queue-to-TC Mapping Register (EDMACC_QUETCMAP) is shown in Figure 10-47 and described in Table 10-47.

Table 10-46. EDMACC_QUETCMAP Instances

Instance	Physical Address
EDMACC_0	0270 0280h
EDMACC_1	0272 8280h

Figure 10-47. EDMACC_QUETCMAP Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	TCNUMQ3			RESERVED	TCNUMQ2		
R-0h	R/W-3h			R-0h	R/W-2h		
7	6	5	4	3	2	1	0
RESERVED	TCNUMQ1			RESERVED	TCNUMQ0		
R-0h	R/W-1h			R-0h	R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 10-47. EDMACC_QUETCMAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14-12	TCNUMQ3	R/W	3h	TC number for queue 3. A value of 0 means Q3 TRs are submitted to TC0 and a value of 3 means Q3 TRs are submitted to TC3.
11	RESERVED	R	0h	Reserved
10-8	TCNUMQ2	R/W	2h	TC number for queue 2. A value of 0 means Q2 TRs are submitted to TC0 and a value of 3 means Q2 TRs are submitted to TC3.
7	RESERVED	R	0h	Reserved
6-4	TCNUMQ1	R/W	1h	TC number for queue 1. A value of 0 means Q1 TRs are submitted to TC0 and a value of 3 means Q1 TRs are submitted to TC3.
3	RESERVED	R	0h	Reserved
2-0	TCNUMQ0	R/W	0h	TC number for queue 0. A value of 0 means Q0 TRs are submitted to TC0 and a value of 3 means Q0 TRs are submitted to TC3.

Table 10-48. Register Call Summary for EDMACC_QUETCMAP

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_QUETCMAP Register (Offset = 280h) [reset = 3210h]: [0][1]

10.6.1.1.8 EDMACC_QUEPRI Register (Offset = 284h) [reset = 0h]

The queue priority register (EDMACC_QUEPRI) allows you to change the priority of the individual queues and the priority of the transfer request (TR) associated with the events queued in the queue. It essentially governs the priority of the associated transfer controller(s) read/write commands with respect to the other bus masters in the device. You can modify the EDMATC priority to obtain the desired system performance.

The EDMACC_QUEPRI is shown in Figure 10-48 and described in Table 10-50.

Table 10-49. EDMACC_QUEPRI Instances

Instance	Physical Address
EDMACC_0	0270 0284h
EDMACC_1	0272 8284h

Figure 10-48. EDMACC_QUEPRI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	PRIQ3			RESERVED	PRIQ2		
R-0h	R/W-0h			R-0h	R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	PRIQ1			RESERVED	PRIQ0		
R-0h	R/W-0h			R-0h	R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 10-50. EDMACC_QUEPRI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14-12	PRIQ3	R/W	0h	Priority level for queue 3. Dictates the priority level used by TC3 relative to other masters in the device. A value of 0 means highest priority and a value of 7 means lowest priority.
11	RESERVED	R	0h	Reserved
10-8	PRIQ2	R/W	0h	Priority level for queue 2. Dictates the priority level used by TC2 relative to other masters in the device. A value of 0 means highest priority and a value of 7 means lowest priority.
7	RESERVED	R	0h	Reserved
6-4	PRIQ1	R/W	0h	Priority level for queue 1. Dictates the priority level used by TC1 relative to other masters in the device. A value of 0 means highest priority and a value of 7 means lowest priority.
3	RESERVED	R	0h	Reserved
2-0	PRIQ0	R/W	0h	Priority level for queue 0. Dictates the priority level used by TC0 relative to other masters in the device. A value of 0 means highest priority and a value of 7 means lowest priority.

Table 10-51. Register Call Summary for EDMACC_QUEPRI

Event Queue(s) • Performance Considerations: [0][1][2]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]

Table 10-51. Register Call Summary for EDMACC_QUEPRI (continued)

EDMA Channel Controller (EDMACC) Registers • EDMACC_QUEPRI Register (Offset = 284h) [reset = 0h]: [0][1]
EDMA Prioritization • System (Transfer Controller) Priority: [0]

10.6.1.1.9 EDMACC_EMR Register (Offset = 300h) [reset = 0h]

For a particular DMA channel, if a second event is received prior to the first event getting cleared/serviced, the bit corresponding to that channel is set/asserted in the event missed registers (EDMACC_EMR/EDMACC_EMRH). All trigger types are treated individually, that is, manual triggered (EDMACC_ESR/EDMACC_ESRH), chain triggered (EDMACC_CER/EDMACC_CERH), and event triggered (EDMACC_ER/EDMACC_ERH) are all treated separately. The EDMACC_EMR/EDMACC_EMRH bits for a channel are also set if an event on that channel encounters a NULL entry (or a NULL TR is serviced). If any EDMACC_EMR/EDMACC_EMRH bit is set (and all errors, including bits in other error registers (EDMACC_QEMR, EDMACC_CCERR) were previously cleared), the EDMACC generates an error interrupt. See for details on EDMACC error interrupt generation.

The EDMACC_EMR is shown in Figure 10-49 and described in Table 10-53.

Table 10-52. EDMACC_EMR Instances

Instance	Physical Address
EDMACC_0	0270 0300h
EDMACC_1	0272 8300h

Figure 10-49. EDMACC_EMR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EMR31-EMR0																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-53. EDMACC_EMR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	EMR31-EMR0	R	0h	Channel 31-0 event missed. EN is cleared by writing a 1 to the corresponding bit in the event missed clear register (EDMACC_EMCR). 0h = No missed event. 1h = Missed event occurred.

Table 10-54. Register Call Summary for EDMACC_EMR

EDMA Debug/Programming Tips • Debug Checklist: [0][1]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]

Table 10-54. Register Call Summary for EDMACC_EMR (continued)

EDMA Channel Controller (EDMACC) Registers • EDMACC_EMCR Register (Offset = 308h) [reset = 0h]: [0][1][2] • EDMACC_EMCRH Register (Offset = 30Ch) [reset = 0h]: [0][1] • EDMACC_EMRH Register (Offset = 304h) [reset = 0h]: [0][1][2] • EDMACC_EMR Register (Offset = 300h) [reset = 0h]: [0][1][2][3] • EDMACC_ESR Register (Offset = 1010h) [reset = 0h]: [0][1] • EDMACC_CCERR Register (Offset = 318h) [reset = 0h]: [0] • EDMACC_ER Register (Offset = 1000h) [reset = 0h]: [0][1] • EDMACC_CER Register (Offset = 1018h) [reset = 0h]: [0][1] • EDMACC_EEVAL Register (Offset = 320h) [reset = 0h]: [0][1] • EDMACC_QEMR Register (Offset = 310h) [reset = 0h]: [0]
EDMA Interrupts • Interrupt Evaluation Operations: [0] • Error Interrupts: [0]
Parameter RAM (PaRAM) • Dummy Versus Null Transfer Comparison: [0][1] • Dummy PaRAM Set: [0][1] • Null PaRAM Set: [0][1]
Initiating a DMA Transfer • Chain-Triggered Transfer Request: [0][1] • Event-Triggered Transfer Request: [0][1] • Manually-Triggered Transfer Request: [0][1]

10.6.1.1.10 EDMACC_EMRH Register (Offset = 304h) [reset = 0h]

For a particular DMA channel, if a second event is received prior to the first event getting cleared/serviced, the bit corresponding to that channel is set/asserted in the event missed registers (EDMACC_EMR/EDMACC_EMRH). All trigger types are treated individually, that is, manual triggered (EDMACC_ESR/EDMACC_ESRH), chain triggered (EDMACC_CER/EDMACC_CERH), and event triggered (EDMACC_ER/EDMACC_ERH) are all treated separately. The EDMACC_EMR/EDMACC_EMRH bits for a channel are also set if an event on that channel encounters a NULL entry (or a NULL TR is serviced). If any EDMACC_EMR/EDMACC_EMRH bit is set (and all errors, including bits in other error registers (EDMACC_QEMR, EDMACC_CCERR) were previously cleared), the EDMACC generates an error interrupt. See for details on EDMACC error interrupt generation.

The EDMACC_EMRH is shown in Figure 10-50 and described in Table 10-56.

Table 10-55. EDMACC_EMRH Instances

Instance	Physical Address
EDMACC_0	0270 0304h
EDMACC_1	0272 8304h

Figure 10-50. EDMACC_EMRH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EMR63-EMR32																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-56. EDMACC_EMRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	EMR63-EMR32	R	0h	Channel 63-32 event missed. EN is cleared by writing a 1 to the corresponding bit in the event missed clear register high (EDMACC_EMCRH). 0h = No missed event. 1h = Missed event occurred.

Table 10-57. Register Call Summary for EDMACC_EMRH

EDMA Debug/Programming Tips • Debug Checklist: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_EMCR Register (Offset = 308h) [reset = 0h]: [0][1] • EDMACC_EMCRH Register (Offset = 30Ch) [reset = 0h]: [0][1][2] • EDMACC_EMRH Register (Offset = 304h) [reset = 0h]: [0][1][2][3] • EDMACC_EMR Register (Offset = 300h) [reset = 0h]: [0][1][2] • EDMACC_ESR Register (Offset = 1010h) [reset = 0h]: [0][1] • EDMACC_CCERR Register (Offset = 318h) [reset = 0h]: [0] • EDMACC_ER Register (Offset = 1000h) [reset = 0h]: [0][1] • EDMACC_CER Register (Offset = 1018h) [reset = 0h]: [0][1] • EDMACC_EEVAL Register (Offset = 320h) [reset = 0h]: [0][1] • EDMACC_QEMR Register (Offset = 310h) [reset = 0h]: [0]
EDMA Interrupts • Interrupt Evaluation Operations: [0] • Error Interrupts: [0]
Parameter RAM (PaRAM) • Dummy Versus Null Transfer Comparison: [0] • Dummy PaRAM Set: [0] • Null PaRAM Set: [0]

10.6.1.1.11 EDMACC_EMCR Register (Offset = 308h) [reset = 0h]

Once a missed event is posted in the event missed registers (EDMACC_EMR/EDMACC_EMRH), the bit remains set and you need to clear the set bit(s). This is done by way of DSP writes to the event missed clear registers (EDMACC_EMCR/EDMACC_EMCRH). Writing a 1 to any of the bits clears the corresponding missed event (bit) in EDMACC_EMR/EDMACC_EMRH; writing a 0 has no effect.

The EDMACC_EMCR is shown in Figure 10-51 and described in Table 10-59.

Table 10-58. EDMACC_EMCR Instances

Instance	Physical Address
EDMACC_0	0270 0308h
EDMACC_1	0272 8308h

Figure 10-51. EDMACC_EMCR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EMCR31-EMCR0																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 10-59. EDMACC_EMCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	EMCR31-EMCR0	W	0h	Event missed 31-0 clear. All error bits must be cleared before additional error interrupts will be asserted by the EDMACC 0h = No effect. 1h = Corresponding missed event bit in the event missed register (EDMACC_EMR) is cleared.

Table 10-60. Register Call Summary for EDMACC_EMCR

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_EMCR Register (Offset = 308h) [reset = 0h]: [0][1] • EDMACC_EMCRH Register (Offset = 30Ch) [reset = 0h]: [0] • EDMACC_EMR Register (Offset = 300h) [reset = 0h]: [0]

10.6.1.1.12 EDMACC_EMCRH Register (Offset = 30Ch) [reset = 0h]

Once a missed event is posted in the event missed registers (EDMACC_EMR/EDMACC_EMRH), the bit remains set and you need to clear the set bit(s). This is done by way of DSP writes to the event missed clear registers (EDMACC_EMCR/EDMACC_EMCRH). Writing a 1 to any of the bits clears the corresponding missed event (bit) in EDMACC_EMR/EDMACC_EMRH; writing a 0 has no effect.

The EDMACC_EMCRH is shown in Figure 10-52 and described in Table 10-62.

Table 10-61. EDMACC_EMCRH Instances

Instance	Physical Address
EDMACC_0	0270 030Ch
EDMACC_1	0272 830Ch

Figure 10-52. EDMACC_EMCRH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EMCR63-EMCR32																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 10-62. EDMACC_EMCRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	EMCR63-EMCR32	W	0h	Event missed 32-63 clear. All error bits must be cleared before additional error interrupts will be asserted by the EDMACC. 0h = No effect. 1h = Corresponding missed event bit in the event missed register high (EDMACC_EMRH) is cleared.

Table 10-63. Register Call Summary for EDMACC_EMCRH

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_EMCR Register (Offset = 308h) [reset = 0h]: [0] • EDMACC_EMCRH Register (Offset = 30Ch) [reset = 0h]: [0][1] • EDMACC_EMRH Register (Offset = 304h) [reset = 0h]: [0]

10.6.1.1.13 EDMACC_QEMR Register (Offset = 310h) [reset = 0h]

For a particular QDMA channel, if two QDMA events are detected without the first event getting cleared/serviced, the bit corresponding to that channel is set/asserted in the QDMA event missed register (EDMACC_QEMR). The EDMACC_QEMR bits for a channel are also set if a QDMA event on the channel encounters a NULL entry (or a NULL TR is serviced). If any EDMACC_QEMR bit is set (and all errors, including bits in other error registers (EDMACC_EMR/EDMACC_EMRH, EDMACC_CCERR) were previously cleared), the EDMACC generates an error interrupt. See for details on EDMACC error interrupt generation.

The EDMACC_QEMR is shown in Figure 10-53 and described in Table 10-65.

Table 10-64. EDMACC_QEMR Instances

Instance	Physical Address
EDMACC_0	0270 0310h
EDMACC_1	0272 8310h

Figure 10-53. EDMACC_QEMR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								QEMR7-QEMR0							
R-0h																								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 10-65. EDMACC_QEMR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	QEMR7-QEMR0	R	0h	Channel 7-0 QDMA event missed. 0h = No missed event. 1h = Missed event occurred.

Table 10-66. Register Call Summary for EDMACC_QEMR

EDMA Debug/Programming Tips • Debug Checklist: [0][1]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_QER Register (Offset = 1080h) [reset = 0h]: [0][1] • EDMACC_EMRH Register (Offset = 304h) [reset = 0h]: [0] • EDMACC_QEMCR Register (Offset = 314h) [reset = 0h]: [0][1][2] • EDMACC_EMR Register (Offset = 300h) [reset = 0h]: [0] • EDMACC_CCERR Register (Offset = 318h) [reset = 0h]: [0] • EDMACC_EEVAL Register (Offset = 320h) [reset = 0h]: [0][1] • EDMACC_QEMR Register (Offset = 310h) [reset = 0h]: [0][1][2][3]
EDMA Interrupts • Interrupt Evaluation Operations: [0] • Error Interrupts: [0]
Parameter RAM (PaRAM) • Dummy Versus Null Transfer Comparison: [0] • Dummy PaRAM Set: [0] • Null PaRAM Set: [0]
Initiating a DMA Transfer • Auto-triggered and Link-Triggered Transfer Request: [0]

10.6.1.1.14 EDMACC_QEMCR Register (Offset = 314h) [reset = 0h]

Once a missed event is posted in the QDMA event missed registers (EDMACC_QEMR), the bit remains set and you need to clear the set bit(s). This is done by way of DSP writes to the QDMA event missed clear registers (EDMACC_QEMCR). Writing a 1 to any of the bits clears the corresponding missed event (bit) in EDMACC_QEMR; writing a 0 has no effect.

The EDMACC_QEMCR is shown in Figure 10-54 and described in Table 10-68.

Table 10-67. EDMACC_QEMCR Instances

Instance	Physical Address
EDMACC_0	0270 0314h
EDMACC_1	0272 8314h

Figure 10-54. EDMACC_QEMCR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								QEMCR7-QEMCR0							
R-0h																								W-0h							

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 10-68. EDMACC_QEMCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	QEMCR7-QEMCR0	W	0h	QDMA event missed clear. All error bits must be cleared before additional error interrupts will be asserted by the EDMACC. 0h = No effect. 1h = Corresponding missed event bit in the QDMA event missed register (EDMACC_QEMR) is cleared.

Table 10-69. Register Call Summary for EDMACC_QEMCR

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_QEMCR Register (Offset = 314h) [reset = 0h]: [0][1]

10.6.1.1.15 EDMACC_CCERR Register (Offset = 318h) [reset = 0h]

The EDMACC error register (EDMACC_CCERR) indicates whether or not at any instant of time the number of events queued up in any of the event queues exceeds or equals the threshold/watermark value that is set in the queue watermark threshold register (EDMACC_QWMTHRA). Additionally, EDMACC_CCERR also indicates if when the number of outstanding TRs that have been programmed to return transfer completion code (TRs which have the TCINTEN or TCCHEN bit in OPT set) to the EDMACC has exceeded the maximum allowed value of 63. If any bit in EDMACC_CCERR is set (and all errors, including bits in other error registers (EDMACC_EMR/EDMACC_EMRH, EDMACC_QEMR) were previously cleared), the EDMACC generates an error interrupt. See for details on EDMACC error interrupt generation. Once the error bits are set in EDMACC_CCERR, they can only be cleared by writing to the corresponding bits in the EDMACC error clear register (EDMACC_CCERRCLR).

The EDMACC_CCERR is shown in Figure 10-55 and described in Table 10-71.

Table 10-70. EDMACC_CCERR Instances

Instance	Physical Address
EDMACC_0	0270 0318h
EDMACC_1	0272 8318h

Figure 10-55. EDMACC_CCERR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							TCCERR
R-0h							R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				QTHRXCD3	QTHRXCD2	QTHRXCD1	QTHRXCD0
R-0h				R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 10-71. EDMACC_CCERR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	TCCERR	R	0h	Transfer completion code error. TCCERR is cleared by writing a 1 to the corresponding bit in the EDMACC error clear register (EDMACC_CCERRCLR). 0h = Total number of allowed TCCs outstanding has not been reached. 1h = Total number of allowed TCCs has been reached.
15-4	RESERVED	R	0h	Reserved
3	QTHRXCD3	R	0h	Queue threshold error for queue 3. QTHRXCD3 is cleared by writing a 1 to the corresponding bit in the EDMACC error clear register (EDMACC_CCERRCLR). 0h = Watermark/threshold has not been exceeded. 1h = Watermark/threshold has been exceeded.
2	QTHRXCD2	R	0h	Queue threshold error for queue 2. QTHRXCD2 is cleared by writing a 1 to the corresponding bit in the EDMACC error clear register (EDMACC_CCERRCLR). 0h = Watermark/threshold has not been exceeded. 1h = Watermark/threshold has been exceeded.

Table 10-71. EDMACC_CCERR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	QTHRXCD1	R	0h	Queue threshold error for queue 1. QTHRXCD1 is cleared by writing a 1 to the corresponding bit in the EDMACC error clear register (EDMACC_CCERRCLR). 0h = Watermark/threshold has not been exceeded. 1h = Watermark/threshold has been exceeded.
0	QTHRXCD0	R	0h	Queue threshold error for queue 0. QTHRXCD0 is cleared by writing a 1 to the corresponding bit in the EDMACC error clear register (EDMACC_CCERRCLR). 0h = Watermark/threshold has not been exceeded. 1h = Watermark/threshold has been exceeded.

Table 10-72. Register Call Summary for EDMACC_CCERR

Event Queue(s) • Queue Resource Tracking: [0][1]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_CCERRCLR Register (Offset = 31Ch) [reset = 0h]: [0][1][2][3][4][5][6] • EDMACC_QWMTHRA Register (Offset = 620h) [reset = 2020202h]: [0][1][2][3] • EDMACC_EMRH Register (Offset = 304h) [reset = 0h]: [0] • EDMACC_EMR Register (Offset = 300h) [reset = 0h]: [0] • EDMACC_CCERR Register (Offset = 318h) [reset = 0h]: [0][1][2][3][4] • EDMACC_EEVAL Register (Offset = 320h) [reset = 0h]: [0][1] • EDMACC_QEMR Register (Offset = 310h) [reset = 0h]: [0]
EDMA Interrupts • Interrupt Evaluation Operations: [0] • Error Interrupts: [0][1]

10.6.1.1.16 EDMACC_CCERRCLR Register (Offset = 31Ch) [reset = 0h]

The EDMACC error clear register (EDMACC_CCERRCLR) is used to clear any error bits that are set in the EDMACC error register (EDMACC_CCERR). In addition, EDMACC_CCERRCLR also clears the values of some bit fields in the queue status registers (QSTAT *n*) associated with a particular event queue. Writing a 1 to any of the bits clears the corresponding bit in EDMACC_CCERR; writing a 0 has no effect.

The EDMACC_CCERRCLR is shown in Figure 10-56 and described in Table 10-74.

Table 10-73. EDMACC_CCERRCLR Instances

Instance	Physical Address
EDMACC_0	0270 031Ch
EDMACC_1	0272 831Ch

Figure 10-56. EDMACC_CCERRCLR Register

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							TCCERR
W-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED				QTHRXCD3	QTHRXCD2	QTHRXCD1	QTHRXCD0
W-0h				W-0h	W-0h	W-0h	W-0h

LEGEND: W = Write Only; *-n* = value after reset

Table 10-74. EDMACC_CCERRCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	W	0h	Reserved
16	TCCERR	W	0h	Transfer completion code error clear 0h = No effect. 1h = Clears the TCCERR bit in the EDMACC error register (EDMACC_CCERR).
15-4	RESERVED	W	0h	Reserved
3	QTHRXCD3	W	0h	Queue threshold error clear for queue 3. 0h = No effect. 1h = Clears the QTHRXCD3 bit in the EDMACC error register (EDMACC_CCERR) and the WM and THRXCD bits in the queue status register 3 (EDMACC_QSTAT3).
2	QTHRXCD2	W	0h	Queue threshold error clear for queue 2. 0h = No effect. 1h = Clears the QTHRXCD2 bit in the EDMACC error register (EDMACC_CCERR) and the WM and THRXCD bits in the queue status register 2 (QSTAT2).
1	QTHRXCD1	W	0h	Queue threshold error clear for queue 1. 0h = No effect. 1h = Clears the QTHRXCD1 bit in the EDMACC error register (EDMACC_CCERR) and the WM and THRXCD bits in the queue status register 1 (QSTAT1).

Table 10-74. EDMACC_CCERRCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	QTHRXCD0	W	0h	Queue threshold error clear for queue 0. 0h = No effect. 1h = Clears the QTHRXCD0 bit in the EDMACC error register (EDMACC_CCERR) and the WM and THRXCD bits in the queue status register 0 (EDMACC_QSTAT0).

Table 10-75. Register Call Summary for EDMACC_CCERRCLR

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_CCERRCLR Register (Offset = 31Ch) [reset = 0h]: [0][1][2] • EDMACC_QSTAT0 to EDMACC_QSTAT3 Register (Offset = 600h to 60Ch) [reset = 0h]: [0][1] • EDMACC_CCERR Register (Offset = 318h) [reset = 0h]: [0][1][2][3][4][5]

10.6.1.1.17 EDMACC_EEVAL Register (Offset = 320h) [reset = 0h]

The EDMACC error interrupt is asserted whenever an error bit is set in any of the error registers (EDMACC_EMR/EDMACC_EMRH, EDMACC_QEMR, and EDMACC_CCERR). For subsequent error bits that get set, the EDMACC error interrupt is reasserted only when transitioning from an "all the error bits cleared" to "at least one error bit is set". Alternatively, a DSP write of 1 to the EVAL bit in the error evaluation register (EDMACC_EEVAL) results in reasserting the EDMACC error interrupt, if there are any outstanding error bits set due to subsequent error conditions. Writes of 0 have no effect.

The EDMACC_EEVAL is shown in Figure 10-57 and described in Table 10-77.

Table 10-76. EDMACC_EEVAL Instances

Instance	Physical Address
EDMACC_0	0270 0320h
EDMACC_1	0272 8320h

Figure 10-57. EDMACC_EEVAL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						SET	EVAL
R-0h						W-0h	W-0h

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 10-77. EDMACC_EEVAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	SET	W	0h	Always write 0 to this bit; writes of 1 to this bit are not supported and attempts to do so may result in undefined behavior.
0	EVAL	W	0h	Error interrupt evaluate. 0h = No effect. 1h = EDMACC error interrupt will be pulsed if any errors have not been cleared in any of the error registers (EDMACC_EMR/EDMACC_EMRH, EDMACC_QEMR, or EDMACC_CCERR).

Table 10-78. Register Call Summary for EDMACC_EEVAL

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_EEVAL Register (Offset = 320h) [reset = 0h]: [0][1]
EDMA Interrupts • Interrupt Evaluation Operations: [0][1] • Error Interrupts: [0]

10.6.1.2 Region Access Enable Registers

The region access enable register group consists of the DMA access enable registers (DRAE *m* and DRAEH *m*) and the QDMA access enable registers (QRAE *m*). Where *m* is the number of shadow regions in the EDMACC memory map for a device. You can configure these registers to assign ownership of DMA/QDMA channels to a particular shadow region.

10.6.1.2.1 EDMACC_DRAE0 Register (Offset = 340h) [reset = 0h]

The DMA region access enable register for shadow region *0* (EDMACC_DRAE0/EDMACC_DRAEH0) is programmed to allow or disallow read/write accesses on a bit-by-bit bases for all DMA registers in the shadow region *0* view of the DMA channel registers. See the EDMACC register memory map (Table 10-27) for a list of all the DMA channel and interrupt registers mapped in the shadow region view. Additionally, the EDMACC_DRAE0/EDMACC_DRAEH0 configuration determines completion of which DMA channels will result in assertion of the shadow region *0* DMA completion interrupt (see).

The EDMACC_DRAE0 is shown in Figure 10-58 and described in Table 10-80.

Table 10-79. EDMACC_DRAE0 Instances

Instance	Physical Address
EDMACC_0	0270 0340h
EDMACC_1	0272 8340h

Figure 10-58. EDMACC_DRAE0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRAE31-DRAE0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-80. EDMACC_DRAE0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DRAE31-DRAE0	R/W	0h	DMA region access enable for bit N/channel N in region *0*. 0h = Accesses via region m address space to bit N in any DMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of a transfer completion interrupt for shadow region *0*. 1h = Accesses via region *0* address space to bit N in any DMA channel register are allowed. Reads return the value from bit N and writes modify the state of bit N. Enabled interrupt bits for bit N contribute to the generation of a transfer completion interrupt for shadow region *0*.

Table 10-81. Register Call Summary for EDMACC_DRAE0

EDMA Debug/Programming Tips • Miscellaneous Programming/Debug Tips: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_DRAE0 Register (Offset = 340h) [reset = 0h]: [0][1][2]
EDMA Interrupts • Enabling Transfer Completion Interrupts: [0][1]

10.6.1.2.2 EDMACC_DRAEH0 Register (Offset = 344h) [reset = 0h]

The EDMACC_DRAEH0 is shown in Figure 10-59 and described in Table 10-83.

Table 10-82. EDMACC_DRAEH0 Instances

Instance	Physical Address
EDMACC_0	0270 0344h
EDMACC_1	0272 8344h

Figure 10-59. EDMACC_DRAEH0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRAEH31-DRAEH0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-83. EDMACC_DRAEH0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DRAEH31-DRAEH0	R/W	0h	DMA region access enable for bit N/channel N in region *0*. 0h = Accesses via region m address space to bit N in any DMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of a transfer completion interrupt for shadow region *0*. 1h = Accesses via region *0* address space to bit N in any DMA channel register are allowed. Reads return the value from bit N and writes modify the state of bit N. Enabled interrupt bits for bit N contribute to the generation of a transfer completion interrupt for shadow region *0*.

Table 10-84. Register Call Summary for EDMACC_DRAEH0

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_DRAE0 Register (Offset = 340h) [reset = 0h]: [0][1] • EDMACC_DRAEH0 Register (Offset = 344h) [reset = 0h]: [0]

10.6.1.2.3 EDMACC_DRAE1 Register (Offset = 348h) [reset = 0h]

The DMA region access enable register for shadow region *1* (EDMACC_DRAE1/EDMACC_DRAEH1) is programmed to allow or disallow read/write accesses on a bit-by-bit bases for all DMA registers in the shadow region *1* view of the DMA channel registers. See the EDMACC register memory map (Table 10-27) for a list of all the DMA channel and interrupt registers mapped in the shadow region view. Additionally, the EDMACC_DRAE1/EDMACC_DRAEH1 configuration determines completion of which DMA channels will result in assertion of the shadow region *1* DMA completion interrupt (see).

The EDMACC_DRAE1 is shown in Figure 10-60 and described in Table 10-86.

Table 10-85. EDMACC_DRAE1 Instances

Instance	Physical Address
EDMACC_0	0270 0348h
EDMACC_1	0272 8348h

Figure 10-60. EDMACC_DRAE1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRAE31-DRAE0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-86. EDMACC_DRAE1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DRAE31-DRAE0	R/W	0h	DMA region access enable for bit N/channel N in region *1*. 0h = Accesses via region m address space to bit N in any DMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of a transfer completion interrupt for shadow region *1*. 1h = Accesses via region *1* address space to bit N in any DMA channel register are allowed. Reads return the value from bit N and writes modify the state of bit N. Enabled interrupt bits for bit N contribute to the generation of a transfer completion interrupt for shadow region *1*.

Table 10-87. Register Call Summary for EDMACC_DRAE1

EDMA Debug/Programming Tips • Miscellaneous Programming/Debug Tips: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_DRAE1 Register (Offset = 348h) [reset = 0h]: [0][1][2]
EDMA Interrupts • Enabling Transfer Completion Interrupts: [0][1]

10.6.1.2.4 EDMACC_DRAEH1 Register (Offset = 34Ch) [reset = 0h]

The EDMACC_DRAEH1 is shown in Figure 10-61 and described in Table 10-89.

Table 10-88. EDMACC_DRAEH1 Instances

Instance	Physical Address
EDMACC_0	0270 034Ch
EDMACC_1	0272 834Ch

Figure 10-61. EDMACC_DRAEH1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRAEH31-DRAEH0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-89. EDMACC_DRAEH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DRAEH31-DRAEH0	R/W	0h	DMA region access enable for bit N/channel N in region *1*. 0h = Accesses via region m address space to bit N in any DMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of a transfer completion interrupt for shadow region *1*. 1h = Accesses via region *1* address space to bit N in any DMA channel register are allowed. Reads return the value from bit N and writes modify the state of bit N. Enabled interrupt bits for bit N contribute to the generation of a transfer completion interrupt for shadow region *1*.

Table 10-90. Register Call Summary for EDMACC_DRAEH1

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_DRAE1 Register (Offset = 348h) [reset = 0h]: [0][1] • EDMACC_DRAEH1 Register (Offset = 34Ch) [reset = 0h]: [0]

10.6.1.2.5 EDMACC_DRAE2 Register (Offset = 350h) [reset = 0h]

The DMA region access enable register for shadow region *2* (EDMACC_DRAE2/EDMACC_DRAEH2) is programmed to allow or disallow read/write accesses on a bit-by-bit bases for all DMA registers in the shadow region *2* view of the DMA channel registers. See the EDMACC register memory map (Table 10-27) for a list of all the DMA channel and interrupt registers mapped in the shadow region view. Additionally, the EDMACC_DRAE2/EDMACC_DRAEH2 configuration determines completion of which DMA channels will result in assertion of the shadow region *2* DMA completion interrupt (see).

The EDMACC_DRAE2 is shown in Figure 10-62 and described in Table 10-92.

Table 10-91. EDMACC_DRAE2 Instances

Instance	Physical Address
EDMACC_0	0270 0350h
EDMACC_1	0272 8350h

Figure 10-62. EDMACC_DRAE2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRAE31-DRAE0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-92. EDMACC_DRAE2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DRAE31-DRAE0	R/W	0h	DMA region access enable for bit N/channel N in region *2*. 0h = Accesses via region m address space to bit N in any DMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of a transfer completion interrupt for shadow region *2*. 1h = Accesses via region *2* address space to bit N in any DMA channel register are allowed. Reads return the value from bit N and writes modify the state of bit N. Enabled interrupt bits for bit N contribute to the generation of a transfer completion interrupt for shadow region *2*.

Table 10-93. Register Call Summary for EDMACC_DRAE2

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_DRAE2 Register (Offset = 350h) [reset = 0h]: [0][1][2]
EDMA Interrupts • Enabling Transfer Completion Interrupts: [0][1]

10.6.1.2.6 EDMACC_DRAEH2 Register (Offset = 354h) [reset = 0h]

The EDMACC_DRAEH2 is shown in Figure 10-63 and described in Table 10-95.

Table 10-94. EDMACC_DRAEH2 Instances

Instance	Physical Address
EDMACC_0	0270 0354h
EDMACC_1	0272 8354h

Figure 10-63. EDMACC_DRAEH2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRAEH31-DRAEH0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-95. EDMACC_DRAEH2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DRAEH31-DRAEH0	R/W	0h	DMA region access enable for bit N/channel N in region *2*. 0h = Accesses via region m address space to bit N in any DMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of a transfer completion interrupt for shadow region *2*. 1h = Accesses via region *2* address space to bit N in any DMA channel register are allowed. Reads return the value from bit N and writes modify the state of bit N. Enabled interrupt bits for bit N contribute to the generation of a transfer completion interrupt for shadow region *2*.

Table 10-96. Register Call Summary for EDMACC_DRAEH2

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_DRAEH2 Register (Offset = 354h) [reset = 0h]: [0] • EDMACC_DRAE2 Register (Offset = 350h) [reset = 0h]: [0][1]

10.6.1.2.7 EDMACC_DRAE3 Register (Offset = 358h) [reset = 0h]

The DMA region access enable register for shadow region *3* (EDMACC_DRAE3/EDMACC_DRAEH3) is programmed to allow or disallow read/write accesses on a bit-by-bit bases for all DMA registers in the shadow region *3* view of the DMA channel registers. See the EDMACC register memory map (Table 10-27) for a list of all the DMA channel and interrupt registers mapped in the shadow region view. Additionally, the EDMACC_DRAE3/EDMACC_DRAEH3 configuration determines completion of which DMA channels will result in assertion of the shadow region *3* DMA completion interrupt (see).

The EDMACC_DRAE3 is shown in Figure 10-64 and described in Table 10-98.

Table 10-97. EDMACC_DRAE3 Instances

Instance	Physical Address
EDMACC_0	0270 0358h
EDMACC_1	0272 8358h

Figure 10-64. EDMACC_DRAE3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRAE31-DRAE0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-98. EDMACC_DRAE3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DRAE31-DRAE0	R/W	0h	DMA region access enable for bit N/channel N in region *3*. 0h = Accesses via region m address space to bit N in any DMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of a transfer completion interrupt for shadow region *3*. 1h = Accesses via region *3* address space to bit N in any DMA channel register are allowed. Reads return the value from bit N and writes modify the state of bit N. Enabled interrupt bits for bit N contribute to the generation of a transfer completion interrupt for shadow region *3*.

Table 10-99. Register Call Summary for EDMACC_DRAE3

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_DRAE3 Register (Offset = 358h) [reset = 0h]: [0][1][2]

10.6.1.2.8 EDMACC_DRAEH3 Register (Offset = 35Ch) [reset = 0h]

The EDMACC_DRAEH3 is shown in Figure 10-65 and described in Table 10-101.

Table 10-100. EDMACC_DRAEH3 Instances

Instance	Physical Address
EDMACC_0	0270 035Ch
EDMACC_1	0272 835Ch

Figure 10-65. EDMACC_DRAEH3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRAEH31-DRAEH0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-101. EDMACC_DRAEH3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DRAEH31-DRAEH0	R/W	0h	DMA region access enable for bit N/channel N in region *3*. 0h = Accesses via region m address space to bit N in any DMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of a transfer completion interrupt for shadow region *3*. 1h = Accesses via region *3* address space to bit N in any DMA channel register are allowed. Reads return the value from bit N and writes modify the state of bit N. Enabled interrupt bits for bit N contribute to the generation of a transfer completion interrupt for shadow region *3*.

Table 10-102. Register Call Summary for EDMACC_DRAEH3

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_DRAEH3 Register (Offset = 35Ch) [reset = 0h]: [0] • EDMACC_DRAE3 Register (Offset = 358h) [reset = 0h]: [0][1]

10.6.1.2.9 EDMACC_DRAE4 Register (Offset = 360h) [reset = 0h]

The DMA region access enable register for shadow region *4* (EDMACC_DRAE4/EDMACC_DRAEH4) is programmed to allow or disallow read/write accesses on a bit-by-bit bases for all DMA registers in the shadow region *4* view of the DMA channel registers. See the EDMACC register memory map (Table 10-27) for a list of all the DMA channel and interrupt registers mapped in the shadow region view. Additionally, the EDMACC_DRAE4/EDMACC_DRAEH4 configuration determines completion of which DMA channels will result in assertion of the shadow region *4* DMA completion interrupt (see).

The EDMACC_DRAE4 is shown in Figure 10-66 and described in Table 10-104.

Table 10-103. EDMACC_DRAE4 Instances

Instance	Physical Address
EDMACC_0	0270 0360h
EDMACC_1	0272 8360h

Figure 10-66. EDMACC_DRAE4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRAE31-DRAE0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-104. EDMACC_DRAE4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DRAE31-DRAE0	R/W	0h	DMA region access enable for bit N/channel N in region *4*. 0h = Accesses via region m address space to bit N in any DMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of a transfer completion interrupt for shadow region *4*. 1h = Accesses via region *4* address space to bit N in any DMA channel register are allowed. Reads return the value from bit N and writes modify the state of bit N. Enabled interrupt bits for bit N contribute to the generation of a transfer completion interrupt for shadow region *4*.

Table 10-105. Register Call Summary for EDMACC_DRAE4

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_DRAE4 Register (Offset = 360h) [reset = 0h]: [0][1][2]

10.6.1.2.10 EDMACC_DRAEH4 Register (Offset = 364h) [reset = 0h]

The EDMACC_DRAEH4 is shown in Figure 10-67 and described in Table 10-107.

Table 10-106. EDMACC_DRAEH4 Instances

Instance	Physical Address
EDMACC_0	0270 0364h
EDMACC_1	0272 8364h

Figure 10-67. EDMACC_DRAEH4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRAEH31-DRAEH0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-107. EDMACC_DRAEH4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DRAEH31-DRAEH0	R/W	0h	DMA region access enable for bit N/channel N in region *4*. 0h = Accesses via region m address space to bit N in any DMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of a transfer completion interrupt for shadow region *4*. 1h = Accesses via region *4* address space to bit N in any DMA channel register are allowed. Reads return the value from bit N and writes modify the state of bit N. Enabled interrupt bits for bit N contribute to the generation of a transfer completion interrupt for shadow region *4*.

Table 10-108. Register Call Summary for EDMACC_DRAEH4

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_DRAE4 Register (Offset = 360h) [reset = 0h]: [0][1] • EDMACC_DRAEH4 Register (Offset = 364h) [reset = 0h]: [0]

10.6.1.2.11 EDMACC_DRAE5 Register (Offset = 368h) [reset = 0h]

The DMA region access enable register for shadow region *5* (EDMACC_DRAE5/EDMACC_DRAEH5) is programmed to allow or disallow read/write accesses on a bit-by-bit bases for all DMA registers in the shadow region *5* view of the DMA channel registers. See the EDMACC register memory map (Table 10-27) for a list of all the DMA channel and interrupt registers mapped in the shadow region view. Additionally, the EDMACC_DRAE5/EDMACC_DRAEH5 configuration determines completion of which DMA channels will result in assertion of the shadow region *5* DMA completion interrupt (see).

The EDMACC_DRAE5 is shown in Figure 10-68 and described in Table 10-110.

Table 10-109. EDMACC_DRAE5 Instances

Instance	Physical Address
EDMACC_0	0270 0368h
EDMACC_1	0272 8368h

Figure 10-68. EDMACC_DRAE5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRAE31-DRAE0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-110. EDMACC_DRAE5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DRAE31-DRAE0	R/W	0h	DMA region access enable for bit N/channel N in region *5*. 0h = Accesses via region m address space to bit N in any DMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of a transfer completion interrupt for shadow region *5*. 1h = Accesses via region *5* address space to bit N in any DMA channel register are allowed. Reads return the value from bit N and writes modify the state of bit N. Enabled interrupt bits for bit N contribute to the generation of a transfer completion interrupt for shadow region *5*.

Table 10-111. Register Call Summary for EDMACC_DRAE5

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_DRAE5 Register (Offset = 368h) [reset = 0h]: [0][1][2]

10.6.1.2.12 EDMACC_DRAEH5 Register (Offset = 36Ch) [reset = 0h]

The EDMACC_DRAEH5 is shown in Figure 10-69 and described in Table 10-113.

Table 10-112. EDMACC_DRAEH5 Instances

Instance	Physical Address
EDMACC_0	0270 036Ch
EDMACC_1	0272 836Ch

Figure 10-69. EDMACC_DRAEH5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRAEH31-DRAEH0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-113. EDMACC_DRAEH5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DRAEH31-DRAEH0	R/W	0h	DMA region access enable for bit N/channel N in region *5*. 0h = Accesses via region m address space to bit N in any DMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of a transfer completion interrupt for shadow region *5*. 1h = Accesses via region *5* address space to bit N in any DMA channel register are allowed. Reads return the value from bit N and writes modify the state of bit N. Enabled interrupt bits for bit N contribute to the generation of a transfer completion interrupt for shadow region *5*.

Table 10-114. Register Call Summary for EDMACC_DRAEH5

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_DRAE5 Register (Offset = 368h) [reset = 0h]: [0][1] • EDMACC_DRAEH5 Register (Offset = 36Ch) [reset = 0h]: [0]

10.6.1.2.13 EDMACC_DRAE6 Register (Offset = 370h) [reset = 0h]

The DMA region access enable register for shadow region *6* (EDMACC_DRAE6/EDMACC_DRAEH6) is programmed to allow or disallow read/write accesses on a bit-by-bit bases for all DMA registers in the shadow region *6* view of the DMA channel registers. See the EDMACC register memory map (Table 10-27) for a list of all the DMA channel and interrupt registers mapped in the shadow region view. Additionally, the EDMACC_DRAE6/EDMACC_DRAEH6 configuration determines completion of which DMA channels will result in assertion of the shadow region *6* DMA completion interrupt (see).

The EDMACC_DRAE6 is shown in Figure 10-70 and described in Table 10-116.

Table 10-115. EDMACC_DRAE6 Instances

Instance	Physical Address
EDMACC_0	0270 0370h
EDMACC_1	0272 8370h

Figure 10-70. EDMACC_DRAE6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRAE31-DRAE0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-116. EDMACC_DRAE6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DRAE31-DRAE0	R/W	0h	DMA region access enable for bit N/channel N in region *6*. 0h = Accesses via region m address space to bit N in any DMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of a transfer completion interrupt for shadow region *6*. 1h = Accesses via region *6* address space to bit N in any DMA channel register are allowed. Reads return the value from bit N and writes modify the state of bit N. Enabled interrupt bits for bit N contribute to the generation of a transfer completion interrupt for shadow region *6*.

Table 10-117. Register Call Summary for EDMACC_DRAE6

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_DRAE6 Register (Offset = 370h) [reset = 0h]: [0][1][2]

10.6.1.2.14 EDMACC_DRAEH6 Register (Offset = 374h) [reset = 0h]

The EDMACC_DRAEH6 is shown in Figure 10-71 and described in Table 10-119.

Table 10-118. EDMACC_DRAEH6 Instances

Instance	Physical Address
EDMACC_0	0270 0374h
EDMACC_1	0272 8374h

Figure 10-71. EDMACC_DRAEH6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRAEH31-DRAEH0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-119. EDMACC_DRAEH6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DRAEH31-DRAEH0	R/W	0h	DMA region access enable for bit N/channel N in region *6*. 0h = Accesses via region m address space to bit N in any DMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of a transfer completion interrupt for shadow region *6*. 1h = Accesses via region *6* address space to bit N in any DMA channel register are allowed. Reads return the value from bit N and writes modify the state of bit N. Enabled interrupt bits for bit N contribute to the generation of a transfer completion interrupt for shadow region *6*.

Table 10-120. Register Call Summary for EDMACC_DRAEH6

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_DRAEH6 Register (Offset = 374h) [reset = 0h]: [0] • EDMACC_DRAE6 Register (Offset = 370h) [reset = 0h]: [0][1]

10.6.1.2.15 EDMACC_DRAE7 Register (Offset = 378h) [reset = 0h]

The DMA region access enable register for shadow region *7* (EDMACC_DRAE7/EDMACC_DRAEH7) is programmed to allow or disallow read/write accesses on a bit-by-bit bases for all DMA registers in the shadow region *7* view of the DMA channel registers. See the EDMACC register memory map (Table 10-27) for a list of all the DMA channel and interrupt registers mapped in the shadow region view. Additionally, the EDMACC_DRAE7/EDMACC_DRAEH7 configuration determines completion of which DMA channels will result in assertion of the shadow region *7* DMA completion interrupt (see).

The EDMACC_DRAE7 is shown in Figure 10-72 and described in Table 10-122.

Table 10-121. EDMACC_DRAE7 Instances

Instance	Physical Address
EDMACC_0	0270 0378h
EDMACC_1	0272 8378h

Figure 10-72. EDMACC_DRAE7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRAE31-DRAE0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-122. EDMACC_DRAE7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DRAE31-DRAE0	R/W	0h	DMA region access enable for bit N/channel N in region *7*. 0h = Accesses via region m address space to bit N in any DMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of a transfer completion interrupt for shadow region *7*. 1h = Accesses via region *7* address space to bit N in any DMA channel register are allowed. Reads return the value from bit N and writes modify the state of bit N. Enabled interrupt bits for bit N contribute to the generation of a transfer completion interrupt for shadow region *7*.

Table 10-123. Register Call Summary for EDMACC_DRAE7

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_DRAE7 Register (Offset = 378h) [reset = 0h]: [0][1][2]
EDMA Interrupts • Enabling Transfer Completion Interrupts: [0][1]

10.6.1.2.16 EDMACC_DRAEH7 Register (Offset = 37Ch) [reset = 0h]

The EDMACC_DRAEH7 is shown in Figure 10-73 and described in Table 10-125.

Table 10-124. EDMACC_DRAEH7 Instances

Instance	Physical Address
EDMACC_0	0270 037Ch
EDMACC_1	0272 837Ch

Figure 10-73. EDMACC_DRAEH7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRAEH31-DRAEH0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 10-125. EDMACC_DRAEH7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DRAEH31-DRAEH0	R/W	0h	DMA region access enable for bit N/channel N in region *7*. 0h = Accesses via region m address space to bit N in any DMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of a transfer completion interrupt for shadow region *7*. 1h = Accesses via region *7* address space to bit N in any DMA channel register are allowed. Reads return the value from bit N and writes modify the state of bit N. Enabled interrupt bits for bit N contribute to the generation of a transfer completion interrupt for shadow region *7*.

Table 10-126. Register Call Summary for EDMACC_DRAEH7

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_DRAEH7 Register (Offset = 37Ch) [reset = 0h]: [0] • EDMACC_DRAE7 Register (Offset = 378h) [reset = 0h]: [0][1]

10.6.1.2.17 EDMACC_QRAE0 to EDMACC_QRAE7 Register (Offset = 380h to 39Ch) [reset = 0h]

The QDMA region access enable register for shadow region *m* (QRAE *m*) is programmed to allow or disallow read/write accesses on a bit-by-bit bases for all QDMA registers in the shadow region *m* view of the QDMA registers. This includes all 8-bit QDMA registers.

The QRAE *m* is shown in Figure 10-74 and described in Table 10-128.

Table 10-127. EDMACC_QRAE0 to EDMACC_QRAE7 Instances

Instance	Physical Address
EDMACC_0	0270 0380h to 0270 039Ch
EDMACC_1	0272 8380h to 0272 839Ch

Figure 10-74. EDMACC_QRAE0 to EDMACC_QRAE7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								QRAE7-QRAE0							
R-0h																								R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 10-128. EDMACC_QRAE0 to EDMACC_QRAE7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	QRAE7-QRAE0	R/W	0h	QDMA region access enable for bit N/QDMA channel N in region *m*. 0h = Accesses via region *m* address space to bit N in any QDMA channel register are not allowed. Reads return 0 on bit N and writes do not modify the state of bit N. 1h = Accesses via region m address space to bit n in any QDMA channel register are allowed. Reads return the value from bit n and writes modify the state of bit N.

Table 10-129. Register Call Summary for EDMACC_QRAE0

EDMA Registers
• EDMA Channel Controller (EDMACC) Registers: [0]

10.6.1.3 Status/Debug Visibility Registers

The following set of registers provide visibility into the event queues and a TR life cycle. These are useful for system debug as they provide in-depth visibility for the events queued up in the event queue and also provide information on what parts of the EDMACC logic are active once the event has been received by the EDMACC.

10.6.1.3.1 EDMACC_Q0E0 to EDMACC_Q3E15 Register (Offset = 400h to 4FCh) [reset = 0h]

The event queue entry registers (Q *x*E *y*) exist for all 16 queue entries (the maximum allowed queue entries) for all event queues in the EDMACC.

There are EDMACC_Q0E0 to Q0E15, Q1E0 to Q1E15, Q2E0 to Q2E15, and Q3E0 to EDMACC_Q3E15. Each register details the event number (ENUM) and the event type (ETYPE). For example, if the value in Q1E4 is read as 000 004Fh, this means the 4th entry in queue 1 is a manually-triggered event on DMA channel 15.

The Q *x*E *y* is shown in Figure 10-75 and described in Table 10-131.

Table 10-130. EDMACC_Q0E0 to EDMACC_Q3E15 Instances

Instance	Physical Address
EDMACC_0	0270 0400h to 0270 04FCh
EDMACC_1	0272 8400h to 0272 84FCh

Figure 10-75. EDMACC_Q0E0 to EDMACC_Q3E15 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
ETYPE		ENUM					
R-0h		R-0h					

LEGEND: R = Read Only; *-n* = value after reset

Table 10-131. EDMACC_Q0E0 to EDMACC_Q3E15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	ETYPE	R	0h	Event entry y in queue x. Specifies the specific event type for the given entry in the event queue. 0h = Event triggered via EDMACC_ER. 1h = Manual triggered via EDMACC_ESR. 2h = Chain triggered via EDMACC_CER. 3h = Auto-triggered via EDMACC_QER.
5-0	ENUM	R	0h	Event entry y in queue x. Event number: 0h - 7h = QDMA channel number (0 to 7). 0Fh - 3Fh = DMA channel/event number (0 to 63).

Table 10-132. Register Call Summary for EDMACC_Q0E0

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_Q0E0 to EDMACC_Q3E15 Register (Offset = 400h to 4FCh) [reset = 0h]: [0]

10.6.1.3.2 EDMACC_QSTAT0 to EDMACC_QSTAT3 Register (Offset = 600h to 60Ch) [reset = 0h]

The queue status registers (QSTAT *n*) is shown in Figure 4-28 and described in Table 10-134.

Table 10-133. EDMACC_QSTAT0 to EDMACC_QSTAT3 Instances

Instance	Physical Address
EDMACC_0	0270 0600h to 0270 060Ch
EDMACC_1	0272 8600h to 0272 860Ch

Figure 10-76. EDMACC_QSTAT0 to EDMACC_QSTAT3 Register

31	30	29	28	27	26	25	24
RESERVED							THRXCD
R-0h							R-0h
23	22	21	20	19	18	17	16
RESERVED			WM				
R-0h			R-0h				
15	14	13	12	11	10	9	8
RESERVED			NUMVAL				
R-0h			R-0h				
7	6	5	4	3	2	1	0
RESERVED				STRTPTR			
R-0h				R-0h			

LEGEND: R = Read Only; *-n* = value after reset

Table 10-134. EDMACC_QSTAT0 to EDMACC_QSTAT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24	THRXCD	R	0h	Threshold exceeded. THRXCD is cleared by writing a 1 to the corresponding QTHRXCD *n* bit in the EDMACC error clear register (EDMACC_CCERRCLR). 0h = Threshold specified by the Q *n* bit in the queue watermark threshold register (EDMACC_QWMTHRA) has not been exceeded. 1h = Threshold specified by the Q *n* bit in the queue watermark threshold register (EDMACC_QWMTHRA) has been exceeded.
23-21	RESERVED	R	0h	Reserved
20-16	WM	R	0h	Watermark for maximum queue usage. Watermark tracks the most entries that have been in queue *n* since reset or since the last time that the watermark (WM) bit was cleared. WM is cleared by writing a 1 to the corresponding QTHRXCD *n* bit in the EDMACC error clear register (EDMACC_CCERRCLR). 0h - 10h = Legal values are 0 (empty) to 10h (full). 11h - 1Fh = Reserved
15-13	RESERVED	R	0h	Reserved
12-8	NUMVAL	R	0h	Number of valid entries in queue *n*. The total number of entries residing in the queue manager FIFO at a given instant. Always enabled. 0h - 10h = Legal values are 0 (empty) to 10h (full). 11h - 1Fh = Reserved
7-4	RESERVED	R	0h	Reserved
3-0	STRTPTR	R	0h	Start pointer. The offset to the head entry of queue *n*, in units of entries. Always enabled. Legal values are 0 (0th entry) to Fh (15th entry).

Table 10-135. Register Call Summary for EDMACC_QSTAT0

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_CCERRCLR Register (Offset = 31Ch) [reset = 0h]: [0] • EDMACC_QWMTHRA Register (Offset = 620h) [reset = 2020202h]: [0][1]

10.6.1.3.3 EDMACC_QWMTHRA Register (Offset = 620h) [reset = 2020202h]

The queue watermark threshold A register (EDMACC_QWMTHRA) is shown in Figure 10-77 and described in Table 10-137.

Table 10-136. EDMACC_QWMTHRA Instances

Instance	Physical Address
EDMACC_0	0270 0620h
EDMACC_1	0272 8620h

Figure 10-77. EDMACC_QWMTHRA Register

31	30	29	28	27	26	25	24
RESERVED					Q3		
R-0h					R/W-2h		
23	22	21	20	19	18	17	16
RESERVED					Q2		
R-0h					R/W-2h		
15	14	13	12	11	10	9	8
RESERVED					Q1		
R-0h					R/W-2h		
7	6	5	4	3	2	1	0
RESERVED					Q0		
R-0h					R/W-2h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 10-137. EDMACC_QWMTHRA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved
28-24	Q3	R/W	2h	Queue threshold for queue 3 value. The QTHRXCD3 bit in the EDMACC error register (EDMACC_CCERR) and the THRXCD bit in the queue status register 3 (EDMACC_QSTAT3) are set when the number of events in queue 3 at an instant in time (visible via the NUMVAL bit in EDMACC_QSTAT3) equals or exceeds the value specified by Q3. 0h- 10h = The default is 16 (maximum allowed). 11h = Disables the threshold errors. 12h - 1Fh = Reserved
23-21	RESERVED	R	0h	Reserved
20-16	Q2	R/W	2h	Queue threshold for queue 2 value. The QTHRXCD2 bit in the EDMACC error register (EDMACC_CCERR) and the THRXCD bit in the queue status register 2 (QSTAT2) are set when the number of events in queue 2 at an instant in time (visible via the NUMVAL bit in QSTAT2) equals or exceeds the value specified by Q2. 0h - 10h = The default is 16 (maximum allowed). 11h = Disables the threshold errors. 12h = 1Fh = Reserved
15-13	RESERVED	R	0h	Reserved
12-8	Q1	R/W	2h	Queue threshold for queue 1 value. The QTHRXCD1 bit in the EDMACC error register (EDMACC_CCERR) and the THRXCD bit in the queue status register 1 (QSTAT1) are set when the number of events in queue 1 at an instant in time (visible via the NUMVAL bit in QSTAT1) equals or exceeds the value specified by Q1. 0h - 10h =The default is 16 (maximum allowed). 11h = Disables the threshold errors. 12h = 1Fh = Reserved
7-5	RESERVED	R	0h	Reserved

Table 10-137. EDMACC_QWMTHRA Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	Q0	R/W	2h	Queue threshold for queue 0 value. The QTHRXCD0 bit in the EDMACC error register (EDMACC_CCERR) and the THRXCD bit in the queue status register 0 (EDMACC_QSTAT0) are set when the number of events in queue 0 at an instant in time (visible via the NUMVAL bit in EDMACC_QSTAT0) equals or exceeds the value specified by Q0. 0h - 10h =The default is 16 (maximum allowed). 11h = Disables the threshold errors. 12h = 1Fh = Reserved

Table 10-138. Register Call Summary for EDMACC_QWMTHRA

Event Queue(s) • Queue Resource Tracking: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_QSTAT0 to EDMACC_QSTAT3 Register (Offset = 600h to 60Ch) [reset = 0h]: [0][1] • EDMACC_QWMTHRA Register (Offset = 620h) [reset = 2020202h]: [0] • EDMACC_CCERR Register (Offset = 318h) [reset = 0h]: [0]

10.6.1.3.4 EDMACC_CCSTAT Register (Offset = 640h) [reset = 0h]

The EDMACC status register (EDMACC_CCSTAT) has a number of status bits that reflect which parts of the EDMACC logic is active at any given instant of time. The EDMACC_CCSTAT is shown in Figure 10-78 and described in Table 10-140.

Table 10-139. EDMACC_CCSTAT Instances

Instance	Physical Address
EDMACC_0	0270 0640h
EDMACC_1	0272 8640h

Figure 10-78. EDMACC_CCSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				QUEACTV3	QUEACTV2	QUEACTV1	QUEACTV0
R-0h				R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED			COMPACTV				
R-0h			R-0h				
7	6	5	4	3	2	1	0
RESERVED			ACTV	WSTATACTV	TRACTV	QEVTACTV	EVTACTV
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 10-140. EDMACC_CCSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19	QUEACTV3	R	0h	Queue 3 active. 0h = No events are queued in queue 3. 1h = At least one TR is queued in queue 3.
18	QUEACTV2	R	0h	Queue 2 active. 0h = No events are queued in queue 2. 1h = At least one TR is queued in queue 2.
17	QUEACTV1	R	0h	Queue 1 active. 0h = No events are queued in queue 1. 1h = At least one TR is queued in queue 1.
16	QUEACTV0	R	0h	Queue 0 active. 0h = No events are queued in queue 0. 1h = At least one TR is queued in queue 0.
15-14	RESERVED	R	0h	Reserved
13-8	COMPACTV	R	0h	Completion request active. The COMPACTV field reflects the count for the number of completion requests submitted to the transfer controllers. This count increments every time a TR is submitted and is programmed to report completion (the TCINTEN or TCCCHEN bits in OPT in the parameter entry associated with the TR are set). The counter decrements for every valid TCC received back from the transfer controllers. If at any time the count reaches a value of 63, the EDMACC will not service any new TRs until the count is less then 63 (or return a transfer completion code from a transfer controller, which would decrement the count). 0h = No completion requests outstanding. 1h-3Fh = Total of 1 completion request to 63 completion requests are outstanding.

Table 10-140. EDMACC_CCSTAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-5	RESERVED	R	0h	Reserved
4	ACTV	R	0h	Channel controller active. Channel controller active is a logical-OR of each of the *ACTV bits. The ACTV bit remains high through the life of a TR. 0h = Channel is idle. 1h = Channel is busy.
3	WSTATACTV	R	0h	Write status interface active. 0h = Write status req is idle and write status FIFO is idle. 1h = Either the write status request is active or additional write status responses are pending in the write status FIFO.
2	TRACTV	R	0h	Transfer request active. 0h = Transfer request processing/submission logic is inactive. 1h = Transfer request processing/submission logic is active.
1	QEVTACTV	R	0h	QDMA event active. 0h = No enabled QDMA events are active within the EDMACC. 1h = At least one enabled QDMA event (EDMACC_QER) is active within the EDMACC.
0	EVTACTV	R	0h	DMA event active. 0h = No enabled DMA events are active within the EDMACC. 1h = At least one enabled DMA event (EDMACC_ER and EDMACC_EER, EDMACC_ESR, EDMACC_CER) is active within the EDMACC.

Table 10-141. Register Call Summary for EDMACC_CCSTAT

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_CCSTAT Register (Offset = 640h) [reset = 0h]: [0][1]

10.6.1.4 Memory Protection Address Space

10.6.1.4.1 EDMACC_MPFAR Register (Offset = 800h) [reset = 0h]

The memory protection fault address register (EDMACC_MPFAR) is shown in Figure 10-79 and described in Table 10-143. A DSP write of 1 to the MPFCLR bit in the memory protection fault command register (EDMACC_MPFCR) causes any error conditions stored in EDMACC_MPFAR to be cleared.

Table 10-142. EDMACC_MPFAR Instances

Instance	Physical Address
EDMACC_0	0270 0800h
EDMACC_1	0272 8800h

Figure 10-79. EDMACC_MPFAR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FADDR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-143. EDMACC_MPFAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FADDR	R	0h	Fault address. This 32-bit read-only status register contains the fault address when a memory protection violation is detected. This register can only be cleared via the memory protection fault command register (EDMACC_MPFCR).

Table 10-144. Register Call Summary for EDMACC_MPFAR

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_MPFCR Register (Offset = 808h) [reset = 0h]: [0] • EDMACC_MPFAR Register (Offset = 800h) [reset = 0h]: [0][1]

10.6.1.4.2 EDMACC_MPFSR Register (Offset = 804h) [reset = 0h]

The memory protection fault status register (EDMACC_MPFSR) is shown in Figure 10-80 and described in Table 10-146. A DSP write of 1 to the MPFCLR bit in the memory protection fault command register (EDMACC_MPFCR) causes any error conditions stored in EDMACC_MPFSR to be cleared.

Table 10-145. EDMACC_MPFSR Instances

Instance	Physical Address
EDMACC_0	0270 0804h
EDMACC_1	0272 8804h

Figure 10-80. EDMACC_MPFSR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			FID				RESERVED
R-0h			R-0h				R-0h
7	6	5	4	3	2	1	0
RESERVED		SRE	SWE	SXE	URE	UWE	UXE
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 10-146. EDMACC_MPFSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-9	FID	R	0h	Faulted identification. FID contains valid information if any of the MP error bits (UXE, UWE, URE, SXE, SWE, SRE) are nonzero (that is, if an error has been detected.) The FID field contains the privilege ID for the specific request/requester that resulted in an MP error.
8-6	RESERVED	R	0h	Reserved
5	SRE	R	0h	Supervisor read error. 0h = No error detected. 1h = Supervisor level task attempted to read from a MP page without SR permissions.
4	SWE	R	0h	Supervisor write error. 0h = No error detected. 1h = Supervisor level task attempted to write to a MP page without SW permissions.
3	SXE	R	0h	Supervisor execute error. 0h = No error detected. 1h = Supervisor level task attempted to execute from a MP page without SX permissions.
2	URE	R	0h	User read error. 0h = No error detected. 1h = User level task attempted to read from a MP page without UR permissions.
1	UWE	R	0h	User write error. 0h = No error detected. 1h = User level task attempted to write to a MP page without UW permissions.

Table 10-146. EDMACC_MPFSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	UXE	R	0h	User execute error. 0h = No error detected. 1h = User level task attempted to execute from a MP page without UX permissions.

Table 10-147. Register Call Summary for EDMACC_MPFSR

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_MPFSR Register (Offset = 804h) [reset = 0h]: [0][1] • EDMACC_MPFCR Register (Offset = 808h) [reset = 0h]: [0]

10.6.1.4.3 EDMACC_MPFCR Register (Offset = 808h) [reset = 0h]

The memory protection fault command register (EDMACC_MPFCR) is shown in Figure 10-81 and described in Table 10-149.

Table 10-148. EDMACC_MPFCR Instances

Instance	Physical Address
EDMACC_0	0270 0808h
EDMACC_1	0272 8808h

Figure 10-81. EDMACC_MPFCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							MPFCLR
R-0h							W-0h

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 10-149. EDMACC_MPFCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	MPFCLR	W	0h	Fault clear register. 0h = DSP write of 0 has no effect. 1h = DSP write of 1 to the MPFCLR bit causes any error conditions stored in the memory protection fault address register (EDMACC_MPFAR) and the memory protection fault status register (EDMACC_MPFSR) to be cleared.

Table 10-150. Register Call Summary for EDMACC_MPFCR

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller (EDMACC) Registers • EDMACC_MPFSR Register (Offset = 804h) [reset = 0h]: [0] • EDMACC_MPFAR Register (Offset = 800h) [reset = 0h]: [0][1] • EDMACC_MPFCR Register (Offset = 808h) [reset = 0h]: [0]

10.6.1.4.4 EDMACC_MPPAG Register (Offset = 80Ch) [reset = 676h]

The memory protection page attribute register (MPPA *n*) is shown in Figure 10-82 and described in Table 10-152.

Table 10-151. EDMACC_MPPAG Instances

Instance	Physical Address
EDMACC_0	0270 080Ch
EDMACC_1	0272 880Ch

Figure 10-82. EDMACC_MPPAG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
AID5-AID0						EXT	RESERVED
RW-1h						RW-1h	R-1h
7	6	5	4	3	2	1	0
RESERVED		SR	SW	SX	UR	UW	UX
R-0h		RW-1h	RW-1h	RW-0h	RW-1h	RW-1h	RW-0h

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 10-152. EDMACC_MPPAG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-10	AID5-AID0	RW	1h	Allowed ID *n* 0h = Requests with Privilege ID == n are not allowed to region m, regardless of permission settings (UW, UR, SW, SR). 1h = Requests with Privilege ID == n are permitted, if access type is allowed as defined by permission settings (UW, UR, SW, SR).
9	EXT	RW	1h	External Allowed ID. 0h = Requests with Privilege ID >= 6 are not allowed to region m, regardless of permission settings (UW, UR, SW, SR). 1h = Requests with Privilege ID >= 6 are permitted, if access type is allowed as defined by permission settings (UW, UR, SW, SR).
8-6	RESERVED	R	1h	Reserved
5	SR	RW	1h	Supervisor read permission 0h = Supervisor read accesses are not allowed from region *m*. 1h = Supervisor write accesses are allowed from region *m* addresses.
4	SW	RW	1h	Supervisor write permission. 0h = Supervisor write accesses are not allowed to region *m*. 1h = Supervisor write accesses are allowed to region *n* addresses.
3	SX	RW	0h	Supervisor execute permission. 0h = Supervisor execute accesses are not allowed from region *m*. 1h = Supervisor execute accesses are allowed from region *m* addresses.
2	UR	RW	1h	User read permission. 0h = User read accesses are not allowed from region *m*. 1h = User read accesses are allowed from region *n* addresses.

Table 10-152. EDMACC_MPPAG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	UW	RW	1h	User write permission. 0h = User write accesses are not allowed to region *m*. 1h = User write accesses are allowed to region *m* addresses.
0	UX	RW	0h	User execute permission. 0h = User execute accesses are not allowed from region *m*. 1h = User execute accesses are allowed from region *m* addresses.

Table 10-153. Register Call Summary for EDMACC_MPPAG

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
EDMA Channel Controller Regions • Channel Controller Shadow Regions: [0]
Memory Protection • Active Memory Protection: [0][1][2][4]

10.6.1.4.5 EDMACC_MPPA0 to EDMACC_MPPA7 Register (Offset = 810h to 82Ch) [reset = 676h]

The memory protection page attribute register (MPPA *n*) is shown in Figure 10-83 and described in Table 10-155.

Table 10-154. EDMACC_MPPA0 to EDMACC_MPPA7 Instances

Instance	Physical Address
EDMACC_0	0270 0810h to 0270 082Ch
EDMACC_1	0272 8810h to 0272 882Ch

Figure 10-83. EDMACC_MPPA0 to EDMACC_MPPA7 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
AID5-AID0						EXT	RESERVED
RW-1h						RW-1h	R-1h
7	6	5	4	3	2	1	0
RESERVED		SR	SW	SX	UR	UW	UX
R-0h		RW-1h	RW-1h	RW-0h	RW-1h	RW-1h	RW-0h

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 10-155. EDMACC_MPPA0 to EDMACC_MPPA7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-10	AID5-AID0	RW	1h	Allowed ID *n* 0h = Requests with Privilege ID == n are not allowed to region m, regardless of permission settings (UW, UR, SW, SR). 1h = Requests with Privilege ID == n are permitted, if access type is allowed as defined by permission settings (UW, UR, SW, SR).
9	EXT	RW	1h	External Allowed ID. 0h = Requests with Privilege ID >= 6 are not allowed to region m, regardless of permission settings (UW, UR, SW, SR). 1h = Requests with Privilege ID >= 6 are permitted, if access type is allowed as defined by permission settings (UW, UR, SW, SR).
8-6	RESERVED	R	1h	Reserved
5	SR	RW	1h	Supervisor read permission 0h = Supervisor read accesses are not allowed from region *m*. 1h = Supervisor write accesses are allowed from region *m* addresses.
4	SW	RW	1h	Supervisor write permission. 0h = Supervisor write accesses are not allowed to region *m*. 1h = Supervisor write accesses are allowed to region *n* addresses.
3	SX	RW	0h	Supervisor execute permission. 0h = Supervisor execute accesses are not allowed from region *m*. 1h = Supervisor execute accesses are allowed from region *m* addresses.
2	UR	RW	1h	User read permission. 0h = User read accesses are not allowed from region *m*. 1h = User read accesses are allowed from region *n* addresses.

Table 10-155. EDMACC_MPPA0 to EDMACC_MPPA7 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	UW	RW	1h	User write permission. 0h = User write accesses are not allowed to region *m*. 1h = User write accesses are allowed to region *m* addresses.
0	UX	RW	0h	User execute permission. 0h = User execute accesses are not allowed from region *m*. 1h = User execute accesses are allowed from region *m* addresses.

Table 10-156. Register Call Summary for EDMACC_MPPA0

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0]
Memory Protection • Active Memory Protection: [0]

10.6.1.5 DMA Channel Registers

The following sets of registers pertain to the 64 DMA channels. The 64 DMA channels consist of a set of registers (with exception of DMAQNUM *n*) that each have 64 bits and the bit position of each register matches the DMA channel number. Each register is named with the format reg_name that corresponds to DMA channels 0 through 31 and reg_name_High that corresponds to DMA channels 32 through 64. Note that high registers (reg_name_High) are applicable only when the EDMACC supports more than 32 DMA channels.

For example, the event register (EDMACC_ER) corresponds to DMA channel 0 through 31 and the event register high register (EDMACC_ERH) corresponds to DMA channel 32 through 63. The register is typically called the event register.

The DMA channel registers are accessible via read/writes to the global address range. They also are accessible via read/writes to the shadow address range. The read/write ability to the registers in the shadow region are controlled by the DMA region access registers (DRAE *m*/DRAEH *m*). The registers are described in and the details for shadow region/global region usage is explained in .

10.6.1.5.1 EDMACC_ER Register (Offset = 1000h) [reset = 0h]

All external events are captured in the event register (EDMACC_ER/EDMACC_ERH). The events are latched even when the events are not enabled. If the event bit corresponding to the latched event is enabled (EDMACC_EER.E *n*/EDMACC_EERH.E *n* = 1), then the event is evaluated by the EDMACC logic for an associated transfer request submission to the transfer controllers. The event register bits are automatically cleared (EDMACC_ER.E *n*/EDMACC_ERH.E *n* = 0) once the corresponding events are prioritized and serviced. If EDMACC_ER.E *n*/EDMACC_ERH.E *n* are already set and another event is received on the same channel/event, then the corresponding event is latched in the event miss register (EDMACC_EMR.E *n*/EDMACC_EMRH.E *n*), provided that the event was enabled (EDMACC_EER.E *n*/EDMACC_EERH.E *n* = 1).

Event *n* can be cleared by the DSP writing a 1 to corresponding event bit in the event clear register (EDMACC_ECR/EDMACC_ECRH). The setting of an event is a higher priority relative to clear operations (via hardware or software). If set and clear conditions occur concurrently, the set condition wins. If the event was previously set, then EDMACC_EMR/EDMACC_EMRH would be set because an event is lost. If the event was previously clear, then the event remains set and is prioritized for submission to the event queues.

provides the type of synchronization events and the EDMACC channels associated to each of these external events.

The EDMACC_ERH is shown in Figure 10-85 and described in Table 10-161.

Table 10-157. EDMACC_ER Instances

Instance	Physical Address
EDMACC_0	0270 1000h
EDMACC_1	0272 9000h

Figure 10-84. EDMACC_ER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31-E0																															
R-0h																															

LEGEND: R = Read Only; -*n* = value after reset

Table 10-158. EDMACC_ER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	E31-E0	R	0h	Event 31-0. Events 0-31 are captured by the EDMACC and are latched into EDMACC_ER. The events are set (EN = 1) even when events are disabled (E31-E0 = 0 in the event enable register, EDMACC_EER). 0h = EDMACC event is not asserted. 1h = EDMACC event is asserted. Corresponding DMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMATC.

Table 10-159. Register Call Summary for EDMACC_ER

EDMA Debug/Programming Tips • Miscellaneous Programming/Debug Tips: [0] • Debug Checklist: [0] • Setting Up a Transfer: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_EER Register (Offset = 1020h) [reset = 0h]: [0][1][2] • EDMACC_CCSTAT Register (Offset = 640h) [reset = 0h]: [0] • EDMACC_EMRH Register (Offset = 304h) [reset = 0h]: [0] • EDMACC_EMR Register (Offset = 300h) [reset = 0h]: [0] • EDMACC_ECR Register (Offset = 1008h) [reset = 0h]: [0][1][2] • EDMACC_ESR Register (Offset = 1010h) [reset = 0h]: [0] • EDMACC_ER Register (Offset = 1000h) [reset = 0h]: [0][1][2][3] • EDMACC_SER Register (Offset = 1038h) [reset = 0h]: [0][1] • DMA Channel Registers: [0] • EDMACC_Q0E0 to EDMACC_Q3E15 Register (Offset = 400h to 4FCh) [reset = 0h]: [0]
EDMA Channel Controller Regions • Region Overview: [0]
EDMA Prioritization • Channel Priority: [0] • Trigger Source Priority: [0][1][2][3]
EDMA Functional Description • Event Dataflow: [0][1]
Initiating a DMA Transfer • Chain-Triggered Transfer Request: [0] • Event-Triggered Transfer Request: [0][1][2][3][4][5][6][7] • Manually-Triggered Transfer Request: [0]

10.6.1.5.2 EDMACC_ERH Register (Offset = 1004h) [reset = 0h]

The EDMACC_ERH is shown in Figure 10-85 and described in Table 10-161.

Table 10-160. EDMACC_ERH Instances

Instance	Physical Address
EDMACC_0	0270 1004h
EDMACC_1	0272 9004h

Figure 10-85. EDMACC_ERH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63-E32																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-161. EDMACC_ERH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	E63-E32	R	0h	Event 63-32. Events 32-63 are captured by the EDMACC and are latched into EDMACC_ERH. The events are set (EN = 1) even when events are disabled (E63-E32 = 0 in the event enable register high, EDMACC_EERH). 0h = EDMACC event is not asserted. 1h = EDMACC event is asserted. Corresponding DMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMATC.

Table 10-162. Register Call Summary for EDMACC_ERH

EDMA Debug/Programming Tips • Miscellaneous Programming/Debug Tips: [0] • Debug Checklist: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • DMA Channel Registers: [0] • EDMACC_ESR Register (Offset = 1010h) [reset = 0h]: [0] • EDMACC_EMR Register (Offset = 300h) [reset = 0h]: [0] • EDMACC_ERH Register (Offset = 1004h) [reset = 0h]: [0][1] • EDMACC_EMRH Register (Offset = 304h) [reset = 0h]: [0] • EDMACC_SERH Register (Offset = 103Ch) [reset = 0h]: [0] • EDMACC_ER Register (Offset = 1000h) [reset = 0h]: [0][1][2][3] • EDMACC_ECR Register (Offset = 1008h) [reset = 0h]: [0] • EDMACC_ECRH Register (Offset = 100Ch) [reset = 0h]: [0][1] • EDMACC_EERH Register (Offset = 1024h) [reset = 0h]: [0][1] • EDMACC_SER Register (Offset = 1038h) [reset = 0h]: [0] • EDMACC_EER Register (Offset = 1020h) [reset = 0h]: [0]
EDMA Channel Controller Regions • Region Overview: [0]
EDMA Prioritization • Channel Priority: [0]
EDMA Functional Description • Event Dataflow: [0][1]
Initiating a DMA Transfer • Chain-Triggered Transfer Request: [0]

10.6.1.5.3 EDMACC_ECR Register (Offset = 1008h) [reset = 0h]

Once an event has been posted in the event registers (EDMACC_ER/EDMACC_ERH), the event is cleared in two ways. If the event is enabled in the event enable register (EDMACC_EER/EDMACC_EERH) and the EDMACC submits a transfer request for the event to the EDMATC, it clears the corresponding event bit in the event register. If the event is disabled in the event enable register (EDMACC_EER/EDMACC_EERH), the DSP can clear the event by way of the event clear registers (EDMACC_ECR/EDMACC_ECRH).

Writing a 1 to any of the bits clears the corresponding event; writing a 0 has no effect. Once an event bit is set in the event register, it remains set until EDMACC submits a transfer request for that event or the DSP clears the event by setting the corresponding bit in EDMACC_ECR/EDMACC_ECRH.

The EDMACC_ECR is shown in Figure 10-86 and described in Table 10-164.

Table 10-163. EDMACC_ECR Instances

Instance	Physical Address
EDMACC_0	0270 1008h
EDMACC_1	0272 9008h

Figure 10-86. EDMACC_ECR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31-E0																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 10-164. EDMACC_ECR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	E31-E0	W	0h	Event clear for event 31-0. Any of the event bits in EDMACC_ECR is set to clear the event (EN) in the event register (EDMACC_ER). A write of 0 has no effect. 0h = No effect. 1h = EDMACC event is cleared in the event register (EDMACC_ER).

Table 10-165. Register Call Summary for EDMACC_ECR

EDMA Debug/Programming Tips • Miscellaneous Programming/Debug Tips: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_ECR Register (Offset = 1008h) [reset = 0h]: [0][1][2][3] • EDMACC_ESR Register (Offset = 1010h) [reset = 0h]: [0] • EDMACC_ER Register (Offset = 1000h) [reset = 0h]: [0]
EDMA Channel Controller Regions • Region Overview: [0]

10.6.1.5.4 EDMACC_ECRH Register (Offset = 100Ch) [reset = 0h]

The EDMACC_ECRH is shown in Figure 10-87 and described in Table 10-167.

Table 10-166. EDMACC_ECRH Instances

Instance	Physical Address
EDMACC_0	0270 100Ch
EDMACC_1	0272 900Ch

Figure 10-87. EDMACC_ECRH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63-E32																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 10-167. EDMACC_ECRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	E63-E32	W	0h	Event clear for event 63-32. Any of the event bits in EDMACC_ECRH are set to clear the event (EN) in the event register high (EDMACC_ERH). A write of 0 has no effect. 0h = No effect. 1h = EDMACC event is cleared in the event register high (EDMACC_ERH).

Table 10-168. Register Call Summary for EDMACC_ECRH

EDMA Debug/Programming Tips • Miscellaneous Programming/Debug Tips: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_ESR Register (Offset = 1010h) [reset = 0h]: [0] • EDMACC_ECR Register (Offset = 1008h) [reset = 0h]: [0][1] • EDMACC_ECRH Register (Offset = 100Ch) [reset = 0h]: [0][1] • EDMACC_ER Register (Offset = 1000h) [reset = 0h]: [0]
EDMA Channel Controller Regions • Region Overview: [0]

10.6.1.5.5 EDMACC_ESR Register (Offset = 1010h) [reset = 0h]

The event set registers (EDMACC_ESR/EDMACC_ESRH) allow the DSP (EDMA programmers) to manually set events to initiate DMA transfer requests. DSP writes of 1 to any event set register (E *n*) bits set the corresponding bits in the registers. The set event is evaluated by the EDMACC logic for an associated transfer request submission to the transfer controllers. Writing a 0 has no effect.

The event set registers operate independent of the event registers (EDMACC_ER/EDMACC_ERH), and a write of 1 is always considered a valid event regardless of whether the event is enabled (the corresponding event bits are set or cleared in EDMACC_EER.E *n*/EDMACC_EERH.E *n*).

Once the event is set in the event set registers, it cannot be cleared by DSP writes, in other words, the event clear registers (EDMACC_ECR/EDMACC_ECRH) have no effect on the state of EDMACC_ESR/EDMACC_ESRH. The bits will only be cleared once the transfer request corresponding to the event has been submitted to the transfer controller. The setting of an event is a higher priority relative to clear operations (via hardware). If set and clear conditions occur concurrently, the set condition wins. If the event was previously set, then EDMACC_EMR/EDMACC_EMRH would be set because an event is lost. If the event was previously clear, then the event remains set and is prioritized for submission to the event queues.

Manually-triggered transfers via writes to EDMACC_ESR/EDMACC_ESRH allow the DSP to submit DMA requests in the system, these are relevant for memory-to-memory transfer scenarios. If the EDMACC_ESR.E *n*/EDMACC_ESRH.E *n* bit is already set and another DSP write of 1 is attempted to the same bit, then the corresponding event is latched in the event missed registers (EDMACC_EMR.E *n*/EDMACC_EMRH.E *n* = 1).

The EDMACC_ESR is shown in Figure 10-88 and described in Table 10-170.

Table 10-169. EDMACC_ESR Instances

Instance	Physical Address
EDMACC_0	0270 1010h
EDMACC_1	0272 9010h

Figure 10-88. EDMACC_ESR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31-E0																															
RW-0h																															

LEGEND: RW = Read/Write; -*n* = value after reset

Table 10-170. EDMACC_ESR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	E31-E0	RW	0h	Event set for event 31-0. 0h = No effect. 1h = Corresponding DMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMATC.

Table 10-171. Register Call Summary for EDMACC_ESR

EDMA Debug/Programming Tips • Setting Up a Transfer: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]

Table 10-171. Register Call Summary for EDMACC_ESR (continued)

EDMA Channel Controller (EDMACC) Registers • EDMACC_CCSTAT Register (Offset = 640h) [reset = 0h]: [0] • EDMACC_EMRH Register (Offset = 304h) [reset = 0h]: [0] • EDMACC_EMR Register (Offset = 300h) [reset = 0h]: [0] • EDMACC_ESR Register (Offset = 1010h) [reset = 0h]: [0][1][2][3][4] • EDMACC_EER Register (Offset = 1020h) [reset = 0h]: [0] • EDMACC_SER Register (Offset = 1038h) [reset = 0h]: [0] • EDMACC_Q0E0 to EDMACC_Q3E15 Register (Offset = 400h to 4FCh) [reset = 0h]: [0]
Peripheral Servicing Example • Breaking Up Large Transfers with Intermediate Chaining: [0]
EDMA Channel Controller Regions • Region Overview: [0]
EDMA Prioritization • Trigger Source Priority: [0][1]
EDMA Functional Description • Event Dataflow: [0][1] • Initiating a DMA Transfer: [0]
Initiating a DMA Transfer • Chain-Triggered Transfer Request: [0] • Comparison Between DMA and QDMA Channels: [0] • Manually-Triggered Transfer Request: [0][1][2][3]

10.6.1.5.6 EDMACC_ESRH Register (Offset = 1014h) [reset = 0h]

The EDMACC_ESRH is shown in Figure 10-89 and described in Table 10-173.

Table 10-172. EDMACC_ESRH Instances

Instance	Physical Address
EDMACC_0	0270 1014h
EDMACC_1	0272 9014h

Figure 10-89. EDMACC_ESRH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63-E32																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 10-173. EDMACC_ESRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	E63-E32	RW	0h	Event set for event 63-32. 0h = No effect. 1h = Corresponding DMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMATC.

Table 10-174. Register Call Summary for EDMACC_ESRH

EDMA Debug/Programming Tips • Setting Up a Transfer: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_EMRH Register (Offset = 304h) [reset = 0h]: [0] • EDMACC_EMR Register (Offset = 300h) [reset = 0h]: [0] • EDMACC_ESR Register (Offset = 1010h) [reset = 0h]: [0][1][2][3] • EDMACC_ESRH Register (Offset = 1014h) [reset = 0h]: [0] • EDMACC_SER Register (Offset = 1038h) [reset = 0h]: [0] • EDMACC_EER Register (Offset = 1020h) [reset = 0h]: [0]
EDMA Channel Controller Regions • Region Overview: [0]
EDMA Functional Description • Event Dataflow: [0][1] • Initiating a DMA Transfer: [0]
Initiating a DMA Transfer • Chain-Triggered Transfer Request: [0]

10.6.1.5.7 EDMACC_CER Register (Offset = 1018h) [reset = 0h]

When the OPTIONS parameter for a PaRAM entry is programmed to returned a chained completion code (ITCCHEN = 1 and/or TCCHEN = 1), then the value dictated by the TCC[5:0] (also programmed in OPT) forces the corresponding event bit to be set in the chained event registers (EDMACC_CER/EDMACC_CERH). The set chained event is evaluated by the EDMACC logic for an associated transfer request submission to the transfer controllers. This results in a chained-triggered transfer.

The chained event registers do not have any enables. The generation of a chained event is essentially enabled by the PaRAM entry that has been configured for intermediate and/or final chaining on transfer completion. The E *n* bit is set (regardless of the state of EDMACC_EER.E *n*/EDMACC_EERH.E *n*) when a chained completion code is returned from one of the transfer controllers or is generated by the EDMACC via the early completion path. The bits in the chained event register are cleared when the corresponding events are prioritized and serviced.

If the E *n* bit is already set and another chaining completion code is return for the same event, then the corresponding event is latched in the event missed registers (EDMACC_EMR.E *n*/EDMACC_EMRH.E *n* = 1). The setting of an event is a higher priority relative to clear operations (via hardware). If set and clear conditions occur concurrently, the set condition wins. If the event was previously set, then EDMACC_EMR/EDMACC_EMRH would be set because an event is lost. If the event was previously clear, then the event remains set and is prioritized for submission to the event queues.

The EDMACC_CER is shown in Figure 10-90 and described in Table 10-176.

Table 10-175. EDMACC_CER Instances

Instance	Physical Address
EDMACC_0	0270 1018h
EDMACC_1	0272 9018h

Figure 10-90. EDMACC_CER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31-E0																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-176. EDMACC_CER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	E31-E0	R	0h	Chained event for event 31-0. 0h = No effect. 1h = Corresponding DMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMATC.

Table 10-177. Register Call Summary for EDMACC_CER

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_SER Register (Offset = 1038h) [reset = 0h]: [0] • EDMACC_CCSTAT Register (Offset = 640h) [reset = 0h]: [0] • EDMACC_EMRH Register (Offset = 304h) [reset = 0h]: [0] • EDMACC_EMR Register (Offset = 300h) [reset = 0h]: [0] • EDMACC_EER Register (Offset = 1020h) [reset = 0h]: [0] • EDMACC_CER Register (Offset = 1018h) [reset = 0h]: [0][1] • EDMACC_Q0E0 to EDMACC_Q3E15 Register (Offset = 400h to 4FCh) [reset = 0h]: [0]
Peripheral Servicing Example • Servicing Input/Output FIFOs with a Single Event: [0][1]

Table 10-177. Register Call Summary for EDMACC_CER (continued)

EDMA Channel Controller Regions • Region Overview: [0]
EDMA Prioritization • Trigger Source Priority: [0][1][2][3]
EDMA Functional Description • Event Dataflow: [0][1] • Completion of a DMA Transfer: [0]
Parameter RAM (PaRAM) • Channel Options Parameter (OPT): [0][1][2][3] • Dummy Versus Null Transfer Comparison: [0]
Initiating a DMA Transfer • Chain-Triggered Transfer Request: [0][1][2][3][4][5]
Completion of a DMA Transfer • Dummy or Null Completion: [0]

10.6.1.5.8 EDMACC_CERH Register (Offset = 101Ch) [reset = 0h]

The EDMACC_CERH is shown in Figure 10-91 and described in Table 10-179.

Table 10-178. EDMACC_CERH Instances

Instance	Physical Address
EDMACC_0	0270 101Ch
EDMACC_1	0272 901Ch

Figure 10-91. EDMACC_CERH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63-E32																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-179. EDMACC_CERH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	E63-E32	R	0h	Chained event set for event 63-32. 0h = No effect. 1h = Corresponding DMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMATC.

Table 10-180. Register Call Summary for EDMACC_CERH

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_SER Register (Offset = 1038h) [reset = 0h]: [0] • EDMACC_EMRH Register (Offset = 304h) [reset = 0h]: [0] • EDMACC_CERH Register (Offset = 101Ch) [reset = 0h]: [0] • EDMACC_EMR Register (Offset = 300h) [reset = 0h]: [0] • EDMACC_CER Register (Offset = 1018h) [reset = 0h]: [0] • EDMACC_EER Register (Offset = 1020h) [reset = 0h]: [0]
EDMA Channel Controller Regions • Region Overview: [0]
EDMA Functional Description • Event Dataflow: [0][1]
Parameter RAM (PaRAM) • Channel Options Parameter (OPT): [0][1][2][3] • Dummy Versus Null Transfer Comparison: [0]
Initiating a DMA Transfer • Chain-Triggered Transfer Request: [0]
Completion of a DMA Transfer • Dummy or Null Completion: [0]

10.6.1.5.9 EDMACC_EER Register (Offset = 1020h) [reset = 0h]

The EDMACC provides the option of selectively enabling/disabling each event in the event registers (EDMACC_ER/EDMACC_ERH) by using the event enable registers (EDMACC_EER/EDMACC_EERH). If an event bit in EDMACC_EER/EDMACC_EERH is set (using the event enable set registers, EDMACC_EESR/EDMACC_EESRH), it will enable that corresponding event. Alternatively, if an event bit in EDMACC_EER/EDMACC_EERH is cleared (using the event enable clear registers, EDMACC_EECR/EDMACC_EECRH), it will disable the corresponding event.

The event registers latch all events that are captured by EDMACC, even if the events are disabled (although EDMACC does not process it). Enabling an event with a pending event already set in the event registers enables the EDMACC to process the already set event like any other new event. The EDMACC_EER/EDMACC_EERH settings do not have any effect on chained events (EDMACC_CER.E *n*/EDMACC_CERH.E *n* = 1) and manually set events (EDMACC_ESR.E *n*/EDMACC_ESRH.E *n* = 1).

The EDMACC_EER is shown in Figure 10-92 and described in Table 10-182.

Table 10-181. EDMACC_EER Instances

Instance	Physical Address
EDMACC_0	0270 1020h
EDMACC_1	0272 9020h

Figure 10-92. EDMACC_EER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31-E0																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-182. EDMACC_EER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	E31-E0	R	0h	Event enable for events 31-0. 0h = Event is not enabled. An external event latched in the event register (EDMACC_ER) is not evaluated by the EDMACC. 1h = Event is enabled. An external event latched in the event register (EDMACC_ER) is evaluated by the EDMACC.

Table 10-183. Register Call Summary for EDMACC_EER

EDMA Debug/Programming Tips • Miscellaneous Programming/Debug Tips: [0] • Debug Checklist: [0] • Setting Up a Transfer: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_CCSTAT Register (Offset = 640h) [reset = 0h]: [0] • EDMACC_EECR Register (Offset = 1028h) [reset = 0h]: [0][1][2] • EDMACC_ECR Register (Offset = 1008h) [reset = 0h]: [0][1] • EDMACC_ESR Register (Offset = 1010h) [reset = 0h]: [0] • EDMACC_EESR Register (Offset = 1030h) [reset = 0h]: [0][1][2] • EDMACC_ER Register (Offset = 1000h) [reset = 0h]: [0][1][2] • EDMACC_CER Register (Offset = 1018h) [reset = 0h]: [0] • EDMACC_EER Register (Offset = 1020h) [reset = 0h]: [0][1][2][3][4]
EDMA Transfer Examples • Peripheral Servicing Example: [0]

Table 10-183. Register Call Summary for EDMACC_EER (continued)

EDMA Channel Controller Regions • Region Overview: [0][1]
Initiating a DMA Transfer • Event-Triggered Transfer Request: [0][1][2][3][4] • Manually-Triggered Transfer Request: [0]
Memory Protection • Write access to shadow region 7's event enable set register (EDMACC_EESR): [0][1][2][3] • Access Denied to an EDMACC Register: [0][1][2][3] • Active Memory Protection: [0][1][2][3][4][5][6][7]

10.6.1.5.10 EDMACC_EERH Register (Offset = 1024h) [reset = 0h]

The EDMACC_EERH is shown in Figure 10-93 and described in Table 10-185.

Table 10-184. EDMACC_EERH Instances

Instance	Physical Address
EDMACC_0	0270 1024h
EDMACC_1	0272 9024h

Figure 10-93. EDMACC_EERH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63-E32																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-185. EDMACC_EERH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	E63-E32	R	0h	Event enable for events 63-32. 0h = Event is not enabled. An external event latched in the event register high (EDMACC_ERH) is not evaluated by the EDMACC. 1h = Event is enabled. An external event latched in the event register high (EDMACC_ERH) is evaluated by the EDMACC.

Table 10-186. Register Call Summary for EDMACC_EERH

EDMA Debug/Programming Tips • Miscellaneous Programming/Debug Tips: [0] • Debug Checklist: [0] • Setting Up a Transfer: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_ESR Register (Offset = 1010h) [reset = 0h]: [0] • EDMACC_EESRH Register (Offset = 1034h) [reset = 0h]: [0] • EDMACC_ER Register (Offset = 1000h) [reset = 0h]: [0][1] • EDMACC_ECR Register (Offset = 1008h) [reset = 0h]: [0][1] • EDMACC_CER Register (Offset = 1018h) [reset = 0h]: [0] • EDMACC_ERH Register (Offset = 1004h) [reset = 0h]: [0] • EDMACC_EESR Register (Offset = 1030h) [reset = 0h]: [0][1] • EDMACC_EECR Register (Offset = 1028h) [reset = 0h]: [0][1] • EDMACC_EECRH Register (Offset = 102Ch) [reset = 0h]: [0] • EDMACC_EERH Register (Offset = 1024h) [reset = 0h]: [0] • EDMACC_EER Register (Offset = 1020h) [reset = 0h]: [0][1][2][3]
EDMA Channel Controller Regions • Region Overview: [0]

10.6.1.5.11 EDMACC_EECR Register (Offset = 1028h) [reset = 0h]

The event enable registers (EDMACC_EER/EDMACC_EERH) cannot be modified by directly writing to them. The intent is to ease the software burden for the case where multiple tasks are attempting to simultaneously modify these registers. The event enable clear registers (EDMACC_EECR/EDMACC_EECRH) are used to disable events. Writes of 1 to the bits in EDMACC_EECR/EDMACC_EECRH clear the corresponding event bits in EDMACC_EER/EDMACC_EERH; writes of 0 have no effect.

The EDMACC_EECR is shown in Figure 10-94 and described in Table 10-188.

Table 10-187. EDMACC_EECR Instances

Instance	Physical Address
EDMACC_0	0270 1028h
EDMACC_1	0272 9028h

Figure 10-94. EDMACC_EECR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31-E0																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 10-188. EDMACC_EECR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	E31-E0	W	0h	Event enable clear for events 31-0. 0h = No effect. 1h = Event is disabled. Corresponding bit in the event enable register (EDMACC_EER) is cleared.

Table 10-189. Register Call Summary for EDMACC_EECR

EDMA Debug/Programming Tips • Miscellaneous Programming/Debug Tips: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_EECR Register (Offset = 1028h) [reset = 0h]: [0][1][2] • EDMACC_EER Register (Offset = 1020h) [reset = 0h]: [0]
EDMA Channel Controller Regions • Region Overview: [0]

10.6.1.5.12 EDMACC_EECRH Register (Offset = 102Ch) [reset = 0h]

The EDMACC_EECRH is shown in Figure 10-95 and described in Table 10-191.

Table 10-190. EDMACC_EECRH Instances

Instance	Physical Address
EDMACC_0	0270 102Ch
EDMACC_1	0272 902Ch

Figure 10-95. EDMACC_EECRH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63-E32																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 10-191. EDMACC_EECRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	E63-E32	W	0h	Event enable clear for events 63-32. 0h = No effect. 1h = Event is disabled. Corresponding bit in the event enable register high (EDMACC_EERH) is cleared (EN = 0).

Table 10-192. Register Call Summary for EDMACC_EECRH

EDMA Debug/Programming Tips • Miscellaneous Programming/Debug Tips: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_EECR Register (Offset = 1028h) [reset = 0h]: [0][1] • EDMACC_EECRH Register (Offset = 102Ch) [reset = 0h]: [0] • EDMACC_EER Register (Offset = 1020h) [reset = 0h]: [0]
EDMA Channel Controller Regions • Region Overview: [0]

10.6.1.5.13 EDMACC_EESR Register (Offset = 1030h) [reset = 0h]

The event enable registers (EDMACC_EER/EDMACC_EERH) cannot be modified by directly writing to them. The intent is to ease the software burden for the case where multiple tasks are attempting to simultaneously modify these registers. The event enable set registers (EDMACC_EESR/EDMACC_EESRH) are used to enable events. Writes of 1 to the bits in EDMACC_EESR/EDMACC_EESRH set the corresponding event bits in EDMACC_EER/EDMACC_EERH; writes of 0 have no effect.

The EDMACC_EESR is shown in Figure 10-96 and described in Table 10-194.

Table 10-193. EDMACC_EESR Instances

Instance	Physical Address
EDMACC_0	0270 1030h
EDMACC_1	0272 9030h

Figure 10-96. EDMACC_EESR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31-E0																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 10-194. EDMACC_EESR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	E31-E0	W	0h	Event enable set for events 31-0. 0h = No effect. 1h = Event is enabled. Corresponding bit in the event enable register (EDMACC_EER) is set (EN = 1).

Table 10-195. Register Call Summary for EDMACC_EESR

EDMA Debug/Programming Tips • Miscellaneous Programming/Debug Tips: [0] • Setting Up a Transfer: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_EESR Register (Offset = 1030h) [reset = 0h]: [0][1][2] • EDMACC_EER Register (Offset = 1020h) [reset = 0h]: [0]
EDMA Channel Controller Regions • Region Overview: [0]
Memory Protection • Write access to shadow region 7's event enable set register (EDMACC_EESR): [0][1][2] • Access Denied to an EDMACC Register: [0][1][2] • Active Memory Protection: [0][1][2][3][4][5][6]

10.6.1.5.14 EDMACC_EESRH Register (Offset = 1034h) [reset = 0h]

The EDMACC_EESRH is shown in Figure 10-97 and described in Table 10-197.

Table 10-196. EDMACC_EESRH Instances

Instance	Physical Address
EDMACC_0	0270 1034h
EDMACC_1	0272 9034h

Figure 10-97. EDMACC_EESRH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63-E32																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 10-197. EDMACC_EESRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	E63-E32	W	0h	Event enable set for events 63-32. 0h = No effect. 1h = Event is enabled. Corresponding bit in the event enable register high (EDMACC_EERH) is set (EN = 1)

Table 10-198. Register Call Summary for EDMACC_EESRH

EDMA Debug/Programming Tips • Miscellaneous Programming/Debug Tips: [0] • Setting Up a Transfer: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_EESR Register (Offset = 1030h) [reset = 0h]: [0][1] • EDMACC_EESRH Register (Offset = 1034h) [reset = 0h]: [0] • EDMACC_EER Register (Offset = 1020h) [reset = 0h]: [0]
EDMA Channel Controller Regions • Region Overview: [0]

10.6.1.5.15 EDMACC_SER Register (Offset = 1038h) [reset = 0h]

The secondary event registers (EDMACC_SER/EDMACC_SERH) provide information on the state of a DMA channel or event (0 through 63). If the EDMACC receives a TR synchronization due to a manual-trigger, event-trigger, or chained-trigger source (EDMACC_ESR.E *n*/EDMACC_ESRH.E *n* = 1, EDMACC_ER.E *n*/EDMACC_ERH.E *n* = 1, or EDMACC_CER.E *n*/EDMACC_CERH.E *n* = 1), which results in the setting of a corresponding event bit in EDMACC_SER/EDMACC_SERH (EDMACC_SER.E *n*/EDMACC_SERH.E *n* = 1), it implies that the corresponding DMA event is in the queue.

Once a bit corresponding to an event is set in EDMACC_SER/EDMACC_SERH, the EDMACC does not prioritize additional events on the same DMA channel. Depending on the condition that lead to the setting of the EDMACC_SER bits, either the EDMACC hardware or the software (using EDMACC_SECR/EDMACC_SECRH) needs to clear the EDMACC_SER/EDMACC_SERH bits for the EDMACC to evaluate subsequent events (subsequent transfers) on the same channel. See for additional conditions that can cause the secondary event registers to be set.

The EDMACC_SERH is shown in Figure 10-99 and described in Table 10-203.

Table 10-199. EDMACC_SER Instances

Instance	Physical Address
EDMACC_0	0270 1038h
EDMACC_1	0272 9038h

Figure 10-98. EDMACC_SER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SER31-SER0																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-200. EDMACC_SER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SER31-SER0	R	0h	Secondary event register. The secondary event register is used along with the event register (EDMACC_ER) to provide information on the state of an event. 0h = Event is not currently stored in the event queue. 1h = Event is currently stored in the event queue. Event arbiter will not prioritize additional events.

Table 10-201. Register Call Summary for EDMACC_SER

EDMA Debug/Programming Tips • Debug Checklist: [0][1][2]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_QSECR Register (Offset = 1094h) [reset = 0h]: [0] • EDMACC_QSER Register (Offset = 1090h) [reset = 0h]: [0] • EDMACC_SER Register (Offset = 1038h) [reset = 0h]: [0][1][2][3][4][5] • EDMACC_SECR Register (Offset = 1040h) [reset = 0h]: [0][1][2]
EDMA Channel Controller Regions • Region Overview: [0]
EDMA Functional Description • Event Dataflow: [0][1][2]

Table 10-201. Register Call Summary for EDMACC_SER (continued)

Parameter RAM (PaRAM) • Dummy Versus Null Transfer Comparison: [0][1] • Dummy PaRAM Set: [0][1] • Null PaRAM Set: [0][1][2]

10.6.1.5.16 EDMACC_SERH Register (Offset = 103Ch) [reset = 0h]

The EDMACC_SERH is shown in Figure 10-99 and described in Table 10-203.

Table 10-202. EDMACC_SERH Instances

Instance	Physical Address
EDMACC_0	0270 103Ch
EDMACC_1	0272 903Ch

Figure 10-99. EDMACC_SERH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SER63-SER32																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-203. EDMACC_SERH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SER63-SER32	R	0h	Secondary event register. The secondary event register is used along with the event register high (EDMACC_ERH) to provide information on the state of an event. 0h = Event is not currently stored in the event queue. 1h = Event is currently stored in the event queue. Event submission/prioritization logic will not prioritize additional events.

Table 10-204. Register Call Summary for EDMACC_SERH

EDMA Debug/Programming Tips • Debug Checklist: [0][1]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_SECRH Register (Offset = 1044h) [reset = 0h]: [0] • EDMACC_SER Register (Offset = 1038h) [reset = 0h]: [0][1][2][3][4][5] • EDMACC_SERH Register (Offset = 103Ch) [reset = 0h]: [0] • EDMACC_SECR Register (Offset = 1040h) [reset = 0h]: [0][1]
EDMA Channel Controller Regions • Region Overview: [0]
EDMA Functional Description • Event Dataflow: [0][1][2]
Parameter RAM (PaRAM) • Dummy Versus Null Transfer Comparison: [0] • Dummy PaRAM Set: [0] • Null PaRAM Set: [0][1]

10.6.1.5.17 EDMACC_SECR Register (Offset = 1040h) [reset = 0h]

The secondary event clear registers (EDMACC_SECR/EDMACC_SECRH) clear the status of the secondary event registers (EDMACC_SER/EDMACC_SERH). DSP writes of 1 clear the corresponding set bits in EDMACC_SER/EDMACC_SERH. Writes of 0 have no effect.

The EDMACC_SECR is shown in Figure 10-100 and described in Table 10-206.

Table 10-205. EDMACC_SECR Instances

Instance	Physical Address
EDMACC_0	0270 1040h
EDMACC_1	0272 9040h

Figure 10-100. EDMACC_SECR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SECR31-SECR0																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 10-206. EDMACC_SECR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SECR31-SECR0	W	0h	Secondary event clear register 0h = No effect. 1h = Corresponding bit in the secondary event register (EDMACC_SER) is cleared.

Table 10-207. Register Call Summary for EDMACC_SECR

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_QSECR Register (Offset = 1094h) [reset = 0h]: [0] • EDMACC_SER Register (Offset = 1038h) [reset = 0h]: [0] • EDMACC_SECR Register (Offset = 1040h) [reset = 0h]: [0][1]
EDMA Channel Controller Regions • Region Overview: [0]

10.6.1.5.18 EDMACC_SECRH Register (Offset = 1044h) [reset = 0h]

The EDMACC_SECRH is shown in Figure 10-101 and described in Table 10-209.

Table 10-208. EDMACC_SECRH Instances

Instance	Physical Address
EDMACC_0	0270 1044h
EDMACC_1	0272 9044h

Figure 10-101. EDMACC_SECRH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SECR63-SECR32																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 10-209. EDMACC_SECRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SECR63-SECR32	W	0h	Secondary event clear register. 1h = No effect. 2h = Corresponding bit in the secondary event registers high (EDMACC_SERH) is cleared.

Table 10-210. Register Call Summary for EDMACC_SECRH

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_SECRH Register (Offset = 1044h) [reset = 0h]: [0] • EDMACC_SER Register (Offset = 1038h) [reset = 0h]: [0] • EDMACC_SECR Register (Offset = 1040h) [reset = 0h]: [0]
EDMA Channel Controller Regions • Region Overview: [0]

10.6.1.6 Interrupt Registers

All DMA/QDMA channels can be set to assert an EDMACC completion interrupt to the DSP on transfer completion, by appropriately configuring the PaRAM entry associated with the channels. The following set of registers is used for the transfer completion interrupt reporting/generating by the EDMACC. See for more details on EDMACC completion interrupt generation.

10.6.1.6.1 EDMACC_IER Register (Offset = 1050h) [reset = 0h]

Interrupt enable registers (EDMACC_IER/EDMACC_IERH) are used to enable/disable the transfer completion interrupt generation by the EDMACC for all DMA/QDMA channels. The EDMACC_IER/EDMACC_IERH cannot be written to directly. To set any interrupt bit in EDMACC_IER/EDMACC_IERH, a 1 must be written to the corresponding interrupt bit in the interrupt enable set registers (EDMACC_IESR/EDMACC_IESRH). Similarly, to clear any interrupt bit in EDMACC_IER/EDMACC_IERH, a 1 must be written to the corresponding interrupt bit in the interrupt enable clear registers (EDMACC_IECR/EDMACC_IECRH).

The EDMACC_IER is shown in Figure 10-102 and described in Table 10-212.

Table 10-211. EDMACC_IER Instances

Instance	Physical Address
EDMACC_0	0270 1050h
EDMACC_1	0272 9050h

Figure 10-102. EDMACC_IER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IER31-IER0																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-212. EDMACC_IER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IER31-IER0	R	0h	Interrupt enable for channels 31-0. 0h = Interrupt is not enabled. 1h = Interrupt is enabled.

Table 10-213. Register Call Summary for EDMACC_IER

EDMA Debug/Programming Tips • Debug Checklist: [0][1] • Setting Up a Transfer: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_IEVAL Register (Offset = 1078h) [reset = 0h]: [0] • EDMACC_IER Register (Offset = 1050h) [reset = 0h]: [0][1][2][3][4] • EDMACC_IECR Register (Offset = 1058h) [reset = 0h]: [0][1] • EDMACC_IESR Register (Offset = 1060h) [reset = 0h]: [0][1]
EDMA Interrupts • Enabling Transfer Completion Interrupts: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17] • Transfer Completion Interrupts: [0][1] • Interrupt Evaluation Operations: [0]
Peripheral Servicing Example • Servicing Input/Output FIFOs with a Single Event: [0]

Table 10-213. Register Call Summary for EDMACC_IER (continued)

EDMA Channel Controller Regions • Region Interrupts: [0] • Region Overview: [0]
Parameter RAM (PaRAM) • Channel Options Parameter (OPT): [0][1]

10.6.1.6.2 EDMACC_IERH Register (Offset = 1054h) [reset = 0h]

The EDMACC_IERH is shown in Figure 10-103 and described in Table 10-215.

Table 10-214. EDMACC_IERH Instances

Instance	Physical Address
EDMACC_0	0270 1054h
EDMACC_1	0272 9054h

Figure 10-103. EDMACC_IERH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IER63-IER32																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-215. EDMACC_IERH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IER63-IER32	R	0h	Interrupt enable for channels 63-32. 0h = Interrupt is not enabled. 1h = Interrupt is enabled.

Table 10-216. Register Call Summary for EDMACC_IERH

EDMA Debug/Programming Tips • Debug Checklist: [0][1] • Setting Up a Transfer: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_IERH Register (Offset = 1054h) [reset = 0h]: [0] • EDMACC_IESRH Register (Offset = 1064h) [reset = 0h]: [0] • EDMACC_IECRH Register (Offset = 105Ch) [reset = 0h]: [0] • EDMACC_IEVAL Register (Offset = 1078h) [reset = 0h]: [0] • EDMACC_IER Register (Offset = 1050h) [reset = 0h]: [0][1][2][3] • EDMACC_IECR Register (Offset = 1058h) [reset = 0h]: [0] • EDMACC_IESR Register (Offset = 1060h) [reset = 0h]: [0]
EDMA Interrupts • Enabling Transfer Completion Interrupts: [0][1][2] • Transfer Completion Interrupts: [0] • Interrupt Evaluation Operations: [0]
EDMA Channel Controller Regions • Region Overview: [0]
Parameter RAM (PaRAM) • Channel Options Parameter (OPT): [0][1]

10.6.1.6.3 EDMACC_IECR Register (Offset = 1058h) [reset = 0h]

The interrupt enable clear registers (EDMACC_IECR/EDMACC_IECRH) are used to clear interrupts. Writes of 1 to the bits in EDMACC_IECR/EDMACC_IECRH clear the corresponding interrupt bits in the interrupt enable registers (EDMACC_IER/EDMACC_IERH); writes of 0 have no effect.

The EDMACC_IECR is shown in Figure 10-104 and described in Table 10-218.

Table 10-217. EDMACC_IECR Instances

Instance	Physical Address
EDMACC_0	0270 1058h
EDMACC_1	0272 9058h

Figure 10-104. EDMACC_IECR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IECR31-IECR0																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 10-218. EDMACC_IECR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IECR31-IECR0	W	0h	Interrupt enable clear for channels 31-0. 0h = No effect 1h = Corresponding bit in the interrupt enable register (EDMACC_IER) is cleared.

Table 10-219. Register Call Summary for EDMACC_IECR

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_IER Register (Offset = 1050h) [reset = 0h]: [0] • EDMACC_IECR Register (Offset = 1058h) [reset = 0h]: [0][1][2]
EDMA Interrupts • Enabling Transfer Completion Interrupts: [0]
EDMA Channel Controller Regions • Region Overview: [0]

10.6.1.6.4 EDMACC_IECRH Register (Offset = 105Ch) [reset = 0h]

The EDMACC_IECRH is shown in Figure 10-105 and described in Table 10-221.

Table 10-220. EDMACC_IECRH Instances

Instance	Physical Address
EDMACC_0	0270 105Ch
EDMACC_1	0272 905Ch

Figure 10-105. EDMACC_IECRH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IECR63-IECR32																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 10-221. EDMACC_IECRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IECR63-IECR32	W	0h	Interrupt enable clear for channels 63-32. 0h = No effect. 1h = Corresponding bit in the interrupt enable register high (EDMACC_IERH) is cleared.

Table 10-222. Register Call Summary for EDMACC_IECRH

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_IER Register (Offset = 1050h) [reset = 0h]: [0] • EDMACC_IECR Register (Offset = 1058h) [reset = 0h]: [0][1] • EDMACC_IECRH Register (Offset = 105Ch) [reset = 0h]: [0]
EDMA Channel Controller Regions • Region Overview: [0]

10.6.1.6.5 EDMACC_IESR Register (Offset = 1060h) [reset = 0h]

The interrupt enable set registers (EDMACC_IESR/EDMACC_IESRH) are used to enable interrupts. Writes of 1 to the bits in EDMACC_IESR/EDMACC_IESRH set the corresponding interrupt bits in the interrupt enable registers (EDMACC_IER/EDMACC_IERH); writes of 0 have no effect.

The EDMACC_IESR is shown in Figure 10-106 and described in Table 10-224.

Table 10-223. EDMACC_IESR Instances

Instance	Physical Address
EDMACC_0	0270 1060h
EDMACC_1	0272 9060h

Figure 10-106. EDMACC_IESR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IESR31-IESR0																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 10-224. EDMACC_IESR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IESR31-IESR0	W	0h	Interrupt enable set for channels 31-0. 0h = No effect. 1h = Corresponding bit in the interrupt enable register (EDMACC_IER) is set (IN = 1).

Table 10-225. Register Call Summary for EDMACC_IESR

EDMA Debug/Programming Tips • Setting Up a Transfer: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_IER Register (Offset = 1050h) [reset = 0h]: [0] • EDMACC_IESR Register (Offset = 1060h) [reset = 0h]: [0][1][2]
EDMA Interrupts • Enabling Transfer Completion Interrupts: [0]
EDMA Channel Controller Regions • Region Overview: [0]

10.6.1.6.6 EDMACC_IESRH Register (Offset = 1064h) [reset = 0h]

The EDMACC_IESRH is shown in Figure 10-107 and described in Table 10-227.

Table 10-226. EDMACC_IESRH Instances

Instance	Physical Address
EDMACC_0	0270 1064h
EDMACC_1	0272 9064h

Figure 10-107. EDMACC_IESRH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IESR63-IESR32																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 10-227. EDMACC_IESRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IESR63-IESR32	W	0h	Interrupt enable clear for channels 63-32. 0h = No effect. 1h = Corresponding bit in the interrupt enable register high (EDMACC_IERH) is set (IN = 1).

Table 10-228. Register Call Summary for EDMACC_IESRH

EDMA Debug/Programming Tips • Setting Up a Transfer: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_IER Register (Offset = 1050h) [reset = 0h]: [0] • EDMACC_IESR Register (Offset = 1060h) [reset = 0h]: [0][1] • EDMACC_IESRH Register (Offset = 1064h) [reset = 0h]: [0]
EDMA Channel Controller Regions • Region Overview: [0]

10.6.1.6.7 EDMACC_IPR Register (Offset = 1068h) [reset = 0h]

If the TCINTEN and/or ITCINTEN bit in the channel option parameter (OPT) is set in the PaRAM entry associated with the channel (DMA or QDMA), then the EDMATC (for normal completion) or the EDMACC (for early completion) returns a completion code on transfer or intermediate transfer completion. The value of the returned completion code is equal to the TCC bit in OPT for the PaRAM entry associated with the channel.

When an interrupt transfer completion code with TCC = *n* is detected by the EDMACC, then the corresponding bit is set in the interrupt pending register (EDMACC_IPR.I *n*, if N = 0 to 31; EDMACC_IPRH.I *n*, if N = 32 to 63). Note that once a bit is set in the interrupt pending registers, it remains set; it is your responsibility to clear these bits. The bits set in EDMACC_IPR/EDMACC_IPRH are cleared by writing a 1 to the corresponding bits in the interrupt clear registers (EDMACC_ICR/EDMACC_ICRH).

The EDMACC_IPR is shown in Figure 10-108 and described in Table 10-230.

Table 10-229. EDMACC_IPR Instances

Instance	Physical Address
EDMACC_0	0270 1068h
EDMACC_1	0272 9068h

Figure 10-108. EDMACC_IPR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR31-IPR0																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-230. EDMACC_IPR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IPR31-IPR0	R	0h	Interrupt pending for TCC = 31-0. 0h = Interrupt transfer completion code is not detected or was cleared. 1h = Interrupt transfer completion code is detected (IN = 1, N = TCC[5:0]).

Table 10-231. Register Call Summary for EDMACC_IPR

EDMA Debug/Programming Tips • Debug Checklist: [0][1] • Setting Up a Transfer: [0][1][2][3]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_IPR Register (Offset = 1068h) [reset = 0h]: [0][1][2] • EDMACC_IEVAL Register (Offset = 1078h) [reset = 0h]: [0] • EDMACC_ICR Register (Offset = 1070h) [reset = 0h]: [0][1][2]
EDMA Interrupts • Enabling Transfer Completion Interrupts: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14] • Transfer Completion Interrupts: [0][1][2] • Interrupt Servicing Example 1: [0][1][2][3][4][5] • EDMA Interrupt Servicing: [0][1][2][3][4] • Interrupt Evaluation Operations: [0] • Interrupt Servicing Example 2: [0][1][2][3][4][5] • Clearing Transfer Completion Interrupts: [0][1][2]

Table 10-231. Register Call Summary for EDMACC_IPR (continued)

Peripheral Servicing Example • Synchronization with the DSP: [0][1][2] • Servicing Input/Output FIFOs with a Single Event: [0]
EDMA Channel Controller Regions • Region Overview: [0]
EDMA Functional Description • Event Dataflow: [0][1] • Completion of a DMA Transfer: [0]
Parameter RAM (PaRAM) • Channel Options Parameter (OPT): [0][1][2][3] • Dummy Versus Null Transfer Comparison: [0]
Completion of a DMA Transfer • Dummy or Null Completion: [0]

10.6.1.6.8 EDMACC_IPRH Register (Offset = 106Ch) [reset = 0h]

The EDMACC_IPRH is shown in Figure 10-109 and described in Table 10-233.

Table 10-232. EDMACC_IPRH Instances

Instance	Physical Address
EDMACC_0	0270 106Ch
EDMACC_1	0272 906Ch

Figure 10-109. EDMACC_IPRH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR63-IPR32																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-233. EDMACC_IPRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IPR63-IPR32	R	0h	Interrupt pending for TCC = 63-32. 0h = Interrupt transfer completion code is not detected or was cleared. 1h = Interrupt transfer completion code is detected (IN = 1, N = TCC[5:0]).

Table 10-234. Register Call Summary for EDMACC_IPRH

EDMA Debug/Programming Tips • Debug Checklist: [0][1] • Setting Up a Transfer: [0][1][2][3]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_IPR Register (Offset = 1068h) [reset = 0h]: [0][1] • EDMACC_IPRH Register (Offset = 106Ch) [reset = 0h]: [0] • EDMACC_IEVAL Register (Offset = 1078h) [reset = 0h]: [0] • EDMACC_ICR Register (Offset = 1070h) [reset = 0h]: [0][1] • EDMACC_ICRH Register (Offset = 1074h) [reset = 0h]: [0]
EDMA Interrupts • Enabling Transfer Completion Interrupts: [0][1][2][3] • Transfer Completion Interrupts: [0][1][2] • Interrupt Servicing Example 1: [0][1][2][3][4][5] • EDMA Interrupt Servicing: [0][1][2][3][4] • Interrupt Evaluation Operations: [0] • Interrupt Servicing Example 2: [0][1][2][3][4][5] • Clearing Transfer Completion Interrupts: [0][1]
EDMA Channel Controller Regions • Region Overview: [0]
EDMA Functional Description • Event Dataflow: [0][1]
Parameter RAM (PaRAM) • Channel Options Parameter (OPT): [0][1][2][3] • Dummy Versus Null Transfer Comparison: [0]
Completion of a DMA Transfer • Dummy or Null Completion: [0]

10.6.1.6.9 EDMACC_ICR Register (Offset = 1070h) [reset = 0h]

The bits in the interrupt pending registers (EDMACC_IPR/EDMACC_IPRH) are cleared by writing a 1 to the corresponding bits in the interrupt clear registers (EDMACC_ICR/EDMACC_ICRH). Writes of 0 have no effect. All set bits in EDMACC_IPR/EDMACC_IPRH must be cleared to allow EDMACC to assert additional transfer completion interrupts.

The EDMACC_ICR is shown in Figure 10-110 and described in Table 10-236.

Table 10-235. EDMACC_ICR Instances

Instance	Physical Address
EDMACC_0	0270 1070h
EDMACC_1	0272 9070h

Figure 10-110. EDMACC_ICR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ICR31-ICR0																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 10-236. EDMACC_ICR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ICR31-ICR0	W	0h	Interrupt clear register for TCC = 31-0. 0h = No effect. 1h = Corresponding bit in the interrupt pending register (EDMACC_IPR) is cleared (IN = 0).

Table 10-237. Register Call Summary for EDMACC_ICR

EDMA Debug/Programming Tips • Setting Up a Transfer: [0][1]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_IPR Register (Offset = 1068h) [reset = 0h]: [0] • EDMACC_ICR Register (Offset = 1070h) [reset = 0h]: [0][1]
EDMA Interrupts • Interrupt Servicing Example 1: [0] • Clearing Transfer Completion Interrupts: [0][1]
EDMA Channel Controller Regions • Region Overview: [0]

10.6.1.6.10 EDMACC_ICRH Register (Offset = 1074h) [reset = 0h]

The EDMACC_ICRH is shown in Figure 10-111 and described in Table 10-239.

Table 10-238. EDMACC_ICRH Instances

Instance	Physical Address
EDMACC_0	0270 1074h
EDMACC_1	0272 9074h

Figure 10-111. EDMACC_ICRH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ICR63-ICR32																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 10-239. EDMACC_ICRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ICR63-ICR32	W	0h	Interrupt clear register for TCC = 63-32. 0h = No effect. 1h = Corresponding bit in the interrupt pending register high (EDMACC_IPRH) is cleared.

Table 10-240. Register Call Summary for EDMACC_ICRH

EDMA Debug/Programming Tips • Setting Up a Transfer: [0][1]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_IPR Register (Offset = 1068h) [reset = 0h]: [0] • EDMACC_ICR Register (Offset = 1070h) [reset = 0h]: [0] • EDMACC_ICRH Register (Offset = 1074h) [reset = 0h]: [0]
EDMA Interrupts • Interrupt Servicing Example 1: [0] • Clearing Transfer Completion Interrupts: [0]
EDMA Channel Controller Regions • Region Overview: [0]

10.6.1.6.11 EDMACC_IEVAL Register (Offset = 1078h) [reset = 0h]

The interrupt evaluate register (EDMACC_IEVAL) is the only register that physically exists in both the global region and the shadow regions. In other words, the read/write accessibility for the shadow region EDMACC_IEVAL is not affected by the DMA/QDMA region access registers (DRAE *m*/DRAEH *m*, QRAE *n*/QRAEH *n*). EDMACC_IEVAL is needed for robust ISR operations to ensure that interrupts are not missed by the DSP.

The EDMACC_IEVAL is shown in Figure 10-112 and described in Table 10-242.

Table 10-241. EDMACC_IEVAL Instances

Instance	Physical Address
EDMACC_0	0270 1078h
EDMACC_1	0272 9078h

Figure 10-112. EDMACC_IEVAL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						SET	EVAL
R-0h						W-0h	W-0h

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 10-242. EDMACC_IEVAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	SET	W	0h	Always write 0. Writes of 1 to this bit are not supported and attempts to do so may result in undefined behavior.
0	EVAL	W	0h	Interrupt evaluate 0h = No effect. 1h = Causes EDMACC completion interrupt to be pulsed, if any enabled (EDMACC_IER *n*/EDMACC_IERH *n* = 1) interrupts are still pending (EDMACC_IPR *n*/EDMACC_IPRH *n* = 1). The EDMACC completion interrupt that is pulsed depends on which EDMACC_IEVAL is being exercised. For example, writing to the EVAL bit in EDMACC_IEVAL pulses the global completion interrupt, but writing to the EVAL bit in IEVAL0 pulses the region 0 completion interrupt.

Table 10-243. Register Call Summary for EDMACC_IEVAL

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_IEVAL Register (Offset = 1078h) [reset = 0h]: [0][1][2][3][4][5]

Table 10-243. Register Call Summary for EDMACC_IEVAL (continued)

EDMA Interrupts • EDMA Interrupt Servicing: [0] • Interrupt Evaluation Operations: [0][1][2] • Interrupt Servicing Example 2: [0] • Error Interrupts: [0]
EDMA Channel Controller Regions • : [1]

10.6.1.7 QDMA Registers

The following sets of registers control the QDMA channels in the EDMACC. The QDMA channels (with the exception of the QDMA queue number register) consist of a set of registers, each of which have a bit location. Each bit position corresponds to a QDMA channel number. The QDMA channel registers are accessible via read/writes to the global address range. They are also accessible via read/writes to the shadow address range. The read/write accessibility in the shadow region address region is controlled by the QDMA region access registers (QRAE *n*/QRAEH *n*). details shadow region/global region usage.

10.6.1.7.1 EDMACC_QER Register (Offset = 1080h) [reset = 0h]

The QDMA event register (EDMACC_QER) channel *n* bit is set (E *n* = 1) when the DSP or any EDMA programmer (including EDMA) performs a write to the trigger word (using the QDMA channel mapping register (QCHMAP *n*)) in the PaRAM entry associated with QDMA channel n (which is also programmed using QCHMAP *n*). The E *n* bit is also set when the EDMACC performs a link update on a PaRAM address that matches the QCHMAP *n* settings. The QDMA event is latched only if the QDMA event enable register (EDMACC_QEER) channel *n* bit is also enabled (EDMACC_QEER.E *n* = 1). Once a bit is set in EDMACC_QER, then the corresponding QDMA event (auto-trigger) is evaluated by the EDMACC logic for an associated transfer request submission to the transfer controllers. See for additional conditions that can lead to the setting of EDMACC_QER bits.

The setting of an event is a higher priority relative to clear operations (via hardware). If set and clear conditions occur concurrently, the set condition wins. If the event was previously set, then the QDMA event missed register (EDMACC_QEMR) would be set because an event is lost. If the event was previously clear, then the event remains set and is prioritized for submission to the event queues.

The set bits in EDMACC_QER are only cleared when the transfer request associated with the corresponding channels has been processed by the EDMACC and submitted to the transfer controller. If the E *n* bit is already set and a QDMA event for the same QDMA channel occurs prior to the original being cleared, then the second missed event is latched in EDMACC_QEMR (E *n* = 1).

The EDMACC_QER is shown in Figure 10-113 and described in Table 10-245.

Table 10-244. EDMACC_QER Instances

Instance	Physical Address
EDMACC_0	0270 1080h
EDMACC_1	0272 9080h

Figure 10-113. EDMACC_QER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								QER7-QER0							
R-0h																								R-0h							

LEGEND: R = Read Only; -*n* = value after reset

Table 10-245. EDMACC_QER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	QER7-QER0	R	0h	QDMA event for channels 7-0. 0h = No effect. 1h = Corresponding QDMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMATC.

Table 10-246. Register Call Summary for EDMACC_QER

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_QSECR Register (Offset = 1094h) [reset = 0h]: [0][1][2] • EDMACC_CCSTAT Register (Offset = 640h) [reset = 0h]: [0] • EDMACC_QER Register (Offset = 1080h) [reset = 0h]: [0][1][2][3][4] • EDMACC_QEER Register (Offset = 1084h) [reset = 0h]: [0][1][2] • EDMACC_Q0E0 to EDMACC_Q3E15 Register (Offset = 400h to 4FCh) [reset = 0h]: [0]
EDMA Channel Controller Regions • Region Overview: [0]
EDMA Prioritization • Channel Priority: [0]
EDMA Functional Description • Event Dataflow: [0][1]
Parameter RAM (PaRAM) • Linking Transfers: [0] • Dummy Versus Null Transfer Comparison: [0]
Initiating a DMA Transfer • Auto-triggered and Link-Triggered Transfer Request: [0][1][2][3][4]

10.6.1.7.2 EDMACC_QEER Register (Offset = 1084h) [reset = 0h]

The EDMACC provides the option of selectively enabling/disabling each channel in the QDMA event register (EDMACC_QER) by using the QDMA event enable register (EDMACC_QEER). If any of the event bits in EDMACC_QEER is set (using the QDMA event enable set register, EDMACC_QEESR), it will enable that corresponding event. Alternatively, if any event bit in EDMACC_QEER is cleared (using the QDMA event enable clear register, EDMACC_QEECR), it will disable the corresponding QDMA channel. The QDMA event register will not latch any event for a QDMA channel, if it is not enabled via EDMACC_QEER.

The EDMACC_QEER is shown in Figure 10-114 and described in Table 10-248.

Table 10-247. EDMACC_QEER Instances

Instance	Physical Address
EDMACC_0	0270 1084h
EDMACC_1	0272 9084h

Figure 10-114. EDMACC_QEER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								QEER7-QEER0							
R-0h																								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 10-248. EDMACC_QEER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	QEER7-QEER0	R	0h	QDMA event enable for channels 7-0. 0h = QDMA channel N is not enabled. QDMA event is not recognized and is not latched in the QDMA event register (EDMACC_QER). 1h = QDMA channel N is enabled. QDMA events are recognized and are latched in the QDMA event register (EDMACC_QER).

Table 10-249. Register Call Summary for EDMACC_QEER

EDMA Debug/Programming Tips • Debug Checklist: [0] • Setting Up a Transfer: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_QER Register (Offset = 1080h) [reset = 0h]: [0][1] • EDMACC_QEESR Register (Offset = 108Ch) [reset = 0h]: [0][1][2] • EDMACC_QEECR Register (Offset = 1088h) [reset = 0h]: [0][1][2] • EDMACC_QEER Register (Offset = 1084h) [reset = 0h]: [0][1][2][3][4]
EDMA Channel Controller Regions • Region Overview: [0] • Channel Controller Shadow Regions: [0]
Initiating a DMA Transfer • Auto-triggered and Link-Triggered Transfer Request: [0][1]

10.6.1.7.3 EDMACC_QEECR Register (Offset = 1088h) [reset = 0h]

The QDMA event enable register (EDMACC_QEER) cannot be modified by directly writing to the register, to ease the software burden when multiple tasks are attempting to simultaneously modify these registers. The QDMA event enable clear register (EDMACC_QEECR) is used to disable events. Writes of 1 to the bits in EDMACC_QEECR clear the corresponding QDMA channel bits in EDMACC_QEER; writes of 0 have no effect.

The EDMACC_QEECR is shown in Figure 10-115 and described in Table 10-251.

Table 10-250. EDMACC_QEECR Instances

Instance	Physical Address
EDMACC_0	0270 1088h
EDMACC_1	0272 9088h

Figure 10-115. EDMACC_QEECR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								QEECR7-QEECR0							
R-0h																								W-0h							

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 10-251. EDMACC_QEECR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	QEECR7-QEECR0	W	0h	QDMA event enable clear for channels 7-0. 0h = No effect. 1h = QDMA event is disabled. Corresponding bit in the QDMA event enable register (EDMACC_QEER) is cleared).

Table 10-252. Register Call Summary for EDMACC_QEECR

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_QEECR Register (Offset = 1088h) [reset = 0h]: [0][1][2] • EDMACC_QEER Register (Offset = 1084h) [reset = 0h]: [0]
EDMA Channel Controller Regions • Region Overview: [0]

10.6.1.7.4 EDMACC_QEESR Register (Offset = 108Ch) [reset = 0h]

The QDMA event enable register (EDMACC_QEER) cannot be modified by directly writing to the register, to ease the software burden when multiple tasks are attempting to simultaneously modify these registers. The QDMA event enable set register (EDMACC_QEESR) is used to enable events. Writes of 1 to the bits in EDMACC_QEESR set the corresponding QDMA channel bits in EDMACC_QEER; writes of 0 have no effect.

The EDMACC_QEESR is shown in Figure 10-116 and described in Table 10-254.

Table 10-253. EDMACC_QEESR Instances

Instance	Physical Address
EDMACC_0	0270 108Ch
EDMACC_1	0272 908Ch

Figure 10-116. EDMACC_QEESR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								QEESR7-QEESR0							
R-0h																								W-0h							

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 10-254. EDMACC_QEESR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	QEESR7-QEESR0	W	0h	QDMA event enable set for channels 7-0. 0h = No effect. 1h = QDMA event is enabled. Corresponding bit in the QDMA event enable register (EDMACC_QEER) is set.

Table 10-255. Register Call Summary for EDMACC_QEESR

EDMA Debug/Programming Tips • Setting Up a Transfer: [0]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_QEESR Register (Offset = 108Ch) [reset = 0h]: [0][1][2] • EDMACC_QEER Register (Offset = 1084h) [reset = 0h]: [0]
EDMA Channel Controller Regions • Region Overview: [0] • Channel Controller Shadow Regions: [0]

10.6.1.7.5 EDMACC_QSER Register (Offset = 1090h) [reset = 0h]

The QDMA secondary event register (EDMACC_QSER) provides information on the state of a QDMA event. If at any time a bit corresponding to a QDMA channel is set in EDMACC_QSER, that implies that the corresponding QDMA event is in the queue. Once a bit corresponding to a QDMA channel is set in EDMACC_QSER, the EDMACC does not prioritize additional events on the same QDMA channel. Depending on the condition that lead to the setting of the EDMACC_QSER bits, either the EDMACC hardware or the software (using EDMACC_QSECR) needs to clear the EDMACC_QSER bits for the EDMACC to evaluate subsequent QDMA events on the channel. Based on whether the associated TR request is valid, or it is a null or dummy TR, the implications on the state of EDMACC_QSER and the required user actions to submit another QDMA transfer might be different. See for additional conditions that can cause the secondary event registers (EDMACC_QSER/EDMACC_SER) to be set.

The EDMACC_QSER is shown in Figure 10-117 and described in Table 10-257.

Table 10-256. EDMACC_QSER Instances

Instance	Physical Address
EDMACC_0	0270 1090h
EDMACC_1	0272 9090h

Figure 10-117. EDMACC_QSER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								QSER7-QSER0							
R-0h																								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 10-257. EDMACC_QSER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	QSER7-QSER0	R	0h	QDMA secondary event register for channels 7-0. 0h = QDMA event is not currently stored in the event queue. 1h = QDMA event is currently stored in the event queue. EDMACC does not prioritize additional events.

Table 10-258. Register Call Summary for EDMACC_QSER

EDMA Debug/Programming Tips • Debug Checklist: [0][1]
EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_QSER Register (Offset = 1090h) [reset = 0h]: [0][1][2][3][4][5][6][7] • EDMACC_QSECR Register (Offset = 1094h) [reset = 0h]: [0][1][2]
EDMA Functional Description • Event Dataflow: [0]
Parameter RAM (PaRAM) • Dummy Versus Null Transfer Comparison: [0] • Dummy PaRAM Set: [0] • Null PaRAM Set: [0][1]

10.6.1.7.6 EDMACC_QSECR Register (Offset = 1094h) [reset = 0h]

The QDMA secondary event clear register (EDMACC_QSECR) clears the status of the QDMA secondary event register (EDMACC_QSER) and the QDMA event register (EDMACC_QER). DSP writes of 1 clear the corresponding set bits in EDMACC_QSER and EDMACC_QER. Writes of 0 have no effect. Note that this differs from the secondary event clear register (EDMACC_SECR) operation, which only clears the secondary event register (EDMACC_SER) bits and does not affect the event registers.

The EDMACC_QSECR is shown in Figure 10-118 and described in Table 10-260.

Table 10-259. EDMACC_QSECR Instances

Instance	Physical Address
EDMACC_0	0270 1094h
EDMACC_1	0272 9094h

Figure 10-118. EDMACC_QSECR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								QSECR7-QSECR0							
R-0h																								W-0h							

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 10-260. EDMACC_QSECR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	QSECR7-QSECR0	W	0h	QDMA secondary event clear for channels 7-0. 0h = No effect. 1h = Corresponding bit in the QDMA secondary event register (EDMACC_QSER) and the QDMA event register (EDMACC_QER) is cleared.

Table 10-261. Register Call Summary for EDMACC_QSECR

EDMA Registers • EDMA Channel Controller (EDMACC) Registers: [0][1]
EDMA Channel Controller (EDMACC) Registers • EDMACC_QSER Register (Offset = 1090h) [reset = 0h]: [0] • EDMACC_QSECR Register (Offset = 1094h) [reset = 0h]: [0][1]

10.6.2 EDMA Transfer Controller (EDMATC) Registers

Table 10-262 lists the base address for each of the EDMATC module instances.

Table 10-263 and Table 10-264 list the memory-mapped registers for the EDMA transfer controller (EDMATC). All register offset addresses not listed in Table 10-263 should be considered as reserved locations and the register contents should not be modified.

Table 10-262. EDMATC Instances

Instance	Base Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0000h
EDMACC_0_TC_1 (EDMATC_1)	0276 8000h
EDMACC_1_TC_0 (EDMATC_2)	027B 0000h
EDMACC_1_TC_1 (EDMATC_3)	027B 8000h

Table 10-263. EDMATC Registers (1/2)

Offset	Acronym	Register Name	EDMACC_0_TC_0 Physical Address	EDMACC_0_TC_1 Physical Address	Section
0h	EDMATC_PID	Peripheral Identification Register	0276 0000h	0276 8000h	Section 10.6.2.1
4h	EDMATC_TCCFG	EDMATC Configuration Register	0276 0004h	0276 8004h	Section 10.6.2.2
100h	EDMATC_TCSTAT	EDMATC Channel Status Register	0276 0100h	0276 8100h	Section 10.6.2.3
120h	EDMATC_ERRSTAT	Error Register	0276 0120h	0276 8120h	Section 10.6.2.4.1
124h	EDMATC_ERREN	Error Enable Register	0276 0124h	0276 8124h	Section 10.6.2.4.2
128h	EDMATC_ERRCLR	Error Clear Register	0276 0128h	0276 8128h	Section 10.6.2.4.3
12Ch	EDMATC_ERRDET	Error Details Register	0276 012Ch	0276 812Ch	Section 10.6.2.4.4
130h	EDMATC_ERRCMD	Error Interrupt Command Register	0276 0130h	0276 8130h	Section 10.6.2.4.5
140h	EDMATC_RDRATE	Read Rate Register	0276 0140h	0276 8140h	Section 10.6.2.5
240h	EDMATC_SAOPT	Source Active Options Register	0276 0240h	0276 8240h	Section 10.6.2.6.1
244h	EDMATC_SASRC	Source Active Source Address Register	0276 0244h	0276 8244h	Section 10.6.2.6.2
248h	EDMATC_SACNT	Source Active Count Register	0276 0248h	0276 8248h	Section 10.6.2.6.3
24Ch	EDMATC_SADST	Source Active Destination Address Register	0276 024Ch	0276 824Ch	Section 10.6.2.6.4
250h	EDMATC_SABIDX	Source Active Source B-Index Register	0276 0250h	0276 8250h	Section 10.6.2.6.5
254h	EDMATC_SAMPPRXY	Source Active Memory Protection Proxy Register	0276 0254h	0276 8254h	Section 10.6.2.6.6
258h	EDMATC_SACNTRLD	Source Active Count Reload Register	0276 0258h	0276 8258h	Section 10.6.2.6.7
25Ch	EDMATC_SASRCBREF	Source Active Source Address B-Reference Register	0276 025Ch	0276 825Ch	Section 10.6.2.6.8
260h	EDMATC_SADSTBREF	Source Active Destination Address B-Reference Register	0276 0260h	0276 8260h	Section 10.6.2.6.9
280h	EDMATC_DFCNTRLD	Destination FIFO Set Count Reload	0276 0280h	0276 8280h	Section 10.6.2.6.10

Table 10-263. EDMATC Registers (1/2) (continued)

Offset	Acronym	Register Name	EDMACC_0_TC_0 Physical Address	EDMACC_0_TC_1 Physical Address	Section
284h	EDMATC_DFSRCBREF	Destination FIFO Set Destination Address B Reference	0276 0284h	0276 8284h	Section 10.6.2.6.11
288h	EDMATC_DFDSTBREF	Destination FIFO Set Destination Address B Reference Register	0276 0288h	0276 8288h	Section 10.6.2.6.12
300h	EDMATC_DFOPT0	Destination FIFO Options Register 0	0276 0300h	0276 8300h	Section 10.6.2.6.13
304h	EDMATC_DFSRC0	Destination FIFO Source Address Register 0	0276 0304h	0276 8304h	Section 10.6.2.6.14
308h	EDMATC_DFCNT0	Destination FIFO Count Register 0	0276 0308h	0276 8308h	Section 10.6.2.6.15
30Ch	EDMATC_DFDST0	Destination FIFO Destination Address Register 0	0276 030Ch	0276 830Ch	Section 10.6.2.6.16
310h	EDMATC_DFBIDX0	Destination FIFO BIDX Register 0	0276 0310h	0276 8310h	Section 10.6.2.6.17
314h	EDMATC_DFMPPRXY0	Destination FIFO Memory Protection Proxy Register 0	0276 0314h	0276 8314h	Section 10.6.2.6.18
340h	EDMATC_DFOPT1	Destination FIFO Options Register 1	0276 0340h	0276 8340h	Section 10.6.2.6.19
344h	EDMATC_DFSRC1	Destination FIFO Source Address Register 1	0276 0344h	0276 8344h	Section 10.6.2.6.20
348h	EDMATC_DFCNT1	Destination FIFO Count Register 1	0276 0348h	0276 8348h	Section 10.6.2.6.21
34Ch	EDMATC_DFDST1	Destination FIFO Destination Address Register 1	0276 034Ch	0276 834Ch	Section 10.6.2.6.22
350h	EDMATC_DFBIDX1	Destination FIFO BIDX Register 1	0276 0350h	0276 8350h	Section 10.6.2.6.23
354h	EDMATC_DFMPPRXY1	Destination FIFO Memory Protection Proxy Register 1	0276 0354h	0276 8354h	Section 10.6.2.6.24
380h	EDMATC_DFOPT2	Destination FIFO Options Register 2	0276 0380h	0276 8380h	Section 10.6.2.6.25
384h	EDMATC_DFSRC2	Destination FIFO Source Address Register 2	0276 0384h	0276 8384h	Section 10.6.2.6.26
388h	EDMATC_DFCNT2	Destination FIFO Count Register 2	0276 0388h	0276 8388h	Section 10.6.2.6.27
38Ch	EDMATC_DFDST2	Destination FIFO Destination Address Register 2	0276 038Ch	0276 838Ch	Section 10.6.2.6.28
390h	EDMATC_DFBIDX2	Destination FIFO BIDX Register 2	0276 0390h	0276 8390h	Section 10.6.2.6.29
394h	EDMATC_DFMPPRXY2	Destination FIFO Memory Protection Proxy Register 2	0276 0394h	0276 8394h	Section 10.6.2.6.30
3C0h	EDMATC_DFOPT3	Destination FIFO Options Register 3	0276 03C0h	0276 83C0h	Section 10.6.2.6.31
3C4h	EDMATC_DFSRC3	Destination FIFO Source Address Register 3	0276 03C4h	0276 83C4h	Section 10.6.2.6.32
3C8h	EDMATC_DFCNT3	Destination FIFO Count Register 3	0276 03C8h	0276 83C8h	Section 10.6.2.6.33
3CCh	EDMATC_DFDST3	Destination FIFO Destination Address Register 3	0276 03CCh	0276 83CCh	Section 10.6.2.6.34
3D0h	EDMATC_DFBIDX3	Destination FIFO BIDX Register 3	0276 03D0h	0276 83D0h	Section 10.6.2.6.35
3D4h	EDMATC_DFMPPRXY3	Destination FIFO Memory Protection Proxy Register 3	0276 03D4h	0276 83D4h	Section 10.6.2.6.36

Table 10-264. EDMATC Registers (2/2)

Offset	Acronym	Register Description	EDMACC_1_TC_0 Physical Address	EDMACC_1_TC_1 Physical Address	Section
0h	EDMATC_PID	Peripheral Identification Register	027B 0000h	027B 8000h	Section 10.6.2.1
4h	EDMATC_TCCFG	EDMATC Configuration Register	027B 0004h	027B 8004h	Section 10.6.2.2
100h	EDMATC_TCSTAT	EDMATC Channel Status Register	027B 0100h	027B 8100h	Section 10.6.2.3
120h	EDMATC_ERRSTAT	Error Register	027B 0120h	027B 8120h	Section 10.6.2.4.1
124h	EDMATC_ERREN	Error Enable Register	027B 0124h	027B 8124h	Section 10.6.2.4.2
128h	EDMATC_ERRCLR	Error Clear Register	027B 0128h	027B 8128h	Section 10.6.2.4.3
12Ch	EDMATC_ERRDET	Error Details Register	027B 012Ch	027B 812Ch	Section 10.6.2.4.4
130h	EDMATC_ERRCMD	Error Interrupt Command Register	027B 0130h	027B 8130h	Section 10.6.2.4.5
140h	EDMATC_RDRATE	Read Rate Register	027B 0140h	027B 8140h	Section 10.6.2.5
240h	EDMATC_SAOPT	Source Active Options Register	027B 0240h	027B 8240h	Section 10.6.2.6.1
244h	EDMATC_SASRC	Source Active Source Address Register	027B 0244h	027B 8244h	Section 10.6.2.6.2
248h	EDMATC_SACNT	Source Active Count Register	027B 0248h	027B 8248h	Section 10.6.2.6.3
24Ch	EDMATC_SADST	Source Active Destination Address Register	027B 024Ch	027B 824Ch	Section 10.6.2.6.4
250h	EDMATC_SABIDX	Source Active Source B-Index Register	027B 0250h	027B 8250h	Section 10.6.2.6.5
254h	EDMATC_SAMPPRXY	Source Active Memory Protection Proxy Register	027B 0254h	027B 8254h	Section 10.6.2.6.6
258h	EDMATC_SACNTRLD	Source Active Count Reload Register	027B 0258h	027B 8258h	Section 10.6.2.6.7
25Ch	EDMATC_SASRCBREF	Source Active Source Address B-Reference Register	027B 025Ch	027B 825Ch	Section 10.6.2.6.8
260h	EDMATC_SADSTBREF	Source Active Destination Address B-Reference Register	027B 0260h	027B 8260h	Section 10.6.2.6.9
280h	EDMATC_DFCNTRLD	Destination FIFO Set Count Reload	027B 0280h	027B 8280h	Section 10.6.2.6.10
284h	EDMATC_DFSRCBREF	Destination FIFO Set Destination Address B Reference	027B 0284h	027B 8284h	Section 10.6.2.6.11
288h	EDMATC_DFDSTBREF	Destination FIFO Set Destination Address B Reference Register	027B 0288h	027B 8288h	Section 10.6.2.6.12

Table 10-264. EDMATC Registers (2/2) (continued)

Offset	Acronym	Register Description	EDMACC_1_TC_0 Physical Address	EDMACC_1_TC_1 Physical Address	Section
300h	EDMATC_DFOPT0	Destination FIFO Options Register 0	027B 0300h	027B 8300h	
304h	EDMATC_DFSRC0	Destination FIFO Source Address Register 0	027B 0304h	027B 8304h	
308h	EDMATC_DFCNT0	Destination FIFO Count Register 0	027B 0308h	027B 8308h	
30Ch	EDMATC_DFDST0	Destination FIFO Destination Address Register 0	027B 030Ch	027B 830Ch	
310h	EDMATC_DFBIDX0	Destination FIFO BIDX Register 0	027B 0310h	027B 8310h	
314h	EDMATC_DFMPPRXY0	Destination FIFO Memory Protection Proxy Register 0	027B 0314h	027B 8314h	
340h	EDMATC_DFOPT1	Destination FIFO Options Register 1	027B 0340h	027B 8340h	
344h	EDMATC_DFSRC1	Destination FIFO Source Address Register 1	027B 0344h	027B 8344h	
348h	EDMATC_DFCNT1	Destination FIFO Count Register 1	027B 0348h	027B 8348h	
34Ch	EDMATC_DFDST1	Destination FIFO Destination Address Register 1	027B 034Ch	027B 834Ch	
350h	EDMATC_DFBIDX1	Destination FIFO BIDX Register 1	027B 0350h	027B 8350h	
354h	EDMATC_DFMPPRXY1	Destination FIFO Memory Protection Proxy Register 1	027B 0354h	027B 8354h	
380h	EDMATC_DFOPT2	Destination FIFO Options Register 2	027B 0380h	027B 8380h	
384h	EDMATC_DFSRC2	Destination FIFO Source Address Register 2	027B 0384h	027B 8384h	
388h	EDMATC_DFCNT2	Destination FIFO Count Register 2	027B 0388h	027B 8388h	
38Ch	EDMATC_DFDST2	Destination FIFO Destination Address Register 2	027B 038Ch	027B 838Ch	
390h	EDMATC_DFBIDX2	Destination FIFO BIDX Register 2	027B 0390h	027B 8390h	
394h	EDMATC_DFMPPRXY2	Destination FIFO Memory Protection Proxy Register 2	027B 0394h	027B 8394h	
3C0h	EDMATC_DFOPT3	Destination FIFO Options Register 3	027B 03C0h	027B 83C0h	
3C4h	EDMATC_DFSRC3	Destination FIFO Source Address Register 3	027B 03C4h	027B 83C4h	
3C8h	EDMATC_DFCNT3	Destination FIFO Count Register 3	027B 03C8h	027B 83C8h	
3CCh	EDMATC_DFDST3	Destination FIFO Destination Address Register 3	027B 03CCh	027B 83CCh	

Table 10-264. EDMATC Registers (2/2) (continued)

Offset	Acronym	Register Description	EDMACC_1_TC_0 Physical Address	EDMACC_1_TC_1 Physical Address	Section
3D0h	EDMATC_DFBIDX3	Destination FIFO BIDX Register 3	027B 03D0h	027B 83D0h	
3D4h	EDMATC_DFMPPRXY3	Destination FIFO Memory Protection Proxy Register 3	027B 03D4h	027B 83D4h	

10.6.2.1 EDMATC_PID Register (Offset = 0h) [reset = 40000301h]

The peripheral identification register (EDMATC_PID) is a constant register that uniquely identifies the EDMATC and specific revision of the EDMATC. The EDMATC_PID is shown in Figure 10-119 and described in Table 10-266.

Table 10-265. EDMATC_PID Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0000h
EDMACC_0_TC_1 (EDMATC_1)	0276 8000h
EDMACC_1_TC_0 (EDMATC_2)	027B 0000h
EDMACC_1_TC_1 (EDMATC_3)	027B 8000h

Figure 10-119. EDMATC_PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PID																															
R-40000301h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-266. EDMATC_PID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PID	R	40000301h	TI internal data. Identifies revision of peripheral.

Table 10-267. Register Call Summary for EDMATC_PID

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_PID Register (Offset = 0h) [reset = 40000301h]: [0][1]

10.6.2.2 EDMATC_TCCFG Register (Offset = 4h) [reset = 225h]

The EDMATC configuration register (EDMATC_TCCFG) is shown in Figure 10-120 and described in Table 10-269.

Table 10-268. EDMATC_TCCFG Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0004h
EDMACC_0_TC_1 (EDMATC_1)	0276 8004h
EDMACC_1_TC_0 (EDMATC_2)	027B 0004h
EDMACC_1_TC_1 (EDMATC_3)	027B 8004h

Figure 10-120. EDMATC_TCCFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						DREGDEPTH	
R-0h						R-2h	
7	6	5	4	3	2	1	0
RESERVED		BUSWIDTH		RESERVED		FIFOSIZE	
R-0h		R-2h		R-0h		R-5h	

LEGEND: R = Read Only; *-n* = value after reset

Table 10-269. EDMATC_TCCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9-8	DREGDEPTH	R	2h	Destination register FIFO depth parameterization. 0h - 1h = Reserved 2h = 4 entry 3h = Reserved
7-6	RESERVED	R	0h	Reserved
5-4	BUSWIDTH	R	2h	Destination register FIFO depth parameterization. 0h - 1h =Reserved 2h = 128-bit 3h = 256-bit
3	RESERVED	R	0h	Reserved
2-0	FIFOSIZE	R	5h	FIFO size. 0h - 3h = Reserved 4h = 512-byte FIFO 5h = 1024-byte FIFO 6h - 7h = Reserved

Table 10-270. Register Call Summary for EDMATC_TCCFG

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_TCCFG Register (Offset = 4h) [reset = 225h]: [0]

10.6.2.3 EDMATC_TCSTAT Register (Offset = 100h) [reset = 100h]

The EDMATC channel status register (EDMATC_TCSTAT) is shown in Figure 10-121 and described in Table 10-272.

Table 10-271. EDMATC_TCSTAT Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0100h
EDMACC_0_TC_1 (EDMATC_1)	0276 8100h
EDMACC_1_TC_0 (EDMATC_2)	027B 0100h
EDMACC_1_TC_1 (EDMATC_3)	027B 8100h

Figure 10-121. EDMATC_TCSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			DFSTRTPTR		RESERVED		
R-0h			R-0h		R-2h		
7	6	5	4	3	2	1	0
RESERVED	DSTACTV			RESERVED	WSACTV	SRCACTV	PROGBUSY
R-2h	R-0h			R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 10-272. EDMATC_TCSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-11	DFSTRTPTR	R	0h	Destination FIFO start pointer. Represents the offset to the head entry of the destination register FIFO, in units of entries.
10-7	RESERVED	R	2h	Reserved
6-4	DSTACTV	R	0h	Destination active state. Specifies the number of transfer requests (TRs) that are resident in the destination register FIFO at a given instant. This bit field can be primarily used for advanced debugging. Legal values are constrained by the destination register FIFO depth parameterization (DSTREGDEPTH) parameter. 0h = Destination FIFO is empty. 1h = Destination FIFO contains 1 TR. 2h = Destination FIFO contains 2 TRs. 3h = Destination FIFO contains 3 TRs. 4h = Destination FIFO contains 4 TRs. (Full if DSTREGDEPTH==4) If the destination register FIFO is empty, then any TR written to Prog Set immediately transitions to the destination register FIFO. If the destination register FIFO is not empty and not full, then any TR written to Prog Set immediately transitions to the destination register FIFO set if the source active state (SRCACTV) bit is set to idle. If the destination register FIFO is full, then TRs cannot transition to the destination register FIFO. The destination register FIFO becomes not full when the TR at the head of the destination register FIFO is completed. 5h - 7h = Reserved
3	RESERVED	R	0h	Reserved

Table 10-272. EDMATC_TCSTAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	WSACTV	R	0h	Write status active. 0h = Write status is not pending. Write status has been received for all previously issued write commands. 1h = Write status is pending. Write status has not been received for all previously issued write commands.
1	SRCACTV	R	0h	Source active state. 0h = Source controller is idle. Source active register set contains a previously processed transfer request. 1h = Source controller is busy servicing a transfer request.
0	PROGBUSY	R	0h	Program register set busy. 0h = Program set idle and is available for programming by the EDMACC. 1h = Program set busy.

Table 10-273. Register Call Summary for EDMATC_TCSTAT

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller Operation • Debug Features: [0] • Destination FIFO Register Pointer: [0]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_TCSTAT Register (Offset = 100h) [reset = 100h]: [0]

10.6.2.4 Error Registers

10.6.2.4.1 EDMATC_ERRSTAT Register (Offset = 120h) [reset = 0h]

The error status register (EDMATC_ERRSTAT) is shown in Figure 10-122 and described in Table 10-275.

Table 10-274. EDMATC_ERRSTAT Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0120h
EDMACC_0_TC_1 (EDMATC_1)	0276 8120h
EDMACC_1_TC_0 (EDMATC_2)	027B 0120h
EDMACC_1_TC_1 (EDMATC_3)	027B 8120h

Figure 10-122. EDMATC_ERRSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MMRAERR	TRERR	RESERVED	BUSERR
R-0h				R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 10-275. EDMATC_ERRSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	MMRAERR	R	0h	MMR address error. 0h = Condition is not detected 1h = User attempted to read or write to an invalid address in configuration memory map.
2	TRERR	R	0h	Transfer request (TR) error event. 0h = Condition is not detected. 1h = TR detected that violates constant addressing mode transfer (SAM or DAM is set) alignment rules or has ACNT or BCNT == 0
1	RESERVED	R	0h	Reserved
0	BUSERR	R	0h	Bus error event. 0h = Condition is not detected. 1h = EDMATC has detected an error at source or destination address. Error information can be read from the error details register (EDMATC_ERRDET).

Table 10-276. Register Call Summary for EDMATC_ERRSTAT

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller Operation • Error Generation: [0][1]

Table 10-276. Register Call Summary for EDMATC_ERRSTAT (continued)

EDMA Transfer Controller (EDMATC) Registers
• EDMATC_ERRSTAT Register (Offset = 120h) [reset = 0h]: [0] • EDMATC_ERRCLR Register (Offset = 128h) [reset = 0h]: [0][1][2][3][4][5] • EDMATC_ERREN Register (Offset = 124h) [reset = 0h]: [0] • EDMATC_ERRCMD Register (Offset = 130h) [reset = 0h]: [0]

10.6.2.4.2 EDMATC_ERREN Register (Offset = 124h) [reset = 0h]

The error enable register (EDMATC_ERREN) is shown in Figure 10-123 and described in Table 10-278.

When any of the enable bits are set, a bit set in the corresponding EDMATC_ERRSTAT causes an assertion of the EDMATC interrupt.

Table 10-277. EDMATC_ERREN Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0124h
EDMACC_0_TC_1 (EDMATC_1)	0276 8124h
EDMACC_1_TC_0 (EDMATC_2)	027B 0124h
EDMACC_1_TC_1 (EDMATC_3)	027B 8124h

Figure 10-123. EDMATC_ERREN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MMRAERR	TRERR	RESERVED	BUSERR
R-0h				R/W-0h	R/W-0h	R-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 10-278. EDMATC_ERREN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	MMRAERR	R/W	0h	Interrupt enable for MMR address error (MMRAERR). 0h = MMRAERR is disabled. 1h = MMRAERR is enabled and contributes to the state of EDMATC error interrupt generation
2	TRERR	R/W	0h	Interrupt enable for transfer request error (TRERR). 0h = TRERR is disabled. 1h = TRERR is enabled and contributes to the state of EDMATC error interrupt generation.
1	RESERVED	R	0h	Reserved. Always write 0 to this bit; writes of 1 to this bit are not supported and attempts to do so may result in undefined behavior.
0	BUSERR	R/W	0h	Bus error event. 0h = BUSERR is disabled. 1h = BUSERR is enabled and contributes to the state of EDMATC error interrupt generation.

Table 10-279. Register Call Summary for EDMATC_ERREN

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller Operation • Error Generation: [0]

Table 10-279. Register Call Summary for EDMATC_ERREN (continued)

EDMA Transfer Controller (EDMATC) Registers
• EDMATC_ERREN Register (Offset = 124h) [reset = 0h]: [0]

10.6.2.4.3 EDMATC_ERRCLR Register (Offset = 128h) [reset = 0h]

The error clear register (EDMATC_ERRCLR) is shown in Figure 10-124 and described in Table 10-281.

Table 10-280. EDMATC_ERRCLR Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0128h
EDMACC_0_TC_1 (EDMATC_1)	0276 8128h
EDMACC_1_TC_0 (EDMATC_2)	027B 0128h
EDMACC_1_TC_1 (EDMATC_3)	027B 8128h

Figure 10-124. EDMATC_ERRCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MMRAERR	TRERR	RESERVED	BUSERR
R-0h				W-0h	W-0h	R-0h	W-0h

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 10-281. EDMATC_ERRCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	MMRAERR	W	0h	Interrupt enable clear for the MMRAERR bit in the error status register (EDMATC_ERRSTAT). 0h = No effect. 1h = Clears the MMRAERR bit in EDMATC_ERRSTAT but does not clear the error details register (EDMATC_ERRDET).
2	TRERR	W	0h	Interrupt enable clear for the TRERR bit in the error status register (EDMATC_ERRSTAT). 0h = No effect. 1h = Clears the TRERR bit in EDMATC_ERRSTAT but does not clear the error details register (EDMATC_ERRDET).
1	RESERVED	R	0h	Reserved
0	BUSERR	W	0h	Interrupt clear for the BUSERR bit in the error status register (EDMATC_ERRSTAT). 0h = No effect. 1h = Clears the BUSERR bit in EDMATC_ERRSTAT and clears the error details register (EDMATC_ERRDET).

Table 10-282. Register Call Summary for EDMATC_ERRCLR

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_ERRCLR Register (Offset = 128h) [reset = 0h]: [0]

10.6.2.4.4 EDMATC_ERRDET Register (Offset = 12Ch) [reset = 0h]

The error details register (EDMATC_ERRDET) is shown in Figure 10-125 and described in Table 10-284

Table 10-283. EDMATC_ERRDET Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 012Ch
EDMACC_0_TC_1 (EDMATC_1)	0276 812Ch
EDMACC_1_TC_0 (EDMATC_2)	027B 012Ch
EDMACC_1_TC_1 (EDMATC_3)	027B 812Ch

Figure 10-125. EDMATC_ERRDET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						TCCHEN	TCINTEN
R-0h						R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED		TCC					
R-0h		R-0h					
7	6	5	4	3	2	1	0
RESERVED				STAT			
R-0h				R-0h			

LEGEND: R = Read Only; *-n* = value after reset

Table 10-284. EDMATC_ERRDET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17	TCCHEN	R	0h	Transfer completion chaining enable. Contains the TCCHEN value in the channel options parameter (OPT) programmed by the channel controller for the read or write transaction that resulted in an error.
16	TCINTEN	R	0h	Transfer completion interrupt enable. Contains the TCINTEN value in the channel options parameter (OPT) programmed by the channel controller for the read or write transaction that resulted in an error.
15-14	RESERVED	R	0h	Reserved
13-8	TCC	R	0h	Transfer complete code. Contains the TCC value in the channel options parameter (OPT) programmed by the channel controller for the read or write transaction that resulted in an error.
7-4	RESERVED	R	0h	Reserved
3-0	STAT	R	0h	Transaction status. Stores the nonzero status/error code that was detected on the read status or write status bus. If read status and write status are returned on the same cycle, then the EDMATC chooses nonzero version. If both are nonzero, then the write status is treated as higher priority. 0h = No error 1h - 7h = Read error 8h - Fh = Write error

Table 10-285. Register Call Summary for EDMATC_ERRDET

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller Operation • Error Generation: [0]

Table 10-285. Register Call Summary for EDMATC_ERRDET (continued)

EDMA Transfer Controller (EDMATC) Registers
• EDMATC_ERRSTAT Register (Offset = 120h) [reset = 0h]: [0] • EDMATC_ERRCLR Register (Offset = 128h) [reset = 0h]: [0][1][2] • EDMATC_ERRDET Register (Offset = 12Ch) [reset = 0h]: [0]

10.6.2.4.5 EDMATC_ERRCMD Register (Offset = 130h) [reset = 0h]

The error command register (EDMATC_ERRCMD) is shown in Figure 10-126 and described in Table 10-287.

Table 10-286. EDMATC_ERRCMD Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0130h
EDMACC_0_TC_1 (EDMATC_1)	0276 8130h
EDMACC_1_TC_0 (EDMATC_2)	027B 0130h
EDMACC_1_TC_1 (EDMATC_3)	027B 8130h

Figure 10-126. EDMATC_ERRCMD Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						SET	EVAL
R-0h						W-0h	W-0h

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 10-287. EDMATC_ERRCMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	SET	W	0h	Always write 0 to this bit; writes of 1 to this bit are not supported and attempts to do so may result in undefined behavior.
0	EVAL	W	0h	Error evaluate. 0h = No effect. 1h = EDMATC error line is pulsed if any of the error status register (EDMATC_ERRSTAT) bits are set.

Table 10-288. Register Call Summary for EDMATC_ERRCMD

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_ERRCMD Register (Offset = 130h) [reset = 0h]: [0]

10.6.2.5 EDMATC_RDRATE Register (Offset = 140h) [reset = 0h]

The EDMA transfer controller issues read commands at a rate controlled by the read rate register (EDMATC_RDRATE). The EDMATC_RDRATE defines the number of idle cycles that the read controller must wait before issuing subsequent commands. This applies both to commands within a transfer request packet (TRP) and for commands that are issued for different transfer requests (TRs). For instance, if EDMATC_RDRATE is set to 4 cycles between reads, there are 3 inactive cycles between reads.

EDMATC_RDRATE allows flexibility in transfer controller access requests to an endpoint. For an application, EDMATC_RDRATE can be manipulated to slow down the access rate, so that the endpoint may service requests from other masters during the inactive EDMATC cycles.

The EDMATC_RDRATE is shown in Figure 10-127 and described in Table 10-290.

NOTE: It is expected that the EDMATC_RDRATE value for a transfer controller is static, as it is decided based on the application requirement. It is not recommended to change this setting on the fly.

Table 10-289. EDMATC_RDRATE Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0140h
EDMACC_0_TC_1 (EDMATC_1)	0276 8140h
EDMACC_1_TC_0 (EDMATC_2)	027B 0140h
EDMACC_1_TC_1 (EDMATC_3)	027B 8140h

Figure 10-127. EDMATC_RDRATE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					RDRATE		
R-0h					R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 10-290. EDMATC_RDRATE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2-0	RDRATE	R/W	0h	Read rate. Controls the number of cycles between read commands. This is a global setting that applies to all TRs for this EDMATC. 0h = Reads issued as fast as possible. 1h = 4 cycles between reads. 2h = 8 cycles between reads. 3h = 16 cycles between reads. 4h = 32 cycles between reads. 5h - 7h = Reserved

Table 10-291. Register Call Summary for EDMATC_RDRATE

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller Operation • Performance Tuning: [0][1][2]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_RDRATE Register (Offset = 140h) [reset = 0h]: [0][1][2][3][4][5][6]

10.6.2.6 EDMATC Channel Registers

The EDMATC channel registers are split into three parts: the programming registers, the source active registers, and the destination FIFO register. This section describes the registers and their functions. The program register set is programmed by the channel controller, and is for internal use. The other two sets are read-only and provided to facilitate advanced debug capabilities.

The number of destination FIFO register sets depends on the destination FIFO depth. Each TC has a destination FIFO depth of 4, so there are four sets of destination FIFO registers for each of the transfer controllers.

10.6.2.6.1 EDMATC_SAOPT Register (Offset = 240h) [reset = 0h]

The source active options register (EDMATC_SAOPT) is shown in Figure 10-128 and described in Table 10-293.

Table 10-292. EDMATC_SAOPT Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0240h
EDMACC_0_TC_1 (EDMATC_1)	0276 8240h
EDMACC_1_TC_0 (EDMATC_2)	027B 0240h
EDMACC_1_TC_1 (EDMATC_3)	027B 8240h

Figure 10-128. EDMATC_SAOPT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED	TCCHEN	RESERVED	TCINTEN	RESERVED		TCC	
R-0h	R/W-0h	R-0h	R/W-0h	R-0h		R/W-0h	
15	14	13	12	11	10	9	8
TCC				RESERVED	FWID		
R/W-0h				R-0h	R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	PRI			RESERVED		DAM	SAM
R-0h	R/W-0h			R-0h		R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 10-293. EDMATC_SAOPT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22	TCCHEN	R/W	0h	Transfer completion chaining enable. 0h = Transfer complete chaining is disabled. 1h = Transfer complete chaining is enabled.
21	RESERVED	R	0h	Reserved
20	TCINTEN	R/W	0h	Transfer complete interrupt enable. 0h = Transfer complete interrupt is disabled. 1h = Transfer complete interrupt is enabled.
19-18	RESERVED	R	0h	Reserved
17-12	TCC	R/W	0h	Transfer complete code. This 6-bit code is used to set the relevant bit in CER or IPR of the EDMACC module.
11	RESERVED	R	0h	Reserved

Table 10-293. EDMATC_SAOPT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10-8	FWID	R/W	0h	FIFO width. Applies if either SAM or DAM is set to constant addressing mode. 0h = FIFO width is 8-bit. 1h = FIFO width is 16-bit. 2h = FIFO width is 32-bit. 3h = FIFO width is 64-bit. 4h = FIFO width is 128-bit. 5h = FIFO width is 256-bit. 6h - 7h = Reserved
7	RESERVED	R	0h	Reserved
6-4	PRI	R/W	0h	Transfer priority. Reflects the values programmed in the QUEPRI register in the EDMACC. 0h = Priority 0 - Highest priority 1h - 6h = Priority 1 to priority 6 7h = Priority 7 - Lowest priority
3-2	RESERVED	R	0h	Reserved
1	DAM	R/W	0h	Destination address mode within an array. 0h = Increment (INCR) mode. Destination addressing within an array increments. 1h = Constant addressing (CONST) mode. Destination addressing within an array wraps around upon reaching FIFO width.
0	SAM	R/W	0h	Source address mode within an array. 0h = Increment (INCR) mode. Source addressing within an array increments. 1h = Constant addressing (CONST) mode. Source addressing within an array wraps around upon reaching FIFO width.

Table 10-294. Register Call Summary for EDMATC_SAOPT

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_SAOPT Register (Offset = 240h) [reset = 0h]: [0] • EDMATC_SASRC Register (Offset = 244h) [reset = 0h]: [0]

10.6.2.6.2 EDMATC_SASRC Register (Offset = 244h) [reset = 0h]

The source active source address register (EDMATC_SASRC) is shown in Figure 10-129 and described in Table 10-296.

Table 10-295. EDMATC_SASRC Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0244h
EDMACC_0_TC_1 (EDMATC_1)	0276 8244h
EDMACC_1_TC_0 (EDMATC_2)	027B 0244h
EDMACC_1_TC_1 (EDMATC_3)	027B 8244h

Figure 10-129. EDMATC_SASRC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-296. EDMATC_SASRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SADDR	R	0h	Source address for program register set. EDMATC updates value according to source addressing mode (SAM bit in the source active options register, EDMATC_SAOPT). Value = 0-FFFF FFFFh

Table 10-297. Register Call Summary for EDMATC_SASRC

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_SASRC Register (Offset = 244h) [reset = 0h]: [0]

10.6.2.6.3 EDMATC_SACNT Register (Offset = 248h) [reset = 0h]

The source active count register (EDMATC_SACNT) is shown in Figure 10-130 and described in Table 10-299.

Table 10-298. EDMATC_SACNT Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0248h
EDMACC_0_TC_1 (EDMATC_1)	0276 8248h
EDMACC_1_TC_0 (EDMATC_2)	027B 0248h
EDMACC_1_TC_1 (EDMATC_3)	027B 8248h

Figure 10-130. EDMATC_SACNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCNT																ACNT															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-299. EDMATC_SACNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	BCNT	R	0h	B dimension count. Number of arrays to be transferred, where each array is ACNT in length. It is decremented after each read command appropriately. Represents the amount of data remaining to be read. It should be 0 when transfer request (TR) is complete. Value = 0-FFFFh
15-0	ACNT	R	0h	A dimension count. Number of bytes to be transferred in first dimension. It is decremented after each read command appropriately. Represents the amount of data remaining to be read. It should be 0 when transfer request (TR) is complete. Value = 0-FFFFh

Table 10-300. Register Call Summary for EDMATC_SACNT

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_SACNT Register (Offset = 248h) [reset = 0h]: [0]

10.6.2.6.4 EDMATC_SADST Register (Offset = 24Ch) [reset = 0h]

The source active destination address register (EDMATC_SADST) is shown in Figure 10-131 and described in Table 10-302.

Table 10-301. EDMATC_SADST Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 024Ch
EDMACC_0_TC_1 (EDMATC_1)	0276 824Ch
EDMACC_1_TC_0 (EDMATC_2)	027B 024Ch
EDMACC_1_TC_1 (EDMATC_3)	027B 824Ch

Figure 10-131. EDMATC_SADST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DADDR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-302. EDMATC_SADST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DADDR	R	0h	Always reads as 0.

Table 10-303. Register Call Summary for EDMATC_SADST

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_SADST Register (Offset = 24Ch) [reset = 0h]: [0]

10.6.2.6.5 EDMATC_SABIDX Register (Offset = 250h) [reset = 0h]

The source active set B-dimension index register (EDMATC_SABIDX) is shown in Figure 10-132 and described in Table 10-305.

Table 10-304. EDMATC_SABIDX Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0250h
EDMACC_0_TC_1 (EDMATC_1)	0276 8250h
EDMACC_1_TC_0 (EDMATC_2)	027B 0250h
EDMACC_1_TC_1 (EDMATC_3)	027B 8250h

Figure 10-132. EDMATC_SABIDX Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSTBIDX																SRCBIDX															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-305. EDMATC_SABIDX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DSTBIDX	R	0h	B-Index offset between destination arrays. Represents the offset in bytes between the starting address of each destination. Always reads as 0.
15-0	SRCBIDX	R	0h	B-Index offset between source arrays. Represents the offset in bytes between the starting address of each source array. Value = 0-FFFFh

Table 10-306. Register Call Summary for EDMATC_SABIDX

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_SABIDX Register (Offset = 250h) [reset = 0h]: [0]

10.6.2.6.6 EDMATC_SAMPPRXY Register (Offset = 254h) [reset = 0h]

The source active memory protection proxy register (EDMATC_SAMPPRXY) is shown in Figure 10-133 and described in Table 10-308.

Table 10-307. EDMATC_SAMPPRXY Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0254h
EDMACC_0_TC_1 (EDMATC_1)	0276 8254h
EDMACC_1_TC_0 (EDMATC_2)	027B 0254h
EDMACC_1_TC_1 (EDMATC_3)	027B 8254h

Figure 10-133. EDMATC_SAMPPRXY Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							PRIV
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED				PRIVID			
R-0h				R-0h			

LEGEND: R = Read Only; *-n* = value after reset

Table 10-308. EDMATC_SAMPPRXY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	PRIV	R	0h	Privilege level. The privilege level used by the host to set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMATC. The privilege ID is used while issuing read and write command to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIV of the host that set up the DMA transaction. 0h = User-level privilege 1h = Supervisor-level privilege
7-4	RESERVED	R	0h	Reserved
3-0	PRIVID	R	0h	Privilege ID. This contains the privilege ID of the host that set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMATC. This PRIVID value is used while issuing read and write commands to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIVID of the host that set up the DMA transaction.

Table 10-309. Register Call Summary for EDMATC_SAMPPRXY

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_SAMPPRXY Register (Offset = 254h) [reset = 0h]: [0]

10.6.2.6.7 EDMATC_SACNTRLD Register (Offset = 258h) [reset = 0h]

The source active count reload register (EDMATC_SACNTRLD) is shown in Figure 10-134 and described in Table 10-311.

Table 10-310. EDMATC_SACNTRLD Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0258h
EDMACC_0_TC_1 (EDMATC_1)	0276 8258h
EDMACC_1_TC_0 (EDMATC_2)	027B 0258h
EDMACC_1_TC_1 (EDMATC_3)	027B 8258h

Figure 10-134. EDMATC_SACNTRLD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ACNTRLD															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-311. EDMATC_SACNTRLD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	ACNTRLD	R	0h	A-count reload value. Represents the originally programmed value of ACNT. The reload value is used to reinitialize ACNT after each array is serviced. Value = 0-FFFFh

Table 10-312. Register Call Summary for EDMATC_SACNTRLD

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_SACNTRLD Register (Offset = 258h) [reset = 0h]: [0]

10.6.2.6.8 EDMATC_SASRCBREF Register (Offset = 25Ch) [reset = 0h]

The source active source address B-reference register (EDMATC_SASRCBREF) is shown in Figure 10-135 and described in Table 10-314.

Table 10-313. EDMATC_SASRCBREF Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 025Ch
EDMACC_0_TC_1 (EDMATC_1)	0276 825Ch
EDMACC_1_TC_0 (EDMATC_2)	027B 025Ch
EDMACC_1_TC_1 (EDMATC_3)	027B 825Ch

Figure 10-135. EDMATC_SASRCBREF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDRBREF																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-314. EDMATC_SASRCBREF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SADDRBREF	R	0h	Source address B-reference. Represents the starting address for the array currently being read. Value = 0-FFFF FFFFh

Table 10-315. Register Call Summary for EDMATC_SASRCBREF

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_SASRCBREF Register (Offset = 25Ch) [reset = 0h]: [0]

10.6.2.6.9 EDMATC_SADSTBREF Register (Offset = 260h) [reset = 0h]

The source active destination address B-reference register (EDMATC_SADSTBREF) is shown in Figure 10-136 and described in Table 10-317.

Table 10-316. EDMATC_SADSTBREF Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0260h
EDMACC_0_TC_1 (EDMATC_1)	0276 8260h
EDMACC_1_TC_0 (EDMATC_2)	027B 0260h
EDMACC_1_TC_1 (EDMATC_3)	027B 8260h

Figure 10-136. EDMATC_SADSTBREF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DADDRBREF																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-317. EDMATC_SADSTBREF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DADDRBREF	R	0h	Always reads as 0.

Table 10-318. Register Call Summary for EDMATC_SADSTBREF

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_SADSTBREF Register (Offset = 260h) [reset = 0h]: [0]

10.6.2.6.10 EDMATC_DFCNTRLD Register (Offset = 280h) [reset = 0h]

The destination FIFO count reload register (EDMATC_DFCNTRLD) is shown in Figure 10-137 and described in Table 10-320.

Table 10-319. EDMATC_DFCNTRLD Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0280h
EDMACC_0_TC_1 (EDMATC_1)	0276 8280h
EDMACC_1_TC_0 (EDMATC_2)	027B 0280h
EDMACC_1_TC_1 (EDMATC_3)	027B 8280h

Figure 10-137. EDMATC_DFCNTRLD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ACNTRLD															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-320. EDMATC_DFCNTRLD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	ACNTRLD	R	0h	A-count reload value. Represents the originally programmed value of ACNT. The reload value is used to reinitialize ACNT after each array is serviced. Value = 0-FFFFh

Table 10-321. Register Call Summary for EDMATC_DFCNTRLD

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFCNTRLD Register (Offset = 280h) [reset = 0h]: [0]

10.6.2.6.11 EDMATC_DFSRCBREF Register (Offset = 284h) [reset = 0h]

The destination FIFO source address B-reference register (EDMATC_DFSRCBREF) is shown in Figure 10-138 and described in Table 10-323.

Table 10-322. EDMATC_DFSRCBREF Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0284h
EDMACC_0_TC_1 (EDMATC_1)	0276 8284h
EDMACC_1_TC_0 (EDMATC_2)	027B 0284h
EDMACC_1_TC_1 (EDMATC_3)	027B 8284h

Figure 10-138. EDMATC_DFSRCBREF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDRBREF																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-323. EDMATC_DFSRCBREF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SADDRBREF	R	0h	Not Applicable. Always read as 0.

Table 10-324. Register Call Summary for EDMATC_DFSRCBREF

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFSRCBREF Register (Offset = 284h) [reset = 0h]: [0]

10.6.2.6.12 EDMATC_DFDSTBREF Register (Offset = 288h) [reset = 0h]

The destination FIFO destination address B-reference register (EDMATC_DFDSTBREF) is shown in Figure 10-139 and described in Table 10-326.

Table 10-325. EDMATC_DFDSTBREF Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0288h
EDMACC_0_TC_1 (EDMATC_1)	0276 8288h
EDMACC_1_TC_0 (EDMATC_2)	027B 0288h
EDMACC_1_TC_1 (EDMATC_3)	027B 8288h

Figure 10-139. EDMATC_DFDSTBREF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DADDRBREF																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-326. EDMATC_DFDSTBREF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DADDRBREF	R	0h	Destination address reference for the destination FIFO register set. Represents the starting address for the array currently being written. Value =0-FFFF FFFFh

Table 10-327. Register Call Summary for EDMATC_DFDSTBREF

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFDSTBREF Register (Offset = 288h) [reset = 0h]: [0]

10.6.2.6.13 EDMATC_DFOPT0 Register (Offset = 300h) [reset = 0h]

The destination FIFO options register (EDMATC_DFOPT0) is shown in Figure 10-140 and described in Table 10-329.

Table 10-328. EDMATC_DFOPT0 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0300h
EDMACC_0_TC_1 (EDMATC_1)	0276 8300h
EDMACC_1_TC_0 (EDMATC_2)	027B 0300h
EDMACC_1_TC_1 (EDMATC_3)	027B 8300h

Figure 10-140. EDMATC_DFOPT0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED	TCCHEN	RESERVED	TCINTEN	RESERVED		TCC	
R-0h	R/W-0h	R-0h	R/W-0h	R-0h		R/W-0h	
15	14	13	12	11	10	9	8
TCC				RESERVED	FWID		
R/W-0h				R-0h	R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	PRI			RESERVED		DAM	SAM
R-0h	R/W-0h			R-0h		R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 10-329. EDMATC_DFOPT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22	TCCHEN	R/W	0h	Transfer completion chaining enable. 0h = Transfer complete chaining is disabled. 1h = Transfer complete chaining is enabled.
21	RESERVED	R	0h	Reserved
20	TCINTEN	R/W	0h	Transfer complete interrupt enable. 0h = Transfer complete interrupt is disabled. 1h = Transfer complete interrupt is enabled.
19-18	RESERVED	R	0h	Reserved.
17-12	TCC	R/W	0h	Transfer complete code. This 6-bit code is used to set the relevant bit in CER or IPR of the EDMACC module.
11	RESERVED	R	0h	Reserved
10-8	FWID	R/W	0h	FIFO width. Applies if either SAM or DAM is set to constant addressing mode. 0h = FIFO width is 8-bit. 1h = FIFO width is 16-bit. 2h = FIFO width is 32-bit. 3h = FIFO width is 64-bit. 4h = FIFO width is 128-bit. 5h = FIFO width is 256-bit. 6h - 7h = Reserved
7	RESERVED	R	0h	Reserved

Table 10-329. EDMATC_DFOPT0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-4	PRI	R/W	0h	Transfer priority. 0h = Priority 0 - Highest priority 1h - 6h = Priority 1 to priority 6 7h = Priority 7 - Lowest priority
3-2	RESERVED	R	0h	Reserved
1	DAM	R/W	0h	Destination address mode within an array. 0h = Increment (INCR) mode. Destination addressing within an array increments. 1h = Constant addressing (CONST) mode. Destination addressing within an array wraps around upon reaching FIFO width.
0	SAM	R/W	0h	Source address mode within an array. 0h = Increment (INCR) mode. Source addressing within an array increments. 1h = Constant addressing (CONST) mode. Source addressing within an array wraps around upon reaching FIFO width.

Table 10-330. Register Call Summary for EDMATC_DFOPT0

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFOPT0 Register (Offset = 300h) [reset = 0h]: [0]

10.6.2.6.14 EDMATC_DFSRC0 Register (Offset = 304h) [reset = 0h]

The destination FIFO source address register (EDMATC_DFSRC0) is shown in Figure 10-141 and described in Table 10-332.

Table 10-331. EDMATC_DFSRC0 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0304h
EDMACC_0_TC_1 (EDMATC_1)	0276 8304h
EDMACC_1_TC_0 (EDMATC_2)	027B 0304h
EDMACC_1_TC_1 (EDMATC_3)	027B 8304h

Figure 10-141. EDMATC_DFSRC0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-332. EDMATC_DFSRC0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SADDR	R	0h	Always read as 0.

Table 10-333. Register Call Summary for EDMATC_DFSRC0

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFSRC0 Register (Offset = 304h) [reset = 0h]: [0]

10.6.2.6.15 EDMATC_DFCNT0 Register (Offset = 308h) [reset = 0h]

The destination FIFO count register (EDMATC_DFCNT0) is shown in Figure 10-142 and described in Table 10-335.

Table 10-334. EDMATC_DFCNT0 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0308h
EDMACC_0_TC_1 (EDMATC_1)	0276 8308h
EDMACC_1_TC_0 (EDMATC_2)	027B 0308h
EDMACC_1_TC_1 (EDMATC_3)	027B 8308h

Figure 10-142. EDMATC_DFCNT0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCNT																ACNT															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-335. EDMATC_DFCNT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	BCNT	R	0h	B-dimension count. Number of arrays to be transferred, where each array is ACNT in length. Count/count remaining for destination register set. Represents the amount of data remaining to be written. Value = 0-FFFFh
15-0	ACNT	R	0h	A-dimension count. Number of bytes to be transferred in first dimension count/count remaining for destination register set. Represents the amount of data remaining to be written. Value = 0-FFFFh

Table 10-336. Register Call Summary for EDMATC_DFCNT0

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFCNT0 Register (Offset = 308h) [reset = 0h]: [0]

10.6.2.6.16 EDMATC_DFDST0 Register (Offset = 30Ch) [reset = 0h]

The destination FIFO destination address register (EDMATC_DFDST0) is shown in Figure 10-143 and described in Table 10-338.

Table 10-337. EDMATC_DFDST0 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 030Ch
EDMACC_0_TC_1 (EDMATC_1)	0276 830Ch
EDMACC_1_TC_0 (EDMATC_2)	027B 030Ch
EDMACC_1_TC_1 (EDMATC_3)	027B 830Ch

Figure 10-143. EDMATC_DFDST0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DADDR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-338. EDMATC_DFDST0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DADDR	R	0h	Destination address for the destination FIFO register set. When a transfer request (TR) is complete, the final value should be the address of the last write command issued.

Table 10-339. Register Call Summary for EDMATC_DFDST0

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFDST0 Register (Offset = 30Ch) [reset = 0h]: [0]

10.6.2.6.17 EDMATC_DFBIDX0 Register (Offset = 310h) [reset = 0h]

The destination FIFO B-index register (EDMATC_DFBIDX0) is shown in Figure 10-144 and described in Table 10-341.

Table 10-340. EDMATC_DFBIDX0 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0310h
EDMACC_0_TC_1 (EDMATC_1)	0276 8310h
EDMACC_1_TC_0 (EDMATC_2)	027B 0310h
EDMACC_1_TC_1 (EDMATC_3)	027B 8310h

Figure 10-144. EDMATC_DFBIDX0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSTBIDX																SRCBIDX															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-341. EDMATC_DFBIDX0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DSTBIDX	R	0h	B-Index offset between destination arrays. Represents the offset in bytes between the starting address of each destination. Value = 0-FFFFh
15-0	SRCBIDX	R	0h	Always reads as 0.

Table 10-342. Register Call Summary for EDMATC_DFBIDX0

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFBIDX0 Register (Offset = 310h) [reset = 0h]: [0]

10.6.2.6.18 EDMATC_DFMPPRXY0 Register (Offset = 314h) [reset = 0h]

The destination FIFO memory protection proxy register (EDMATC_DFMPPRXY0) is shown in Figure 10-145 and described in Table 10-344.

Table 10-343. EDMATC_DFMPPRXY0 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0314h
EDMACC_0_TC_1 (EDMATC_1)	0276 8314h
EDMACC_1_TC_0 (EDMATC_2)	027B 0314h
EDMACC_1_TC_1 (EDMATC_3)	027B 8314h

Figure 10-145. EDMATC_DFMPPRXY0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							PRIV
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED				PRIVID			
R-0h				R-0h			

LEGEND: R = Read Only; *-n* = value after reset

Table 10-344. EDMATC_DFMPPRXY0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	PRIV	R	0h	Privilege level. This contains the Privilege level used by the EDMA programmer to set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMATC. The privilege ID is used while issuing read and write commands to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIV of the host that set up the DMA transaction. 0h = User-level privilege 1h = Supervisor-level privilege
7-4	RESERVED	R	0h	Reserved
3-0	PRIVID	R	0h	Privilege ID. This contains the Privilege ID of the EDMA programmer that set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMATC. This PRIVID value is used while issuing read and write commands to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIVID of the host that set up the DMA transaction.

Table 10-345. Register Call Summary for EDMATC_DFMPPRXY0

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFMPPRXY0 Register (Offset = 314h) [reset = 0h]: [0]

10.6.2.6.19 EDMATC_DFOPT1 Register (Offset = 340h) [reset = 0h]

The destination FIFO options register (EDMATC_DFOPT1) is shown in Figure 10-146 and described in Table 10-347.

Table 10-346. EDMATC_DFOPT1 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0340h
EDMACC_0_TC_1 (EDMATC_1)	0276 8340h
EDMACC_1_TC_0 (EDMATC_2)	027B 0340h
EDMACC_1_TC_1 (EDMATC_3)	027B 8340h

Figure 10-146. EDMATC_DFOPT1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED	TCCHEN	RESERVED	TCINTEN	RESERVED		TCC	
R-0h	R/W-0h	R-0h	R/W-0h	R-0h		R/W-0h	
15	14	13	12	11	10	9	8
TCC				RESERVED	FWID		
R/W-0h				R-0h	R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	PRI			RESERVED		DAM	SAM
R-0h	R/W-0h			R-0h		R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 10-347. EDMATC_DFOPT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22	TCCHEN	R/W	0h	Transfer completion chaining enable. 0h = Transfer complete chaining is disabled. 1h = Transfer complete chaining is enabled.
21	RESERVED	R	0h	Reserved
20	TCINTEN	R/W	0h	Transfer complete interrupt enable. 0h = Transfer complete interrupt is disabled. 1h = Transfer complete interrupt is enabled.
19-18	RESERVED	R	0h	Reserved.
17-12	TCC	R/W	0h	Transfer complete code. This 6-bit code is used to set the relevant bit in CER or IPR of the EDMACC module.
11	RESERVED	R	0h	Reserved
10-8	FWID	R/W	0h	FIFO width. Applies if either SAM or DAM is set to constant addressing mode. 0h = FIFO width is 8-bit. 1h = FIFO width is 16-bit. 2h = FIFO width is 32-bit. 3h = FIFO width is 64-bit. 4h = FIFO width is 128-bit. 5h = FIFO width is 256-bit. 6h - 7h = Reserved
7	RESERVED	R	0h	Reserved

Table 10-347. EDMATC_DFOPT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-4	PRI	R/W	0h	Transfer priority. 0h = Priority 0 - Highest priority 1h - 6h = Priority 1 to priority 6 7h = Priority 7 - Lowest priority
3-2	RESERVED	R	0h	Reserved
1	DAM	R/W	0h	Destination address mode within an array. 0h = Increment (INCR) mode. Destination addressing within an array increments. 1h = Constant addressing (CONST) mode. Destination addressing within an array wraps around upon reaching FIFO width.
0	SAM	R/W	0h	Source address mode within an array. 0h = Increment (INCR) mode. Source addressing within an array increments. 1h = Constant addressing (CONST) mode. Source addressing within an array wraps around upon reaching FIFO width.

Table 10-348. Register Call Summary for EDMATC_DFOPT1

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFOPT1 Register (Offset = 340h) [reset = 0h]: [0]

10.6.2.6.20 EDMATC_DFSRC1 Register (Offset = 344h) [reset = 0h]

The destination FIFO source address register (EDMATC_DFSRC1) is shown in Figure 10-147 and described in Table 10-350.

Table 10-349. EDMATC_DFSRC1 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0344h
EDMACC_0_TC_1 (EDMATC_1)	0276 8344h
EDMACC_1_TC_0 (EDMATC_2)	027B 0344h
EDMACC_1_TC_1 (EDMATC_3)	027B 8344h

Figure 10-147. EDMATC_DFSRC1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-350. EDMATC_DFSRC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SADDR	R	0h	Always read as 0.

Table 10-351. Register Call Summary for EDMATC_DFSRC1

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFSRC1 Register (Offset = 344h) [reset = 0h]: [0]

10.6.2.6.21 EDMATC_DFCNT1 Register (Offset = 348h) [reset = 0h]

The destination FIFO count register (EDMATC_DFCNT1) is shown in Figure 10-148 and described in Table 10-353.

Table 10-352. EDMATC_DFCNT1 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0348h
EDMACC_0_TC_1 (EDMATC_1)	0276 8348h
EDMACC_1_TC_0 (EDMATC_2)	027B 0348h
EDMACC_1_TC_1 (EDMATC_3)	027B 8348h

Figure 10-148. EDMATC_DFCNT1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCNT																ACNT															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-353. EDMATC_DFCNT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	BCNT	R	0h	B-dimension count. Number of arrays to be transferred, where each array is ACNT in length. Count/count remaining for destination register set. Represents the amount of data remaining to be written. Value = 0-FFFFh
15-0	ACNT	R	0h	A-dimension count. Number of bytes to be transferred in first dimension count/count remaining for destination register set. Represents the amount of data remaining to be written. Value = 0-FFFFh

Table 10-354. Register Call Summary for EDMATC_DFCNT1

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFCNT1 Register (Offset = 348h) [reset = 0h]: [0]

10.6.2.6.22 EDMATC_DFDST1 Register (Offset = 34Ch) [reset = 0h]

The destination FIFO destination address register (EDMATC_DFDST1) is shown in Figure 10-149 and described in Table 10-356.

Table 10-355. EDMATC_DFDST1 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 034Ch
EDMACC_0_TC_1 (EDMATC_1)	0276 834Ch
EDMACC_1_TC_0 (EDMATC_2)	027B 034Ch
EDMACC_1_TC_1 (EDMATC_3)	027B 834Ch

Figure 10-149. EDMATC_DFDST1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DADDR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-356. EDMATC_DFDST1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DADDR	R	0h	Destination address for the destination FIFO register set. When a transfer request (TR) is complete, the final value should be the address of the last write command issued.

Table 10-357. Register Call Summary for EDMATC_DFDST1

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFDST1 Register (Offset = 34Ch) [reset = 0h]: [0]

10.6.2.6.23 EDMATC_DFBIDX1 Register (Offset = 350h) [reset = 0h]

The destination FIFO B-index register (EDMATC_DFBIDX1) is shown in Figure 10-150 and described in Table 10-359.

Table 10-358. EDMATC_DFBIDX1 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0350h
EDMACC_0_TC_1 (EDMATC_1)	0276 8350h
EDMACC_1_TC_0 (EDMATC_2)	027B 0350h
EDMACC_1_TC_1 (EDMATC_3)	027B 8350h

Figure 10-150. EDMATC_DFBIDX1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSTBIDX																SRCBIDX															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-359. EDMATC_DFBIDX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DSTBIDX	R	0h	B-Index offset between destination arrays. Represents the offset in bytes between the starting address of each destination. Value = 0-FFFFh
15-0	SRCBIDX	R	0h	Always reads as 0.

Table 10-360. Register Call Summary for EDMATC_DFBIDX1

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFBIDX1 Register (Offset = 350h) [reset = 0h]: [0]

10.6.2.6.24 EDMATC_DFMPPRXY1 Register (Offset = 354h) [reset = 0h]

The destination FIFO memory protection proxy register (EDMATC_DFMPPRXY1) is shown in Figure 10-151 and described in Table 10-362.

Table 10-361. EDMATC_DFMPPRXY1 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0354h
EDMACC_0_TC_1 (EDMATC_1)	0276 8354h
EDMACC_1_TC_0 (EDMATC_2)	027B 0354h
EDMACC_1_TC_1 (EDMATC_3)	027B 8354h

Figure 10-151. EDMATC_DFMPPRXY1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							PRIV
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED				PRIVID			
R-0h				R-0h			

LEGEND: R = Read Only; *-n* = value after reset

Table 10-362. EDMATC_DFMPPRXY1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	PRIV	R	0h	Privilege level. This contains the Privilege level used by the EDMA programmer to set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMATC. The privilege ID is used while issuing read and write commands to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIV of the host that set up the DMA transaction. 0h = User-level privilege 1h = Supervisor-level privilege
7-4	RESERVED	R	0h	Reserved
3-0	PRIVID	R	0h	Privilege ID. This contains the Privilege ID of the EDMA programmer that set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMATC. This PRIVID value is used while issuing read and write commands to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIVID of the host that set up the DMA transaction.

Table 10-363. Register Call Summary for EDMATC_DFMPPRXY1

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFMPPRXY1 Register (Offset = 354h) [reset = 0h]: [0]

10.6.2.6.25 EDMATC_DFOPT2 Register (Offset = 380h) [reset = 0h]

The destination FIFO options register (EDMATC_DFOPT2) is shown in Figure 10-152 and described in Table 10-365.

Table 10-364. EDMATC_DFOPT2 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0380h
EDMACC_0_TC_1 (EDMATC_1)	0276 8380h
EDMACC_1_TC_0 (EDMATC_2)	027B 0380h
EDMACC_1_TC_1 (EDMATC_3)	027B 8380h

Figure 10-152. EDMATC_DFOPT2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED	TCCHEN	RESERVED	TCINTEN	RESERVED		TCC	
R-0h	R/W-0h	R-0h	R/W-0h	R-0h		R/W-0h	
15	14	13	12	11	10	9	8
TCC				RESERVED	FWID		
R/W-0h				R-0h	R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	PRI			RESERVED		DAM	SAM
R-0h	R/W-0h			R-0h		R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 10-365. EDMATC_DFOPT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22	TCCHEN	R/W	0h	Transfer completion chaining enable. 0h = Transfer complete chaining is disabled. 1h = Transfer complete chaining is enabled.
21	RESERVED	R	0h	Reserved
20	TCINTEN	R/W	0h	Transfer complete interrupt enable. 0h = Transfer complete interrupt is disabled. 1h = Transfer complete interrupt is enabled.
19-18	RESERVED	R	0h	Reserved.
17-12	TCC	R/W	0h	Transfer complete code. This 6-bit code is used to set the relevant bit in CER or IPR of the EDMACC module.
11	RESERVED	R	0h	Reserved
10-8	FWID	R/W	0h	FIFO width. Applies if either SAM or DAM is set to constant addressing mode. 0h = FIFO width is 8-bit. 1h = FIFO width is 16-bit. 2h = FIFO width is 32-bit. 3h = FIFO width is 64-bit. 4h = FIFO width is 128-bit. 5h = FIFO width is 256-bit. 6h - 7h = Reserved
7	RESERVED	R	0h	Reserved

Table 10-365. EDMATC_DFOPT2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-4	PRI	R/W	0h	Transfer priority. 0h = Priority 0 - Highest priority 1h - 6h = Priority 1 to priority 6 7h = Priority 7 - Lowest priority
3-2	RESERVED	R	0h	Reserved
1	DAM	R/W	0h	Destination address mode within an array. 0h = Increment (INCR) mode. Destination addressing within an array increments. 1h = Constant addressing (CONST) mode. Destination addressing within an array wraps around upon reaching FIFO width.
0	SAM	R/W	0h	Source address mode within an array. 0h = Increment (INCR) mode. Source addressing within an array increments. 1h = Constant addressing (CONST) mode. Source addressing within an array wraps around upon reaching FIFO width.

Table 10-366. Register Call Summary for EDMATC_DFOPT2

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFOPT2 Register (Offset = 380h) [reset = 0h]: [0]

10.6.2.6.26 EDMATC_DFSRC2 Register (Offset = 384h) [reset = 0h]

The destination FIFO source address register (EDMATC_DFSRC2) is shown in Figure 10-153 and described in Table 10-368.

Table 10-367. EDMATC_DFSRC2 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0384h
EDMACC_0_TC_1 (EDMATC_1)	0276 8384h
EDMACC_1_TC_0 (EDMATC_2)	027B 0384h
EDMACC_1_TC_1 (EDMATC_3)	027B 8384h

Figure 10-153. EDMATC_DFSRC2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-368. EDMATC_DFSRC2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SADDR	R	0h	Always read as 0.

Table 10-369. Register Call Summary for EDMATC_DFSRC2

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFSRC2 Register (Offset = 384h) [reset = 0h]: [0]

10.6.2.6.27 EDMATC_DFCNT2 Register (Offset = 388h) [reset = 0h]

The destination FIFO count register (EDMATC_DFCNT2) is shown in Figure 10-154 and described in Table 10-371.

Table 10-370. EDMATC_DFCNT2 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0388h
EDMACC_0_TC_1 (EDMATC_1)	0276 8388h
EDMACC_1_TC_0 (EDMATC_2)	027B 0388h
EDMACC_1_TC_1 (EDMATC_3)	027B 8388h

Figure 10-154. EDMATC_DFCNT2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCNT																ACNT															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-371. EDMATC_DFCNT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	BCNT	R	0h	B-dimension count. Number of arrays to be transferred, where each array is ACNT in length. Count/count remaining for destination register set. Represents the amount of data remaining to be written. Value = 0-FFFFh
15-0	ACNT	R	0h	A-dimension count. Number of bytes to be transferred in first dimension count/count remaining for destination register set. Represents the amount of data remaining to be written. Value = 0-FFFFh

Table 10-372. Register Call Summary for EDMATC_DFCNT2

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFCNT2 Register (Offset = 388h) [reset = 0h]: [0]

10.6.2.6.28 EDMATC_DFDST2 Register (Offset = 38Ch) [reset = 0h]

The destination FIFO destination address register (EDMATC_DFDST2) is shown in Figure 10-155 and described in Table 10-374.

Table 10-373. EDMATC_DFDST2 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 038Ch
EDMACC_0_TC_1 (EDMATC_1)	0276 838Ch
EDMACC_1_TC_0 (EDMATC_2)	027B 038Ch
EDMACC_1_TC_1 (EDMATC_3)	027B 838Ch

Figure 10-155. EDMATC_DFDST2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DADDR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-374. EDMATC_DFDST2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DADDR	R	0h	Destination address for the destination FIFO register set. When a transfer request (TR) is complete, the final value should be the address of the last write command issued.

Table 10-375. Register Call Summary for EDMATC_DFDST2

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFDST2 Register (Offset = 38Ch) [reset = 0h]: [0]

10.6.2.6.29 EDMATC_DFBIDX2 Register (Offset = 390h) [reset = 0h]

The destination FIFO B-index register (EDMATC_DFBIDX2) is shown in Figure 10-156 and described in Table 10-377.

Table 10-376. EDMATC_DFBIDX2 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0390h
EDMACC_0_TC_1 (EDMATC_1)	0276 8390h
EDMACC_1_TC_0 (EDMATC_2)	027B 0390h
EDMACC_1_TC_1 (EDMATC_3)	027B 8390h

Figure 10-156. EDMATC_DFBIDX2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSTBIDX																SRCBIDX															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-377. EDMATC_DFBIDX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DSTBIDX	R	0h	B-Index offset between destination arrays. Represents the offset in bytes between the starting address of each destination. Value = 0-FFFFh
15-0	SRCBIDX	R	0h	Always reads as 0.

Table 10-378. Register Call Summary for EDMATC_DFBIDX2

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFBIDX2 Register (Offset = 390h) [reset = 0h]: [0]

10.6.2.6.30 EDMATC_DFMPPRXY2 Register (Offset = 394h) [reset = 0h]

The destination FIFO memory protection proxy register (EDMATC_DFMPPRXY2) is shown in Figure 10-157 and described in Table 10-380.

Table 10-379. EDMATC_DFMPPRXY2 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 0394h
EDMACC_0_TC_1 (EDMATC_1)	0276 8394h
EDMACC_1_TC_0 (EDMATC_2)	027B 0394h
EDMACC_1_TC_1 (EDMATC_3)	027B 8394h

Figure 10-157. EDMATC_DFMPPRXY2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							PRIV
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED				PRIVID			
R-0h				R-0h			

LEGEND: R = Read Only; *-n* = value after reset

Table 10-380. EDMATC_DFMPPRXY2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	PRIV	R	0h	Privilege level. This contains the Privilege level used by the EDMA programmer to set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMATC. The privilege ID is used while issuing read and write commands to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIV of the host that set up the DMA transaction. 0h = User-level privilege 1h = Supervisor-level privilege
7-4	RESERVED	R	0h	Reserved
3-0	PRIVID	R	0h	Privilege ID. This contains the Privilege ID of the EDMA programmer that set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMATC. This PRIVID value is used while issuing read and write commands to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIVID of the host that set up the DMA transaction.

Table 10-381. Register Call Summary for EDMATC_DFMPPRXY2

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFMPPRXY2 Register (Offset = 394h) [reset = 0h]: [0]

10.6.2.6.31 EDMATC_DFOPT3 Register (Offset = 3C0h) [reset = 0h]

The destination FIFO options register (EDMATC_DFOPT3) is shown in Figure 10-158 and described in Table 10-383.

Table 10-382. EDMATC_DFOPT3 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 03C0h
EDMACC_0_TC_1 (EDMATC_1)	0276 83C0h
EDMACC_1_TC_0 (EDMATC_2)	027B 03C0h
EDMACC_1_TC_1 (EDMATC_3)	027B 83C0h

Figure 10-158. EDMATC_DFOPT3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED	TCCHEN	RESERVED	TCINTEN	RESERVED		TCC	
R-0h	R/W-0h	R-0h	R/W-0h	R-0h		R/W-0h	
15	14	13	12	11	10	9	8
TCC				RESERVED	FWID		
R/W-0h				R-0h	R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	PRI			RESERVED		DAM	SAM
R-0h	R/W-0h			R-0h		R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 10-383. EDMATC_DFOPT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22	TCCHEN	R/W	0h	Transfer completion chaining enable. 0h = Transfer complete chaining is disabled. 1h = Transfer complete chaining is enabled.
21	RESERVED	R	0h	Reserved
20	TCINTEN	R/W	0h	Transfer complete interrupt enable. 0h = Transfer complete interrupt is disabled. 1h = Transfer complete interrupt is enabled.
19-18	RESERVED	R	0h	Reserved.
17-12	TCC	R/W	0h	Transfer complete code. This 6-bit code is used to set the relevant bit in CER or IPR of the EDMACC module.
11	RESERVED	R	0h	Reserved
10-8	FWID	R/W	0h	FIFO width. Applies if either SAM or DAM is set to constant addressing mode. 0h = FIFO width is 8-bit. 1h = FIFO width is 16-bit. 2h = FIFO width is 32-bit. 3h = FIFO width is 64-bit. 4h = FIFO width is 128-bit. 5h = FIFO width is 256-bit. 6h - 7h = Reserved
7	RESERVED	R	0h	Reserved

Table 10-383. EDMATC_DFOPT3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-4	PRI	R/W	0h	Transfer priority. 0h = Priority 0 - Highest priority 1h - 6h = Priority 1 to priority 6 7h = Priority 7 - Lowest priority
3-2	RESERVED	R	0h	Reserved
1	DAM	R/W	0h	Destination address mode within an array. 0h = Increment (INCR) mode. Destination addressing within an array increments. 1h = Constant addressing (CONST) mode. Destination addressing within an array wraps around upon reaching FIFO width.
0	SAM	R/W	0h	Source address mode within an array. 0h = Increment (INCR) mode. Source addressing within an array increments. 1h = Constant addressing (CONST) mode. Source addressing within an array wraps around upon reaching FIFO width.

Table 10-384. Register Call Summary for EDMATC_DFOPT3

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFOPT3 Register (Offset = 3C0h) [reset = 0h]: [0]

10.6.2.6.32 EDMATC_DFSRC3 Register (Offset = 3C4h) [reset = 0h]

The destination FIFO source address register (EDMATC_DFSRC3) is shown in Figure 10-159 and described in Table 10-386.

Table 10-385. EDMATC_DFSRC3 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 03C4h
EDMACC_0_TC_1 (EDMATC_1)	0276 83C4h
EDMACC_1_TC_0 (EDMATC_2)	027B 03C4h
EDMACC_1_TC_1 (EDMATC_3)	027B 83C4h

Figure 10-159. EDMATC_DFSRC3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-386. EDMATC_DFSRC3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SADDR	R	0h	Always read as 0.

Table 10-387. Register Call Summary for EDMATC_DFSRC3

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFSRC3 Register (Offset = 3C4h) [reset = 0h]: [0]

10.6.2.6.33 EDMATC_DFCNT3 Register (Offset = 3C8h) [reset = 0h]

The destination FIFO count register (EDMATC_DFCNT3) is shown in Figure 10-160 and described in Table 10-389.

Table 10-388. EDMATC_DFCNT3 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 03C8h
EDMACC_0_TC_1 (EDMATC_1)	0276 83C8h
EDMACC_1_TC_0 (EDMATC_2)	027B 03C8h
EDMACC_1_TC_1 (EDMATC_3)	027B 83C8h

Figure 10-160. EDMATC_DFCNT3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCNT																ACNT															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-389. EDMATC_DFCNT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	BCNT	R	0h	B-dimension count. Number of arrays to be transferred, where each array is ACNT in length. Count/count remaining for destination register set. Represents the amount of data remaining to be written. Value = 0-FFFFh
15-0	ACNT	R	0h	A-dimension count. Number of bytes to be transferred in first dimension count/count remaining for destination register set. Represents the amount of data remaining to be written. Value = 0-FFFFh

Table 10-390. Register Call Summary for EDMATC_DFCNT3

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFCNT3 Register (Offset = 3C8h) [reset = 0h]: [0]

10.6.2.6.34 EDMATC_DFDST3 Register (Offset = 3CCh) [reset = 0h]

The destination FIFO destination address register (EDMATC_DFDST3) is shown in Figure 10-161 and described in Table 10-392.

Table 10-391. EDMATC_DFDST3 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 03CCh
EDMACC_0_TC_1 (EDMATC_1)	0276 83CCh
EDMACC_1_TC_0 (EDMATC_2)	027B 03CCh
EDMACC_1_TC_1 (EDMATC_3)	027B 83CCh

Figure 10-161. EDMATC_DFDST3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DADDR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-392. EDMATC_DFDST3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DADDR	R	0h	Destination address for the destination FIFO register set. When a transfer request (TR) is complete, the final value should be the address of the last write command issued.

Table 10-393. Register Call Summary for EDMATC_DFDST3

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFDST3 Register (Offset = 3CCh) [reset = 0h]: [0]

10.6.2.6.35 EDMATC_DFBIDX3 Register (Offset = 3D0h) [reset = 0h]

The destination FIFO B-index register (EDMATC_DFBIDX3) is shown in Figure 10-162 and described in Table 10-395.

Table 10-394. EDMATC_DFBIDX3 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 03D0h
EDMACC_0_TC_1 (EDMATC_1)	0276 83D0h
EDMACC_1_TC_0 (EDMATC_2)	027B 03D0h
EDMACC_1_TC_1 (EDMATC_3)	027B 83D0h

Figure 10-162. EDMATC_DFBIDX3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSTBIDX																SRCBIDX															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 10-395. EDMATC_DFBIDX3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DSTBIDX	R	0h	B-Index offset between destination arrays. Represents the offset in bytes between the starting address of each destination. Value = 0-FFFFh
15-0	SRCBIDX	R	0h	Always reads as 0.

Table 10-396. Register Call Summary for EDMATC_DFBIDX3

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFBIDX3 Register (Offset = 3D0h) [reset = 0h]: [0]

10.6.2.6.36 EDMATC_DFMPPRXY3 Register (Offset = 3D4h) [reset = 0h]

The destination FIFO memory protection proxy register (EDMATC_DFMPPRXY3) is shown in Figure 10-163 and described in Table 10-398.

Table 10-397. EDMATC_DFMPPRXY3 Instances

Instance	Physical Address
EDMACC_0_TC_0 (EDMATC_0)	0276 03D4h
EDMACC_0_TC_1 (EDMATC_1)	0276 83D4h
EDMACC_1_TC_0 (EDMATC_2)	027B 03D4h
EDMACC_1_TC_1 (EDMATC_3)	027B 83D4h

Figure 10-163. EDMATC_DFMPPRXY3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							PRIV
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED				PRIVID			
R-0h				R-0h			

LEGEND: R = Read Only; *-n* = value after reset

Table 10-398. EDMATC_DFMPPRXY3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	PRIV	R	0h	Privilege level. This contains the Privilege level used by the EDMA programmer to set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMATC. The privilege ID is used while issuing read and write commands to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIV of the host that set up the DMA transaction. 0h = User-level privilege 1h = Supervisor-level privilege
7-4	RESERVED	R	0h	Reserved
3-0	PRIVID	R	0h	Privilege ID. This contains the Privilege ID of the EDMA programmer that set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMATC. This PRIVID value is used while issuing read and write commands to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIVID of the host that set up the DMA transaction.

Table 10-399. Register Call Summary for EDMATC_DFMPPRXY3

EDMA Registers • EDMA Transfer Controller (EDMATC) Registers: [0][1]
EDMA Transfer Controller (EDMATC) Registers • EDMATC_DFMPPRXY3 Register (Offset = 3D4h) [reset = 0h]: [0]

Peripherals

Topic **Page**

11.1 Audio Sample Rate Converter (ASRC)

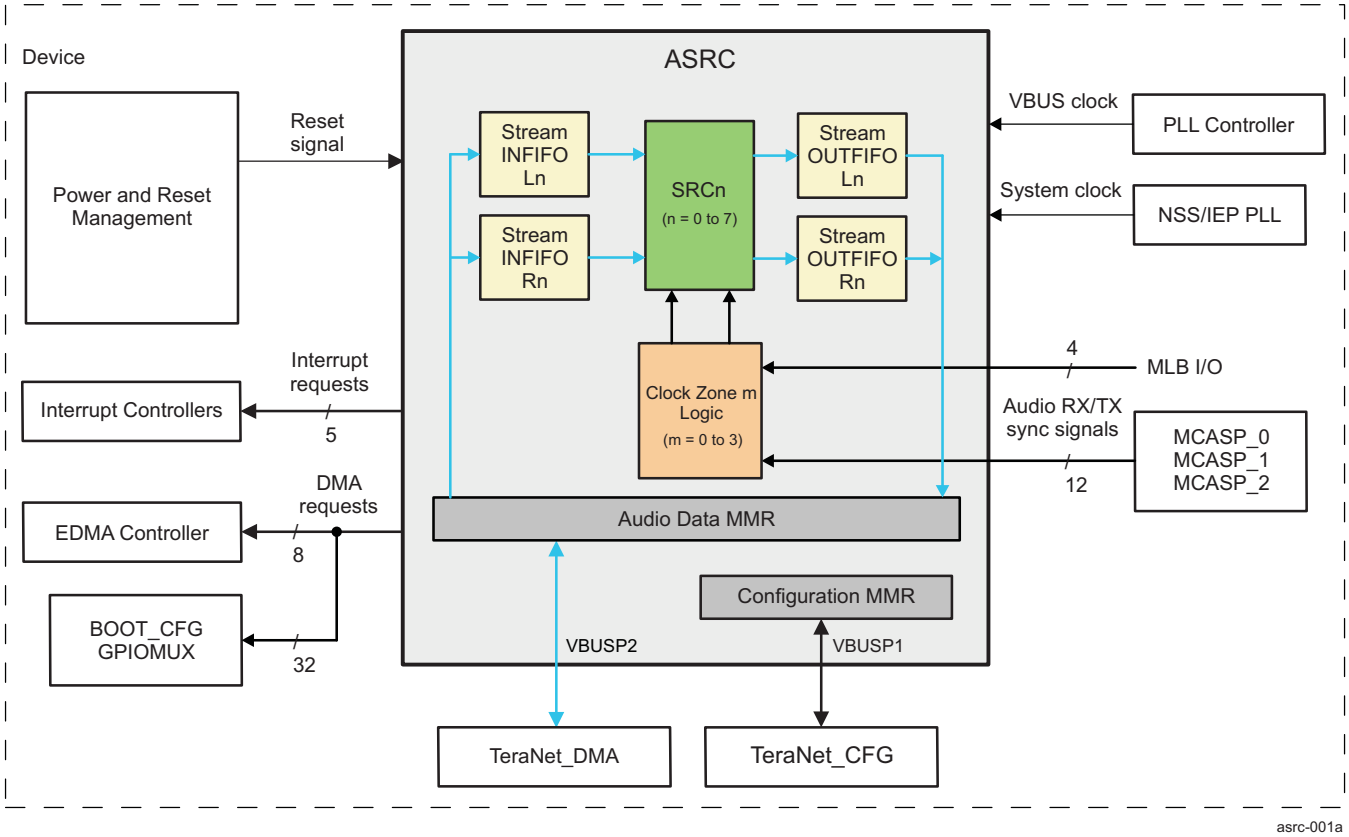
This section describes the asynchronous Audio Sample Rate Converter (ASRC) in the device.

11.1.1 ASRC Overview

The reception of many different audio sources and the transmission of these to different audio zones, may require different audio clocks. The asynchronous Audio Sample Rate Converter (ASRC) module takes samples from one clock zone and moves them to another, while maintaining a high signal to noise ratio to ensure that the output quality is sufficient to meet the requirements for various high-end algorithms.

Figure 11-1 is a simplified overivew diagram of the ASRC module implementation within the device.

Figure 11-1. ASRC Overview Diagram



The ASRC module supports the following main features:

- High performance Asynchronous Sample Rate Converter with 140dB Signal-to-Noise (SNR)
- Up to 8 input and output stereo streams (16 audio channels)
- 4 input and output clock zones
- Support for input and output audio sample rates from 8 kHz to 216 kHz
- Automatically sensing/detection of input sample frequencies
- Attenuation of sampling clock jitter
- 16-, 18-, 20-, 24-bit data input/output
- Input/output sampling ratios from 16:1 to 1:16
- 32-sample input and output FIFO for each channel with independently configurable thresholds
- Group mode, where multiple ASRC blocks use the same timing loop for input or output
- Linear phase FIR filter
- Controllable soft mute

- De-Emphasis supported for 48, 44.1 and 32 kHz sample rates
- Separate DMA events for input and output, for each channel and group
- Interrupts for input, output, and error for channels and groups of channels
- Channels belonging in an input/output clock zone may be configured and managed as a group or as individual streams

The ASRC module has the following main constrains:

- The ASRC output rate divided by input rate must be between 1/16 and 16
- Sample rate must be between 8 and 216 kHz

11.1.2 ASRC Integration

This section describes the ASRC integration in the device.

Figure 11-2. ASRC Integration Diagram

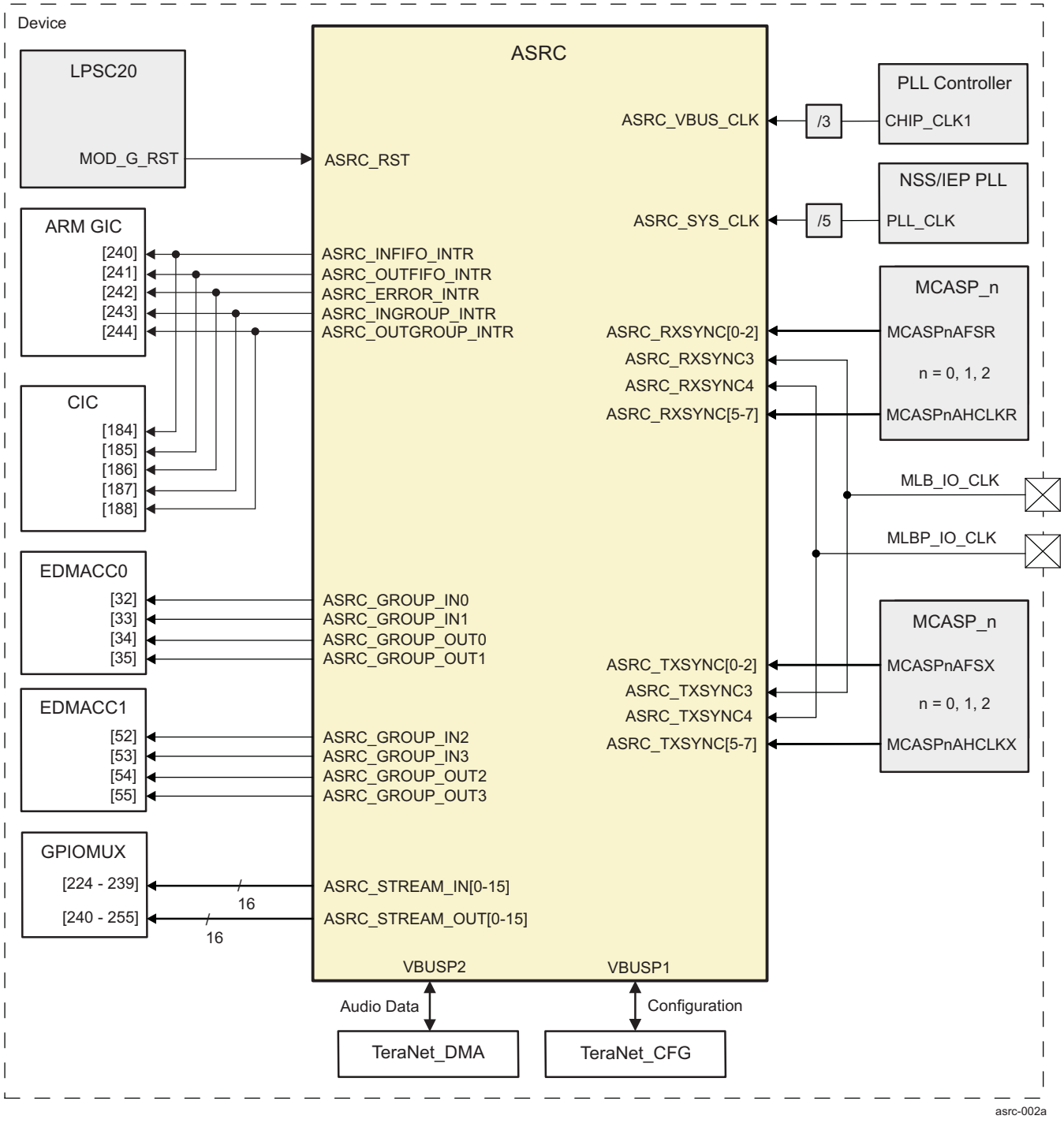


Table 11-3492 through Table 11-3494 summarize the integration of the module in the device.

Table 11-1. ASRC Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
ASRC	PD10	LPSC20	TeraNet_DMA (for data traffic) TeraNet_CFG (for configuration)

Table 11-2. ASRC Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
ASRC	ASRC_SYS_CLK	PLL_CLK / 5	NSS/IEP PLL	System clock (typically required 200 MHz)
	ASRC_VBUS_CLK	CHIP_CLK1 / 3	PLL Controller	VBUS clock
	ASRC_RXSYNC0	MCASP0AFSR	MCASP_0	Receive Frame Sync Channel0
	ASRC_RXSYNC1	MCASP1AFSR	MCASP_1	Receive Frame Sync Channel1
	ASRC_RXSYNC2	MCASP2AFSR	MCASP_2	Receive Frame Sync Channel2
	ASRC_RXSYNC3	MLB_IO_CLK	MLB_CLK pin	Receive Frame Sync Channel3
	ASRC_RXSYNC4	MLBP_IO_CLK	MLBP_CLK_P/N pins	Receive Frame Sync Channel4
	ASRC_RXSYNC5	MCASP0AHCLKR	MCASP_0	Receive Frame Sync Channel5
	ASRC_RXSYNC6	MCASP1AHCLKR	MCASP_1	Receive Frame Sync Channel6
	ASRC_RXSYNC7	MCASP2AHCLKR	MCASP_2	Receive Frame Sync Channel7
	ASRC_TXSYNC0	MCASP0AFSX	MCASP_0	Transmit Frame Sync Channel0
	ASRC_TXSYNC1	MCASP1AFSX	MCASP_1	Transmit Frame Sync Channel1
	ASRC_TXSYNC2	MCASP2AFSX	MCASP_2	Transmit Frame Sync Channel2
	ASRC_TXSYNC3	MLB_IO_CLK	MLB_CLK pin	Transmit Frame Sync Channel3
	ASRC_TXSYNC4	MLBP_IO_CLK	MLBP_CLK_P/N pins	Transmit Frame Sync Channel4
	ASRC_TXSYNC5	MCASP0AHCLKX	MCASP_0	Transmit Frame Sync Channel5
	ASRC_TXSYNC6	MCASP1AHCLKX	MCASP_1	Transmit Frame Sync Channel6
	ASRC_TXSYNC7	MCASP2AHCLKX	MCASP_2	Transmit Frame Sync Channel7
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
ASRC	ASRC_RST	MOD_G_RST	LPSC20	ASRC Reset signal

Table 11-3. ASRC Hardware Requests

Interrupt Requests				
Module Instance	**Event Name**	**Mapped To Input Event [Number]**		**Description**
		ARM GIC	**CIC**	
ASRC	ASRC_INFIFO_INTR	[240]	[184]	ASRC input FIFO interrupt
	ASRC_OUTFIFO_INTR	[241]	[185]	ASRC output FIFO interrupt
	ASRC_ERROR_INTR	[242]	[186]	ASRC error interrupt
	ASRC_INGROUP_INTR	[243]	[187]	ASRC Group input FIFO interrupt
	ASRC_OUTGROUP_INTR	[244]	[188]	ASRC Group output FIFO interrupt

DMA Requests					
Module Instance	**Event Name**	**Mapped To Input Event [Number]**			**Description**
		GPIOMUX	**EDMACC0**	**EDMACC1**	
ASRC	ASRC_STREAM_IN0	[224]	-	-	ASRC Stream0 input event

Table 11-3. ASRC Hardware Requests (continued)

ASRC_STREAM_IN1	[225]	-	-	ASRC Stream1 input event
ASRC_STREAM_IN2	[226]	-	-	ASRC Stream2 input event
ASRC_STREAM_IN3	[227]	-	-	ASRC Stream3 input event
ASRC_STREAM_IN4	[228]	-	-	ASRC Stream4 input event
ASRC_STREAM_IN5	[229]	-	-	ASRC Stream5 input event
ASRC_STREAM_IN6	[230]	-	-	ASRC Stream6 input event
ASRC_STREAM_IN7	[231]	-	-	ASRC Stream7 input event
ASRC_STREAM_IN8	[232]	-	-	ASRC Stream8 input event
ASRC_STREAM_IN9	[233]	-	-	ASRC Stream9 input event
ASRC_STREAM_IN10	[234]	-	-	ASRC Stream10 input event
ASRC_STREAM_IN11	[235]	-	-	ASRC Stream11 input event
ASRC_STREAM_IN12	[236]	-	-	ASRC Stream12 input event
ASRC_STREAM_IN13	[237]	-	-	ASRC Stream13 input event
ASRC_STREAM_IN14	[238]	-	-	ASRC Stream14 input event
ASRC_STREAM_IN15	[239]	-	-	ASRC Stream15 input event
ASRC_STREAM_OUT0	[240]	-	-	ASRC Stream0 output event
ASRC_STREAM_OUT1	[241]	-	-	ASRC Stream1 output event
ASRC_STREAM_OUT2	[242]	-	-	ASRC Stream2 output event
ASRC_STREAM_OUT3	[243]	-	-	ASRC Stream3 output event
ASRC_STREAM_OUT4	[244]	-	-	ASRC Stream4 output event
ASRC_STREAM_OUT5	[245]	-	-	ASRC Stream5 output event
ASRC_STREAM_OUT6	[246]	-	-	ASRC Stream6 output event
ASRC_STREAM_OUT7	[247]	-	-	ASRC Stream7 output event
ASRC_STREAM_OUT8	[248]	-	-	ASRC Stream8 output event
ASRC_STREAM_OUT9	[249]	-	-	ASRC Stream9 output event
ASRC_STREAM_OUT10	[250]	-	-	ASRC Stream10 output event
ASRC_STREAM_OUT11	[251]	-	-	ASRC Stream11 output event
ASRC_STREAM_OUT12	[252]	-	-	ASRC Stream12 output event
ASRC_STREAM_OUT13	[253]	-	-	ASRC Stream13 output event
ASRC_STREAM_OUT14	[254]	-	-	ASRC Stream14 output event
ASRC_STREAM_OUT15	[255]	-	-	ASRC Stream15 output event
ARSC_GROUP_IN0	-	[32]	-	ASRC Group0 input event
ARSC_GROUP_IN1	-	[33]	-	ASRC Group1 input event
ARSC_GROUP_IN2	-	-	[52]	ASRC Group2 input event
ARSC_GROUP_IN3	-	-	[53]	ASRC Group3 input event
ARSC_GROUP_OUT0	-	[34]	-	ASRC Group0 output event
ARSC_GROUP_OUT1	-	[35]	-	ASRC Group1 output event
ARSC_GROUP_OUT2	-	-	[54]	ASRC Group2 output event
ARSC_GROUP_OUT3	-	-	[55]	ASRC Group3 output event

NOTE: For more information on device power, reset, and clock management, see the corresponding sections within Chapter 5, *Device Configuration*.

For more information on device interconnects, see Section 3.1, *Interconnect*.

For more information on device interrupt controllers, see Chapter 9, *Interrupts*.

For more information on device DMA controllers, see Chapter 10, *EDMA Controller*.

For more information on device GPIOMUX, see Section 5.1.3.1.7, *Event Mux Control Registers*.

11.1.3 ASRC Functional Description

11.1.3.1 ASRC Block Diagram and Implementation

The Asynchronous Audio Sample Rate Converter (ASRC) allows the movement of up to 8 audio stereo streams (16 audio channels) from one audio zone to another audio zone via 8 two-channel sample rate converters.

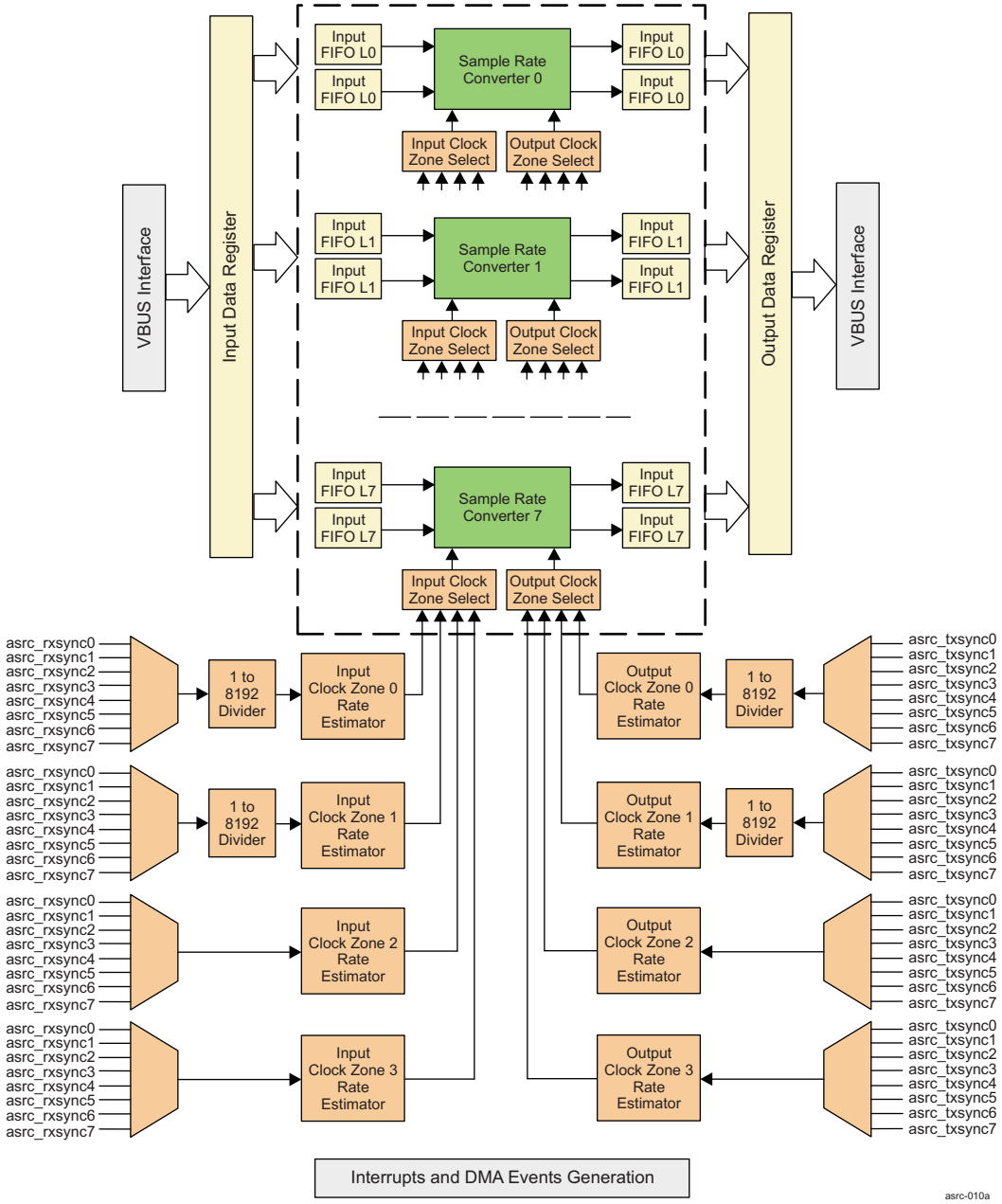
Each sample rate converter performs a digital audio sample rate conversion between input and output clock zone sample rates. The input and output clock zones can be individually selected for each sample rate converter in stream mode configuration. In group mode configuration, the sample rate converters that make up each group share the same input and output clock zone specifications. Each input and output channel is buffered by a 32-sample FIFO. The FIFO threshold values are programmed to a value between 1 and 16 samples.

The ASRC converts audio data between 4 input and 4 output clock zones. The clock zones provide a sample rate (Fs) input into clock rate estimators on the receive side and transmit side. Each ASRC receive and transmit clock zone input can come from one of 8 possible sources as shown in Figure 11-3. The input clocks and output clocks can have any frequency that is within the constraints specified in Section 11.1.1 and are completely asynchronous to each other.

The ASRC works by interpolating the input audio signals that are in the sequential domain of the source audio input clock. The input audio signal is interpolated up to 16x the input rate, then resampled at 16x the output rate, and then downsampled to the desired output clock frequency.

Figure 11-3 shows a simplified block diagram of ASRC module.

Figure 11-3. ASRC Block Diagram



The ASRC can handle multiple channels and switch between each channel as samples become available. The ASRC external interface is VBUS interface. One VBUS Interface is provided for configuration and status registers and another VBUS interface is dedicated for writing and reading audio samples. The ASRC also generate events when a configurable amount of output samples are ready. The ASRC implemets sample packer modules for processing the different streams as data becomes available.

The ASRC is a loosely coupled coprocessor to the main SoC DSP. Data is moved into the ASRC by device EDMA controller and an event is generated when another block of samples is completed for a given audio stream. In order to minimize the number of EDMA channels and events required, the ASRC also supports group events which are generated only when all of the selected Left and Right channels meet a pre-defined FIFO threshold. This permits a single EDMA transfer to perform the input and a single EDMA transfer to perform the output of all of the channels in the group.

As an alternative to EDMA moderated data input and output ASRC data transfers, the ASRC also supports input FIFO and output FIFO interrupts to permit a device processor to control the data transfers. The ASRC generates an error interrupt when a FIFO overflow and underflow error condition occurs.

The ASRC supports two modes of transferring data into the ASRC data inputs and from the ASRC data outputs:

- Group Mode, which enables all of the streams that are part of an input or output clock zone to be transferred using a single EDMA (each group is controlled by separate EDMA. Group mode is well suited for handing stereo or multichannel data from a single source.
- Stream Mode, which enables each stream to be transferred using a separate EDMA (each individual stream can be controlled by a separate EDMA). Stream mode is well suited for handing data from two or more isolated sources.

11.1.3.2 ASRC Data Rate and Traffic Behavior

The ASRC traffic and transaction behavior is a function of the input and output sample rates. The rate of the transactions is 4 bytes per sample. For example, if the input rate is 48 kHz, then the input data rate will be 192 KB/s. The traffic is bursty based upon the threshold size set in the ASRC_SRCFFCTRL_0 register.

The following must be considered:

- Output rate / Input rate ratio must be between 1/16 and 16
- Sample rate must be less than or equal to 216 kHz

11.1.3.3 ASRC Data Interface and Formats

NOTE: The only internal data format is of a 28-bit sample. Externally, the only format supported is a packed 24-bit samples into a 32-bit word, as described in this section.

Data is written to the ASCR and read from the ASRC using the VBUS interface. This interface has separate data register memory regions for Stream and Group input / output. Each region is used as based upon the mode (Group or Stream) that has been selected for each sample rate converter.

The ASRC data format is little endian, with up to 24-bit samples of data packed into 32-bit word.

ASRC supports 16-/18-/20-/24-bit audio data. ASRC understands audio data in the format dictated by Table 11-4, which holds true for both input and output channels.

Table 11-4. ASRC Data Packing at the SRC Interface

Sample Width	[31:24]	[23:8]	[7:6]	[5:4]	[3:0]
24-bit	8'h00	AudioData[23:0]			
20-bit	8'h00	AudioData[19:0]			4'h0
18-bit	8'h00	AudioData[17:0]		6'h00	
16-bit	8'h00	AudioData[15:0]	8'h00		

ASRC also supports automatic HW audio data alignment. Audio data alignment is active by default, but can be disabled via ASRC_SYSCONFIG[1] DATA_FORMAT_DISABLE register bit.

If auto data alignment feature is disabled, then data format for the audio samples at the VBUS interface can be seen in Table 11-5. For input audio samples packed into 32-bit VBUS packet, the most significant 8 bits [31:24] are always ignored by the ASRC internal engine. For output audio samples packed into 32-bit VBUS packet, the most significant 8 bits are always padded with 0's.

Table 11-5. ASRC Data Packing at the VBUS Interface When Auto Alignment is Disabled

Sample Width	[31:24]	[23:8]	[7:6]	[5:4]	[3:0]
24-bit	8'h00	AudioData[23:0]			
20-bit	8'h00	AudioData[19:0]			4'h0

Table 11-5. ASRC Data Packing at the VBUS Interface When Auto Alignment is Disabled (continued)

Sample Width	[31:24]	[23:8]	[7:6]	[5:4]	[3:0]
18-bit	8'h00	AudioData[17:0]		6'h00	
16-bit	8'h00	AudioData[15:0]	8'h00		

In case of automatic HW alignment is being used (this feature is selected by default), input audio samples packed into 32-bit VBUS packet must be LSB aligned. Input data format for the audio samples at the VBUS interface can be seen in Table 11-6.

Table 11-6. ASRC Input Data Packing at the VBUS Interface When Auto Alignment is Enabled

Sample Width	[31:24]	[23:20]	[19:18]	[17:16]	[15:0]
24-bit	xx	AudioData[23:0]			
20-bit	xx		AudioData[19:0]		
18-bit	xx			AudioData[17:0]	
16-bit	xx				AudioData[15:0]

In case of automatic HW alignment is being used, output audio samples will be packed into 32-bit VBUS packet. Audio data will be LSB aligned, little endian and sign extended 2's complement. Output data format for the audio samples at the VBUS interface can be seen in Table 11-7.

Table 11-7. ASRC Output Data Packing at the VBUS Interface When Auto Alignment is Enabled

Sample Width	[31:24]	[23:20]	[19:18]	[17:16]	[15:0]
24-bit	sign extend	AudioData[23:0]			
20-bit	sign extend		AudioData[19:0]		
18-bit	sign extend			AudioData[17:0]	
16-bit	sign extend				AudioData[15:0]

11.1.3.4 ASRC Clock Configuration

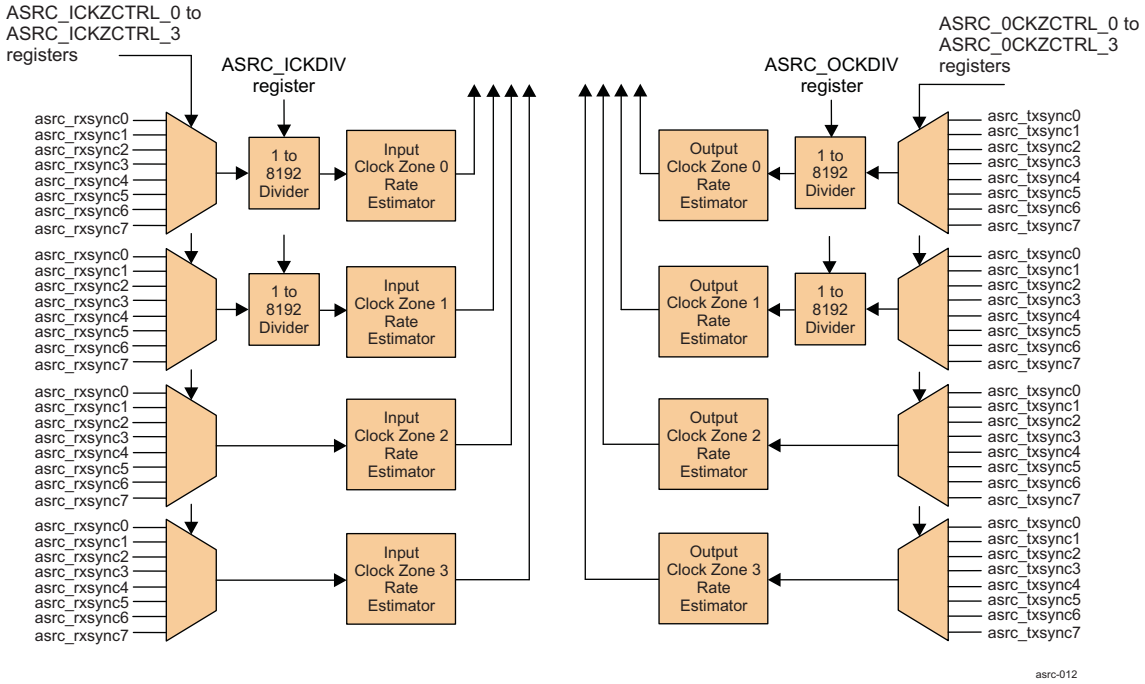
The ASRC_SYS_CLK is used for all filter processing in the module.

The ASRC_VBUS_CLK is used for VBUS interfaces.

There are four dividers present to divide the input audio sample rate clocks (ASRC_RXSYNC0-7 and ASRC_TXSYNC0-7). Two dividers are used for RX side and other two dividers are used for the TX side. The RX side dividers are working on the muxed ASRC_RXSYNC0-7 clocks. The TX side dividers are working on the muxed ASRC_TXSYNC0-7 clocks. The dividers are used for Clock Zone 0 and Clock Zone 1 only, on both RX and TX sides, and are controlled via ASRC_ICKDIV register (for RX side) and ASRC_OCKDIV register (for TX side).

The muxing of ASRC_RXSYNC0-7 clocks is controlled by [3-0] INPUT_CLOCK_ZONE_CLOCK_SOURCE_SELECT bit-field within ASRC_ICKZCTRL_0 to ASRC_ICKZCTRL_3 registers. The muxing of ASRC_TXSYNC0-7 clocks is controlled by [3-0] OUTPUT_CLOCK_ZONE_CLOCK_SOURCE_SELECT bit-field within ASRC_OCKZCNT_0 to ASRC_OCKZCNT_3 registers.

Figure 11-4. ASRC Input and Output Clock Zone Controls



11.1.3.5 ASRC Reset Configuration

The ASRC module is based on a single asynchronous reset that is synchronized to ASRC_SYS_CLK, ASRC_VBUS_CLK, and divider clocks (muxed version of TX and RX SYNC signals). The reset impacts configuration and processing information.

Additionally, each processing path enters a reset state when it is disabled, until the path is enabled with a new stream. This reset clears the data path and all memories associated with the path, when the new stream is started.

11.1.3.6 ASRC Power Management

Power management is accomplished using a standard clock stop protocol interface. When ASRC module is instructed to disable clocks for the internal clock domain, the internal clock network is shut down. This applies to the ASRC_SYS_CLK and ASRC_VBUS_CLK

Before a clock stop request is asserted at ASRC input, SW needs to confirm that ASRC is fully idle. HW will not check internal states to verify ASRC is idle. HW does check that no VBUSP transactions are ongoing before acknowledging the clock stop request.

11.1.3.7 ASRC Interrupts and Data Transfer Model

The ASRC can generate interrupt signals whenever interrupt condition is satisfied. These signals are exported on ASRC module boundary and can be used to trigger DMA transactions. There is a dedicated interrupt signal for audio samples write and for audio samples read, respectively.

All channels share common interrupt signals at the ASRC module boundary: infifo_intr and outfifo_intr, for write and read respectively. All groups also share common interrupt signals at the ASRC module boundary: ingrp_intr and outgrp_intr, for write and read respectively.

- In group mode, where more than one channels are part of one particular group, ingrp_intr and outgrp_intr need to be used, and infifo_intr and outfifo_intr need to be disabled accordingly.
- In stream mode, where a channel is working independently, infifo_intr and outfifo_intr need to be used.

For more details on interrupt descriptions, refer to Table 11-8. Software is aware of the configuration of each channel, whether it is a part of stream mode or group mode, and corresponding interrupts need to be disabled. If a channel is part of any group, then infifo_intr and outfifo_intr for that particular channel need to be disabled.

Table 11-8. ASRC Interrupts

Interrupt Event Name	Status Register	Set Register	Clear Register	Description
INFIFO_INTR	ASRC_IFIRQENSTS	ASRC_IFIRQENSET	ASRC_IFIRQENCLR	Interrupt indicating that one of the input FIFOs has enough space below the configured threshold.
OUTFIFO_INTR	ASRC_OFIRQENSTS	ASRC_OFIRQENSET	ASRC_OFIRQENCLR	Interrupt indicating that one of output FIFOs has enough data above the configured threshold.
INGROUP_INTR	ASRC_IGIRQENSTS	ASRC_IGIRQENSET	ASRC_IGIRQENCLR	Interrupt indicating that the one of the configured Group interrupts has all of the configured Input FIFOs below their respective threshold.
OUTGROUP_INTR	ASRC_OGIRQENSTS	ASRC_OGIRQENSET	ASRC_OGIRQENCLR	Interrupt indicating that one of the configured Group interrupts has all of the configured Output FIFOs above their respective threshold.
ERROR_INTR	ASRC_ERIRQENSTS	ASRC_ERIRQENSET	ASRC_ERIRQENCLR	Interrupt indicating one of the streams has received an error condition, FIFO overflow or underflow

11.1.3.7.1 Input FIFO Interrupts

Each channel has a dedicated input FIFO that is filled when an VBUS write (by a processor, EDMA, or DMA) occurs to that particular channel data register. The ASRC keeps tracks of the number of writes to the input FIFO and the number of reads that are sent into the processing path. The infifo interrupt will fire whenever the Input FIFO for the channel is below the threshold set in the ASRC_SRCFFCTRL_0[7:0] INFIFO_THRESHOLD register field. The interrupt will not be active until channel is enabled. Status for each FIFO can be checked in the ASRC_IFIRQENSTS status register. Once an INFIFO interrupt is fired, next interrupt will be generated only after INFIFO_THRESHOLD number of Audio RX Sync pulses has occured for that particular channel.

When the CPU is used to transfer data to ASRC, whenever an infifo_intr is triggered, SW needs to read the ASRC_IFIRQENSTS status register in order to determine which channels need to be serviced. Based on interrupt status register, software can trigger DMA to initiate transactions to channels. Data transfer will be in the same manner as described in Figure 11-5, *ASRC Stream Mode Audio Data Write for Channel N* with DMA triggered by software instead of infifo_evt event. After servicing the channel, SW needs to clear the interrupt for that particular channel.

11.1.3.7.2 Output FIFO Interrupts

Each channel has a dedicated output FIFO that is filled when the data path writes a sample to the output FIFO. The ASRC keeps tracks of the number of writes to the output FIFO and the number of reads that occur from the VBUS (by a processor, EDMA, or DMA). The outfifo interrupt will fire whenever the amount of available data matches the threshold set in the ASRC_SRCFFCTRL_0[23:16] OUTFIFO_THRESHOLD register field. Status for each FIFO can be checked in the ASRC_OFIRQENSTS status register. Once an OUTFIFO interrupt is fired, next interrupt will be generated only after OUTFIFO_THRESHOLD number of Audio RX Sync pulses has occured for that particular channel.

When the CPU is used to transfer data from ASRC, whenever an outfifo_intr is triggered, SW needs to read the ASRC_OFIRQENSTS status register and needs to determine which channel needs to be serviced. Based on interrupt status register software can trigger DMA to initiate transactions to channels. Data transfer will be in the same manner as described in Figure 11-6, *ASRC Stream Mode Audio Data Read for Channel N* with DMA triggered by software instead of outfifo_evt event. After servicing the channel SW needs to clear the interrupt for that particular channel.

11.1.3.7.3 Group Interrupts

The group interrupt is used to allow multiple stream DMA events to be tied together. That is, multiple channel inputs and outputs to be managed using a single DMA. A group is configured by writing to either the ASRC_IGRPSEL_0 to ASRC_IGRPSEL_3 registers for input groups, or the ASRC_OGRPSEL_0 to ASRC_OGRPSEL_3 registers for output groups. In the ASRC_IGRPSEL and ASRC_OGRPSEL registers there is a bit field for each channel. The group is determined by all the channels that have a 1 in them. When all of the selected streams in the input or output group are above the threshold, the corresponding group interrupt will fire. Once an ingrp_intr interrupt is fired, next interrupt will be generated only after INFIFO_THRESHOLD number of Audio RX Sync pulses has occured for all the channels of that group. Once an outgrp_intr interrupt is fired, next interrupt will be generated only after OUTFIFO_THRESHOLD number of Audio RX Sync pulses has occured for all the channels of that group.

When the CPU is used to transfer data to ASRC, whenever an ingrp_intr is triggered, SW needs to read the ASRC_IGIRQENSTS status register and needs to determine which group channels needs to be serviced. Based on interrupt status register, software can trigger DMA to initiate transactions to channels of that particular group. Data transfer will be in the same manner as described in Figure 11-7, *ASRC Group Mode Audio Data Write for Channels of Group N* with DMA triggered by software instead of ingrp_evt event. After servicing the channel, SW needs to clear the interrupt for that particular group.

When the CPU is used to transfer data from ASRC, whenever there is outgrp_intr is triggered, SW needs to read the ASRC_OGIRQENSTS status register and needs to determine which group channels needs to be serviced. Based on interrupt status register, software can trigger DMA to initiate transactions to channels of that particular group. Data transfer will be in the same manner as described in Figure 11-8, *ASRC Group Mode Audio Data Read from Channels of Group N* with DMA triggered by software instead of outgrp_evt event. After servicing the channel, SW needs to clear the interrupt for that particular group.

11.1.3.7.4 Error Interrupts

The error interrupt is used to signal a FIFO error condition. Below is a list of events that will cause the error interrupt to fire:

- Input FIFO overflow / underflow
- Output FIFO overflow / underflow

Overflow and underflow status for each SRC channel can be read through the ASRC_SRCFFCTRL_0 register. Clearing ERROR_INTR also clears the overflow/underflow status for that channel.

11.1.3.8 ASRC DMA Events and Data Transfer Model

The ASRC can be configured to generate individual EDMA events to initiate EDMA transfers for fillimg the input FIFOs in each operating mode, Group or Stream. Similarly, the ASRC is able to generate individual EDMA events to initiate EDMA transfers to empty the output FIFOs for each operating mode. Each event is triggered when the specific Group or Stream mode FIFO threshold condition has been met. The event generation is enabled on a group and stream basis. There are 4 group input events and 4 group output events. There are 16 stream input events and 8 stream output events (one for each input and output channel). The group and stream mode events are enabled when setting up the Stream or Group ASRC configuration

ASRC has individual event signals that generate pulse on ASRC external ports to trigger a DMA transaction. There are dedicated DMA events for audio samples write and read, respectively. Each channel/stream and each group has separate DMA event at the ASRC module boundary. The total number of events is determined by the number of channels and the number of groups. These events can be used to transfer the fixed block of data to ASRC.

- In group mode, where more than one channels are part of one particular group, ingrp_evt and

outgrp_evt for that group need to be used, and infifo_evt and outfifo_evt need to be ignored.

- In stream mode, where channel is working independently, infifo_evt and outfifo_evt for that channel need to be used.

Table 11-9 describes the DMA events, and provides mapping to ASRC boundary signals covered in Table 11-3494.

Table 11-9. ASRC DMA Events

DMA Event	Mapped to ASRC Boundary Signal	Number of Events	Description
infifo_evt[15:0]	ASRC_STREAM_IN[15:0]	16	The event will send out a pulse when the threshold for the input FIFO has fallen below the configured level. Once an INFIFO event is fired, next event will be generated only after ASRC_SRCFFCTRL_0[7:0] INFIFO_THRESHOLD number of Audio RX Sync pulses for that particular channel.
outfifo_evt[15:0]	ASRC_STREAM_OUT[15:0]	16	The event will send out a pulse when the threshold for the output FIFO has reached the configured level. Once an OUTFIFO event is fired, next event will be generated only after ASRC_SRCFFCTRL_0[23:16] OUTFIFO_THRESHOLD number of Audio RX Sync pulses for that particular channel.
ingrp_evt[3:0]	ASRC_GROUP_IN[3:0]	4	The event will send out a pulse when all input FIFOs in the configured group has fallen below their configured level. Once an INGROUP event is fired, next event will be generated only after INFIFO_THRESHOLD number of Audio RX Sync pulses for all the channels of that group.
outgrp_evt[3:0]	ASRC_GROUP_OUT[3:0]	4	The event will send out a pulse when all output FIFOs in the configured group has reached their configured level. Once an OUTGROUP event is fired, next event will be generated only after OUTFIFO_THRESHOLD number of Audio RX Sync pulses for all the channels of that group.

11.1.3.8.1 Stream Mode DMA Events

In stream mode, data transfer for channel number N is shown in Figure 11-5 and Figure 11-6.

Figure 11-5. ASRC Stream Mode Audio Data Write for Channel N

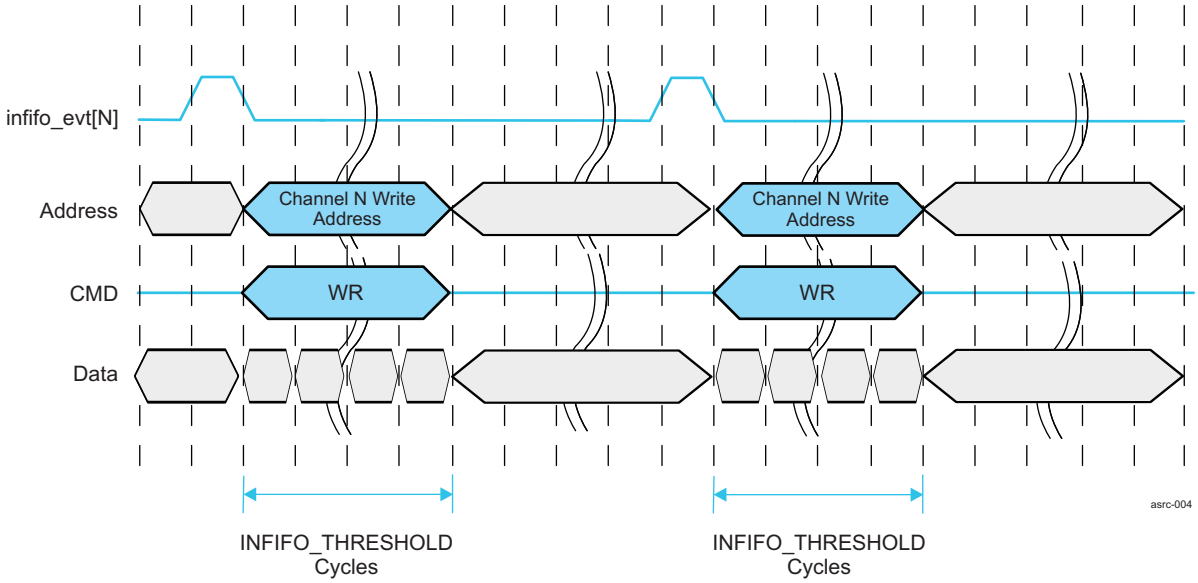
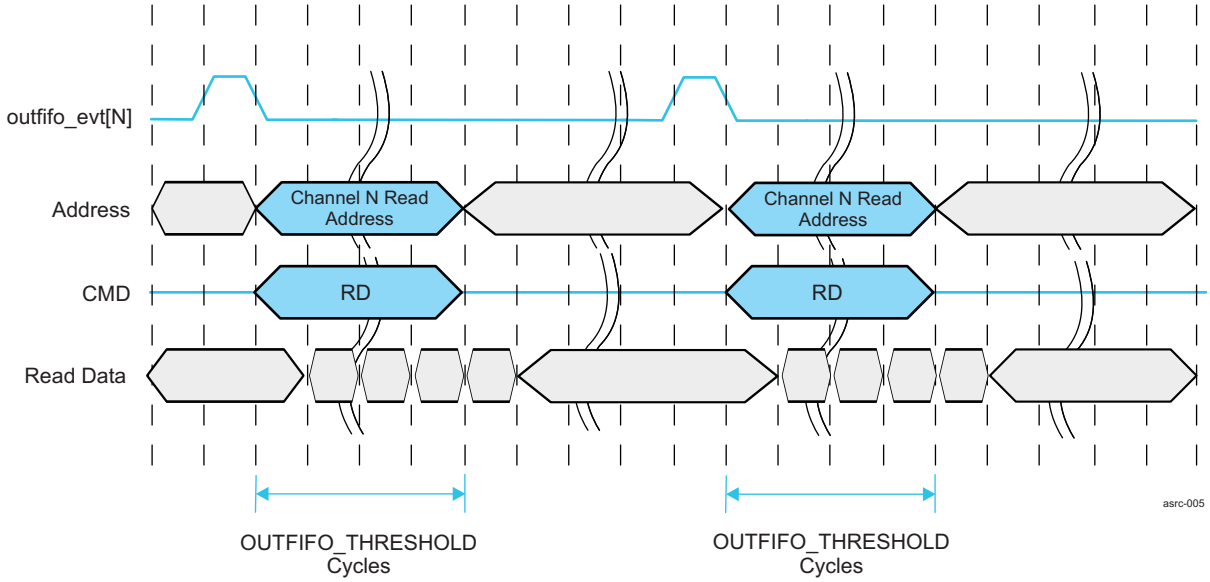


Figure 11-6. ASRC Stream Mode Audio Data Read for Channel N



It is not required for the device EDMA to service an event in one cycle, as it may be busy with other transfers also. If EDMA channel is shared between more than one channels, then it can queue the events and service whenever it has the resources available.

For each infifo_evt[N], that is infifo event for channel N, ASRC ensures that there is space available for at least INFIFO_THRESHOLD number of samples in the INFIFO of that channel. This INFIFO_THRESHOLD value can be set in ASRC_SRCFFCTRL_0[7:0] INFIFO_THRESHOLD register bitfield of that channel.

For each outfifo_evt[N], that is outfifo event for channel N, ASRC ensures that there is at least OUTFIFO_THRESHOLD number of samples available in OUTFIFO of that channel. This OUTFIFO_THRESHOLD value can be set in ASRC_SRCFFCTRL_0[23:16] OUTFIFO_THRESHOLD register bitfield of that channel.

11.1.3.8.2 Group Mode DMA Events

In group mode data transfer for all the channels of group number N is shown in Figure 11-7 and Figure 11-8.

Figure 11-7. ASRC Group Mode Audio Data Write for Channels of Group N

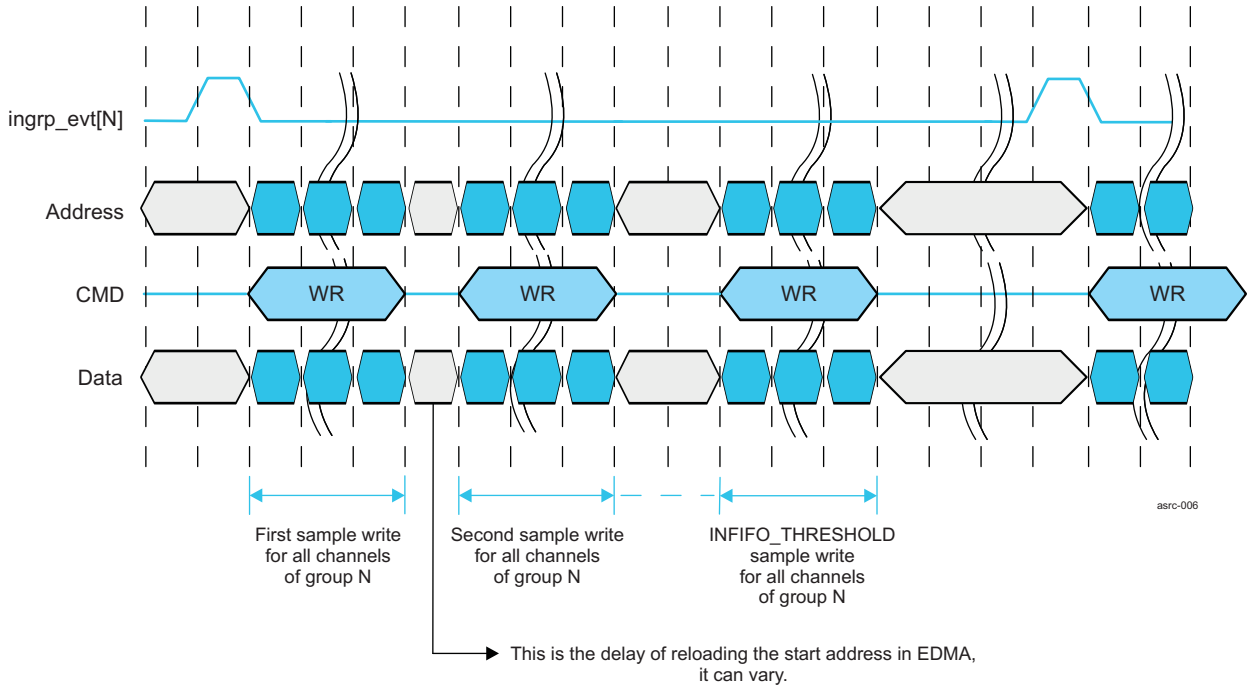
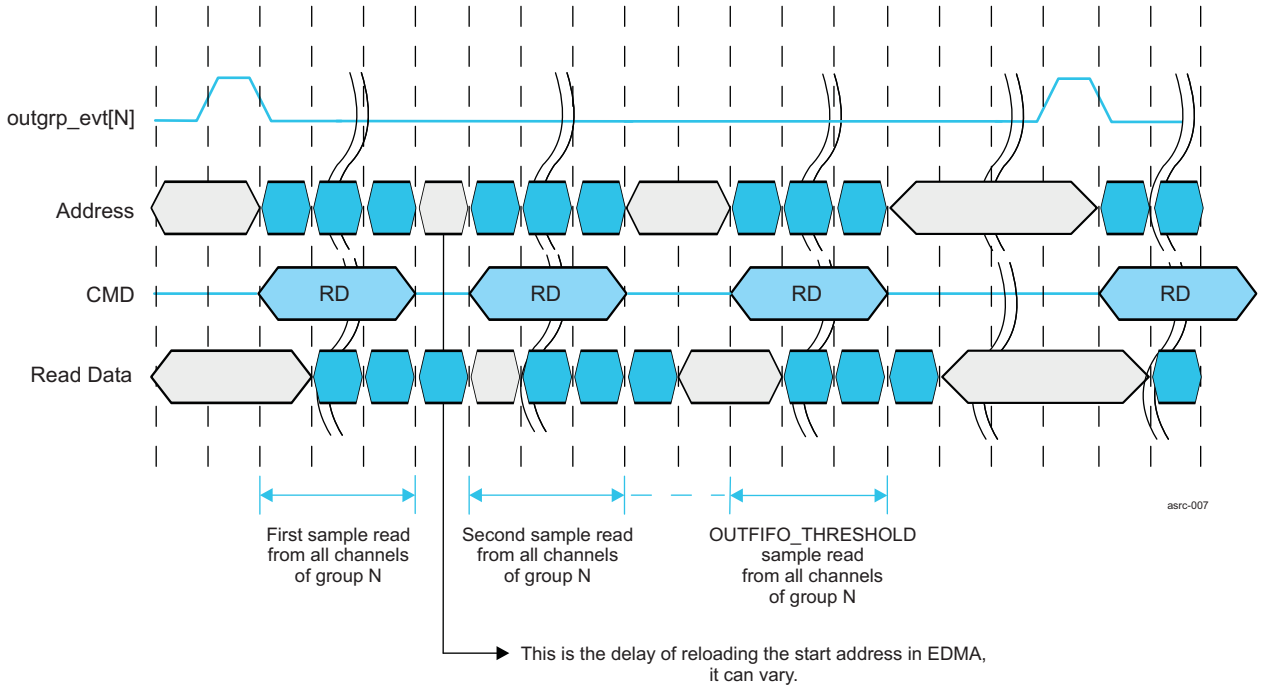


Figure 11-8. ASRC Group Mode Audio Data Read from Channels of Group N



It is not required for the device EDMA to service an event in one cycle as it may be busy with other transfers also. If EDMA channel is shared between more than one channels or groups, then it can queue the events and service whenever it has the resources available.

For each ingrp_evt[N], that is infifo event for group N, ASRC ensures that there is space available for at least INFIFO_THRESHOLD number of samples in the INFIFO of all the channels of that group. This INFIFO_THRESHOLD value can be set in ASRC_GFFCTRL_0[7:0] INFIFO_THRESHOLD register bitfield of that particular group. In this mode one sample is written in each of the channels of that group. The EDMA will update the adress to follow the channel number sequence of that group. This step is repeated INFIFO_THRESHOLD number of times to fill in INFIFO_THRESHOLD number of samples in each channel of that particular group.

For each outgrp_evt[N], that is outfifo event for group N, ASRC ensures that there is at least OUTFIFO_THRESHOLD number of samples available in OUTFIFO of all the channels of that group. This OUTFIFO_THRESHOLD value can be set in ASRC_GFFCTRL_0[23:16] OUTFIFO_THRESHOLD register bitfield of that particular group. In this mode one sample is read from all the channels of that group. The EDMA will update the address to follow the channel number sequence of that group. This step is repeated OUTFIFO_THRESHOLD number of times to drain OUTFIFO_THRESHOLD number of samples from each channel of that particular group.

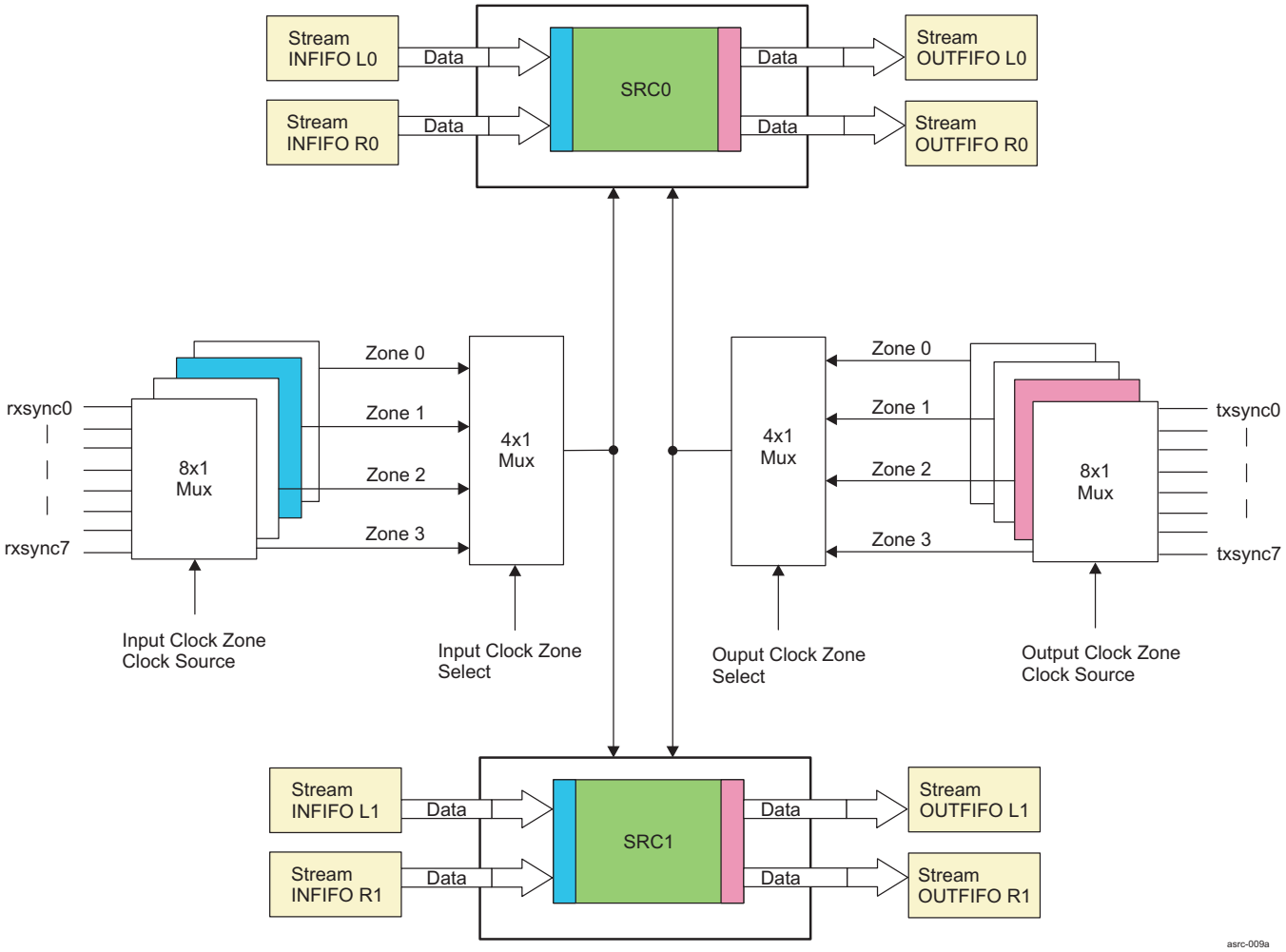
11.1.3.9 ASRC Group Mode Configuration

NOTE: Interleaving of audio data must be handled outside the ASRC module.

The ASRC supports 4 input and 4 output groups that can be configured. Each group can be composed of 1 to 16 channels. A stereo pair can only be assigned to one group. In group mode, a single INFIFO interrupt/event is triggered based upon status of Input FIFO status of all channels of the group. Similarly, a single OUTFIFO interrupt/event is triggered based upon status of Output FIFO status of all channels of the group. Each group can be controlled by separate EDMA.

Figure 11-9 shows and an example for group mode where all four input channels of SRC0 and SRC1 are part of input Clock Zone 1 and all four output channels are part of output Clock Zone 2.

Figure 11-9. ASRC Group Mode Example



The basic functionality of the ASRC follows the steps below for setting up a single group. For more details on programming model, refer to Section 11.1.3.9.1, *Group Mode Configuration Sequence*.

- Configure the Input and Output Clock Zone inputs to use one of the 8 possible input and output clock sources.
- Set the input and output channels.
- Configure INFIFO threshold and OUTFIFO threshold values.
- Configure Input and Output Word Length.
- Configure other conversion settings like de-emphasis, group delay, direct down conversion, output dither enable, and mute.
- Enable the conversion by enabling events or interrupts.

At this point, the ASRC automation starts. The ASRC clock zone input and output rate estimators will start estimation of the input and output sample rates. Once the estimation has settled, the ASRC will initiate input and output event or interrupt generation based upon the input and output FIFO status.

11.1.3.9.1 Group Mode Configuration Sequence

1. Configure Clock Zones:
   a. Program Input Clock Zone Clock Source and Loop Setup:
      - ASRC_ICKZCTRL_0[3-0] INPUT_CLOCK_ZONE_CLOCK_SOURCE_SELECT register bitfield
      - ASRC_ICKZCTRL_0[16-9] LOOP_SETUP register bitfield

b. Program Output Clock Zone Clock Source and Loop Setup
   - ASRC_OCKZCTRL_0[3-0] OUTPUT_CLOCK_ZONE_CLOCK_SOURCE_SELECT register bitfield
   - ASRC_OCKZCTRL_0[16-9] LOOP_SETUP register bitfield

2. Program Clock Zone 0 divider and Clock Zone 1 divider values (optional step, required if ASRC module dividers are being used):
   - ASRC_ICKDIV[13-0] CLOCK_ZONE0_DIVIDE
   - ASRC_ICKDIV[29-16] CLOCK_ZONE1_DIVIDE
   - ASRC_OCKDIV[13-0] CLOCK_ZONE0_DIVIDE
   - ASRC_OCKDIV[29-16] CLOCK_ZONE1_DIVIDE
3. Program Clock Zone 0 divider enable and Clock Zone 1 divider enable (optional step, required if ASRC module dividers are being used):
   - ASRC_ICKDIV[14] CLOCK_ZONE0_DIVIDE_EN
   - ASRC_ICKDIV[30] CLOCK_ZONE1_DIVIDE_EN
   - ASRC_OCKDIV[14] CLOCK_ZONE0_DIVIDE_EN
   - ASRC_OCKDIV[30] CLOCK_ZONE1_DIVIDE_EN
4. Program Input Group and Output Group Select Registers (select channels):
   - ASRC_IGRPSEL_0 to ASRC_IGRPSEL_3[...] CHANNEL_#_GROUP_ENABLE
   - ASRC_OGRPSEL_0 to ASRC_OGRPSEL_3[...] CHANNEL_#_GROUP_ENABLE
5. Program INFIFO threshold and OUTFIFO threshold values:
   - ASRC_GFFCTRL_0[7-0] INFIFO_THRESHOLD
   - ASRC_GFFCTRL_0[23-16] OUTFIFO_THRESHOLD
6. Program Output Word Length, Group Delay, De-emphasis Mode, Attenuation, Direct Down Sample, Mute, Dither Enable, Output Clock Zone Select and Input Clock Zone Select:
   - ASRC_GSRCCTRL_0[29-28] OUTPUT_WORD_LENGTH
   - ASRC_GSRCCTRL_0[27-26] GROUP_DELAY
   - ASRC_GSRCCTRL_0[25-24] DE_EMPHASIS_MODE
   - ASRC_GSRCCTRL_0[23-16] ATTENUATION
   - ASRC_GSRCCTRL_0[10] DIRECT_DOWN_SAMPLE
   - ASRC_GSRCCTRL_0[9] MUTE
   - ASRC_GSRCCTRL_0[8] DITHER_ENABLE
   - ASRC_GSRCCTRL_0[7-6] INPUT_WORD_LENGTH
   - ASRC_GSRCCTRL_0[5-3] OUTPUT_CLOCK_ZONE_SELECT
   - ASRC_GSRCCTRL_0[2-0] INPUT_CLOCK_ZONE_SELECT
7. Program Data Alignment Disable:
   - ASRC_SYSCONFIG[1] DATA_FORMAT_DISABLE
8. Program Channel Enable (channel enable will be effective, only if that particular channel is the part of a that particular group):
   - ASRC_GSRCCTRL_0[31-30] CHANNEL_ENABLE
9. Program DMA transfers:
   - Once the INGROUP_EVT event is triggered, a maximum block of ASRC_GFFCTRL_0[7-0] INFIFO_THRESHOLD audio samples can be written to the data register of each channel of that group. Once the OUTGROUP_EVT event is triggered, ASRC_GFFCTRL_0[23-16] OUTFIFO_THRESHOLD samples can be read from OUTPUT FIFO DATA registers (ASRC_GOFDATAL_0 and ASRC_GOFDATAR_0) of channel in the group. Events will not be triggered until Input Clock Recovery Loop and Output Clock Recovery Loop are settled. For this SW must check the status of bit [8] SETTLE in ASRC_ICKZCTRL_0 and ASRC_OCKZCTRL_0 registers, respectively.

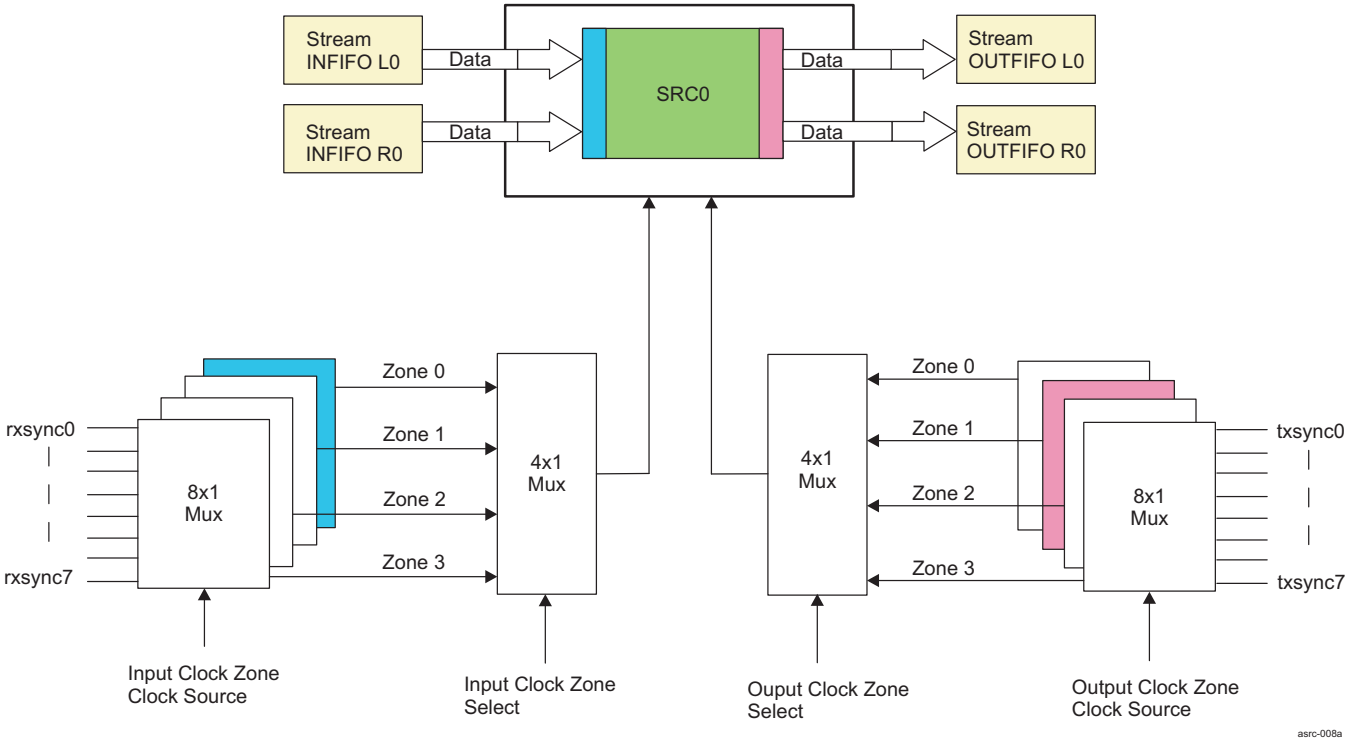
11.1.3.10 ASRC Stream Mode Configuration

NOTE: Interleaving of audio data must be handled outside the ASRC module.

In stream mode each stereo stream / channel can be controlled by a separate EDMA. Both channels of one SRC must be in same clock zone. User can process the audio samples by putting the sample in appropriate channel data register. In stream mode the individual channel FIFO interrupt / interrupts are not consolidated. In stream mode, a stereo pair is individually assigned to an input and output clock zone. A stereo pair is assigned even if only one channel is active. A stereo pair can only be part of one clock zone.

Figure 11-10 shows and an example for stream mode where the two input channels of SRC0 (which are part of input Clock Zone 0) are moved to output Clock Zone 1.

Figure 11-10. ASRC Stream Mode Example



The basic functionality of the ASRC in stream mode follows the steps below for setting up a single channel. For more details on programming model, refer to Section 11.1.3.10.1, *Stream Mode Configuration Sequence*.

- Configure the Input and Output Clock Zone inputs to use one of the 8 possible input and output clock sources.
- Set the input and output channels.
- Configure INFIFO threshold and OUTFIFO threshold values.
- Configure Input and Output Word Length.
- Configure other conversion settings like de-emphasis, group delay, direct down conversion, output dither enable, and mute.
- Enable the conversion by enabling events or interrupts.

At this point, the ASRC automation starts. The ASRC clock zone input and output rate estimators will start estimation of the input and output sample rates. Once the estimation has settled, the ASRC will initiate input and output event or interrupt generation based upon the input and output FIFO status.

11.1.3.10.1 Stream Mode Configuration Sequence

1. Configure Clock Zones:
   a. Program Input Clock Zone Clock Source and Loop Setup:
      - ASRC_ICKZCTRL_0[3-0] INPUT_CLOCK_ZONE_CLOCK_SOURCE_SELECT register bitfield
      - ASRC_ICKZCTRL_0[16-9] LOOP_SETUP register bitfield
   b. Program Output Clock Zone Clock Source and Loop Setup
      - ASRC_OCKZCTRL_0[3-0] OUTPUT_CLOCK_ZONE_CLOCK_SOURCE_SELECT register bitfield
      - ASRC_OCKZCTRL_0[16-9] LOOP_SETUP register bitfield
2. Program Clock Zone 0 divider and Clock Zone 1 divider values (optional step, required if ASRC module dividers are being used):
   - ASRC_ICKDIV[13-0] CLOCK_ZONE0_DIVIDE
   - ASRC_ICKDIV[29-16] CLOCK_ZONE1_DIVIDE
   - ASRC_OCKDIV[13-0] CLOCK_ZONE0_DIVIDE
   - ASRC_OCKDIV[29-16] CLOCK_ZONE1_DIVIDE
3. Program Clock Zone 0 divider enable and Clock Zone 1 divider enable (optional step, required if ASRC module dividers are being used):
   - ASRC_ICKDIV[14] CLOCK_ZONE0_DIVIDE_EN
   - ASRC_ICKDIV[30] CLOCK_ZONE1_DIVIDE_EN
   - ASRC_OCKDIV[14] CLOCK_ZONE0_DIVIDE_EN
   - ASRC_OCKDIV[30] CLOCK_ZONE1_DIVIDE_EN
4. Program INFIFO threshold and OUTFIFO threshold values:
   - ASRC_SRCFFCTRL_0[7-0] INFIFO_THRESHOLD
   - ASRC_SRCFFCTRL_0[23-16] OUTFIFO_THRESHOLD
5. Program Output Word Length, Group Delay, De-emphasis Mode, Attenuation, Direct Down Sample, Mute, Dither Enable, Output Clock Zone Select and Input Clock Zone Select:
   - ASRC_SRCCTRL_0[29-28] OUTPUT_WORD_LENGTH
   - ASRC_SRCCTRL_0[27-26] GROUP_DELAY
   - ASRC_SRCCTRL_0[25-24] DE_EMPHASIS_MODE
   - ASRC_SRCCTRL_0[23-16] ATTENUATION
   - ASRC_SRCCTRL_0[10] DIRECT_DOWN_SAMPLE
   - ASRC_SRCCTRL_0[9] MUTE
   - ASRC_SRCCTRL_0[8] DITHER_ENABLE
   - ASRC_SRCCTRL_0[7-6] INPUT_WORD_LENGTH
   - ASRC_SRCCTRL_0[5-3] OUTPUT_CLOCK_ZONE_SELECT
   - ASRC_SRCCTRL_0[2-0] INPUT_CLOCK_ZONE_SELECT
6. Program Data Alignment Disable (optional step, if desired):
   - ASRC_SYSCONFIG[1] DATA_FORMAT_DISABLE
7. Program Channel Enable:
   - ASRC_SRCCTRL_0[31-30] CHANNEL_ENABLE
8. Program DMA transfers:
   - Once the INFIFO_EVT event is triggered, a maximum block of ASRC_SRCFFCTRL_0[7-0] INFIFO_THRESHOLD audio samples can be written to the data register. If OUTFIFO_EVT event is triggered, then ASRC_SRCFFCTRL_0[23-16] OUTFIFO_THRESHOLD samples can be read from output data register. Events will not be triggered until Input Clock Recovery Loop and Output Clock Recovery Loop are settled. For this SW must check the status of [8] SETTLE bit in ASRC_ICKZCTRL_0 and ASRC_OCKZCTRL_0 registers, respectively.

The typical time that for an ASRC clock rate estimator to settle is proportionate to the clock rate that is input to the rate estimator. The typical setting time in ms is approximately 8850 / sample rate in kHz. A 192 kHz clock will typically settle in 46 ms.

11.1.3.11 ASRC Programming Ranges and Restrictions

The following programming ranges and restrictions are valid:

- The programmed values of [7-0] INFIFO_THRESHOLD and [23-16] OUTFIFO_THRESHOLD register bitfields should be within 1-32 range. This applies to ASRC_SRCFFCTRL_0 and ASRC_GFFCTRL_0 registers.
- Only the following settings for [16-9] LOOP_SETUP bit-field are allowed: 8'h00, 8'h40, 8'h80, 8'hC0. This applies to ASRC_ICKZCTRL_0 and ASRC_OCKZCTRL_0 registers.
- ASRC_IGRPSEL_0 to ASRC_IGRPSEL_3 and ASRC_OGRPSEL_0 to ASRC_OGRPSEL_3 registers should have same channels enabled for group number 0 to 3.

11.1.3.12 ASRC Registers Shadowing

ASRC module has no shadowing of registers. A stream must be disabled and a new stream enabled to change an active stream.

11.1.4 ASRC Registers

11.1.4.1 ASRC Configuration and Status Registers

Table 11-11 lists the memory-mapped registers for the ASRC configuration and status. All register address offsets not listed in Table 11-11 should be considered as reserved locations and corresponding register content should not be modified.

Table 11-10. ASRC Configuration and Status Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Table 11-11. ASRC Configuration and Status Registers

Offset	Acronym	Register Name	ASRC_0_CFG Physical Address	Section
0h	ASRC_PID	Peripheral Identification Register	021E 0000h	Section 11.1.4.1.1
10h	ASRC_SYSCONFIG	System Configuration Register	021E 0010h	Section 11.1.4.1.2
20h	ASRC_IRQEOI	End of Interrupt Register	021E 0020h	Section 11.1.4.1.3
24h	ASRC_IFIRQRAW	Input FIFO Interrupt Status Raw/Set Register	021E 0024h	Section 11.1.4.1.4
28h	ASRC_IFIRQENSTS	Input FIFO Interrupt Status Enabled/Clear Register	021E 0028h	Section 11.1.4.1.5
2Ch	ASRC_IFIRQENSET	Input FIFO Interrupt Enable/Set Register	021E 002Ch	Section 11.1.4.1.6
30h	ASRC_IFIRQENCLR	Input FIFO Interrupt Enable/Clear Register	021E 0030h	Section 11.1.4.1.7
34h	ASRC_OFIRQRAW	Output FIFO Interrupt Status Raw/Set Register	021E 0034h	Section 11.1.4.1.8
38h	ASRC_OFIRQENSTS	Output FIFO Interrupt Status Enabled/Clear Register	021E 0038h	Section 11.1.4.1.9
3Ch	ASRC_OFIRQENSET	Output FIFO Interrupt Enable/Set Register	021E 003Ch	Section 11.1.4.1.10

Table 11-11. ASRC Configuration and Status Registers (continued)

Offset	Acronym	Register Name	ASRC_0_CFG Physical Address	Section
40h	ASRC_OFIRQENCLR	Output FIFO Interrupt Enable/Clear Register	021E 0040h	Section 11.1.4.1.11
44h	ASRC_IGIRQRAW	Input Group Interrupt Status Raw/Set Register	021E 0044h	Section 11.1.4.1.12
48h	ASRC_IGIRQENSTS	Input Group Interrupt Status Enabled/Clear Register	021E 0048h	Section 11.1.4.1.13
4Ch	ASRC_IGIRQENSET	Input Group Interrupt Enable/Set Register	021E 004Ch	Section 11.1.4.1.14
50h	ASRC_IGIRQENCLR	Input Group Interrupt Enable/Clear Register 0	021E 0050h	Section 11.1.4.1.15
54h	ASRC_OGIRQRAW	Output Group Interrupt Status Raw/Set Register	021E 0054h	Section 11.1.4.1.16
58h	ASRC_OGIRQENSTS	Output Group Interrupt Status Enabled/Clear Register	021E 0058h	Section 11.1.4.1.17
5Ch	ASRC_OGIRQENSET	Output Group Interrupt Enable/Set Register	021E 005Ch	Section 11.1.4.1.18
60h	ASRC_OGIRQENCLR	Output Group Interrupt Enable/Clear Register 0	021E 0060h	Section 11.1.4.1.19
64h	ASRC_ERIRQRAW	Error Interrupt Status Raw/Set Register	021E 0064h	Section 11.1.4.1.20
68h	ASRC_ERIRQENSTS	Error Interrupt Status Enabled/Clear Register	021E 0068h	Section 11.1.4.1.21
6Ch	ASRC_ERIRQENSET	Error Interrupt Enable/Set Register	021E 006Ch	Section 11.1.4.1.22
70h	ASRC_ERIRQENCLR	Error Interrupt Enable/Clear Register	021E 0070h	Section 11.1.4.1.23
74h to 80h	ASRC_IGRPSEL_0 to ASRC_IGRPSEL_3	Input Group 0 to Input Group 3 Select Register	021E 0074h to 021E 0080h	Section 11.1.4.1.24
94h to A0h	ASRC_OGRPSEL_0 to ASRC_OGRPSEL_3	Output Group 0 to Output Group 3 Select Register	021E 0094h to 021E 00A0h	Section 11.1.4.1.25
400h	ASRC_ICKDIV	Input Clock Zone Divider Register	021E 0400h	Section 11.1.4.1.26
404h	ASRC_OCKDIV	Output Clock Zone Divider Register	021E 0404h	Section 11.1.4.1.27
100h	ASRC_SRCFFCTRL_0	SRC FIFO Control 0 Register	021E 0100h	Section 11.1.4.1.28
104h	ASRC_SRCCTRL_0	SRC Control 0 Register	021E 0104h	Section 11.1.4.1.29
108h	ASRC_SRCSTS_0	SRC Status 0 Register	021E 0108h	Section 11.1.4.1.30
110h	ASRC_SRCFFCTRL_1	SRC FIFO Control 1 Register	021E 0110h	Section 11.1.4.1.28
114h	ASRC_SRCCTRL_1	SRC Control 1 Register	021E 0114h	Section 11.1.4.1.29
118h	ASRC_SRCSTS_1	SRC Status 1 Register	021E 0118h	Section 11.1.4.1.30
120h	ASRC_SRCFFCTRL_2	SRC FIFO Control 2 Register	021E 0120h	Section 11.1.4.1.28
124h	ASRC_SRCCTRL_2	SRC Control 2 Register	021E 0124h	Section 11.1.4.1.29
128h	ASRC_SRCSTS_2	SRC Status 2 Register	021E 0128h	Section 11.1.4.1.30
130h	ASRC_SRCFFCTRL_3	SRC FIFO Control 3 Register	021E 0130h	Section 11.1.4.1.28

Table 11-11. ASRC Configuration and Status Registers (continued)

Offset	Acronym	Register Name	ASRC_0_CFG Physical Address	Section
134h	ASRC_SRCCTRL_3	SRC Control 3 Register	021E 0134h	Section 11.1.4.1.29
138h	ASRC_SRCSTS_3	SRC Status 3 Register	021E 0138h	Section 11.1.4.1.30
140h	ASRC_SRCFFCTRL_4	SRC FIFO Control 4 Register	021E 0140h	Section 11.1.4.1.28
144h	ASRC_SRCCTRL_4	SRC Control 4 Register	021E 0144h	Section 11.1.4.1.29
148h	ASRC_SRCSTS_4	SRC Status 4 Register	021E 0148h	Section 11.1.4.1.30
150h	ASRC_SRCFFCTRL_5	SRC FIFO Control 5 Register	021E 0150h	Section 11.1.4.1.28
154h	ASRC_SRCCTRL_5	SRC Control 5 Register	021E 0154h	Section 11.1.4.1.29
158h	ASRC_SRCSTS_5	SRC Status 5 Register	021E 0158h	Section 11.1.4.1.30
160h	ASRC_SRCFFCTRL_6	SRC FIFO Control 6 Register	021E 0160h	Section 11.1.4.1.28
164h	ASRC_SRCCTRL_6	SRC Control 6 Register	021E 0164h	Section 11.1.4.1.29
168h	ASRC_SRCSTS_6	SRC Status 6 Register	021E 0168h	Section 11.1.4.1.30
170h	ASRC_SRCFFCTRL_7	SRC FIFO Control 7 Register	021E 0170h	Section 11.1.4.1.28
174h	ASRC_SRCCTRL_7	SRC Control 7 Register	021E 0174h	Section 11.1.4.1.29
178h	ASRC_SRCSTS_7	SRC Status 7 Register	021E 0178h	Section 11.1.4.1.30
180h	ASRC_GFFCTRL_0	Group FIFO Control 0 Register	021E 0180h	Section 11.1.4.1.31
184h	ASRC_GSRCCTRL_0	Group SRC Control 0 Register	021E 0184h	Section 11.1.4.1.32
188h	ASRC_GFFCTRL_1	Group FIFO Control 1 Register	021E 0188h	Section 11.1.4.1.31
18Ch	ASRC_GSRCCTRL_1	Group SRC Control 1 Register	021E 018Ch	Section 11.1.4.1.32
190h	ASRC_GFFCTRL_2	Group FIFO Control 2 Register	021E 0190h	Section 11.1.4.1.31
194h	ASRC_GSRCCTRL_2	Group SRC Control 2 Register	021E 0194h	Section 11.1.4.1.32
198h	ASRC_GFFCTRL_3	Group FIFO Control 3 Register	021E 0198h	Section 11.1.4.1.31
19Ch	ASRC_GSRCCTRL_3	Group SRC Control 3 Register	021E 019Ch	Section 11.1.4.1.32
200h	ASRC_ICKGENSTL_0	Reserved	021E 0200h	Section 11.1.4.1.33
204h	ASRC_ICKGENSTH_0	Reserved	021E 0204h	Section 11.1.4.1.34
208h	ASRC_ICKGENRTL_0	Reserved	021E 0208h	Section 11.1.4.1.35
20Ch	ASRC_ICKGENRTH_0	Reserved	021E 020Ch	Section 11.1.4.1.36
210h	ASRC_ICKLPRTL_0	Reserved	021E 0210h	Section 11.1.4.1.37

Table 11-11. ASRC Configuration and Status Registers (continued)

Offset	Acronym	Register Name	ASRC_0_CFG Physical Address	Section
214h	ASRC_ICKPRTH_0	Reserved	021E 0214h	Section 11.1.4.1.38
218h	ASRC_ICKZCNT_0	Input Clockzone Count 0 Register	021E 0218h	Section 11.1.4.1.39
21Ch	ASRC_ICKZCTRL_0	Input Clockzone Control 0 Register	021E 021Ch	Section 11.1.4.1.40
220h	ASRC_OCKGENSTL_0	Reserved	021E 0220h	Section 11.1.4.1.41
224h	ASRC_OCKGENSTH_0	Reserved	021E 0224h	Section 11.1.4.1.42
228h	ASRC_OCKGENRTL_0	Reserved	021E 0228h	Section 11.1.4.1.43
22Ch	ASRC_OCKGENRTH_0	Reserved	021E 022Ch	Section 11.1.4.1.44
230h	ASRC_OCKLPRTL_0	Reserved	021E 0230h	Section 11.1.4.1.45
234h	ASRC_OCKLPRTH_0	Reserved	021E 0234h	Section 11.1.4.1.46
238h	ASRC_OCKLPSTL_0	Reserved	021E 0238h	Section 11.1.4.1.47
23Ch	ASRC_OCKLPSTH_0	Reserved	021E 023Ch	Section 11.1.4.1.48
240h	ASRC_OCKZCNT_0	Output Clockzone Count 0 Register	021E 0240h	Section 11.1.4.1.49
244h	ASRC_OCKZCTRL_0	Output Clockzone Control 0 Register	021E 0244h	Section 11.1.4.1.50
248h	ASRC_ICKLPORTL_0	Reserved	021E 0248h	Section 11.1.4.1.51
24Ch	ASRC_ICKLPORTH_0	Reserved	021E 024Ch	Section 11.1.4.1.52
250h	ASRC_OCKLPORTL_0	Reserved	021E 0250h	Section 11.1.4.1.53
254h	ASRC_OCKLPORTH_0	Reserved	021E 0254h	Section 11.1.4.1.54
280h	ASRC_ICKGENSTL_1	Reserved	021E 0280h	Section 11.1.4.1.33
284h	ASRC_ICKGENSTH_1	Reserved	021E 0284h	Section 11.1.4.1.34
288h	ASRC_ICKGENRTL_1	Reserved	021E 0288h	Section 11.1.4.1.35
28Ch	ASRC_ICKGENRTH_1	Reserved	021E 028Ch	Section 11.1.4.1.36
290h	ASRC_ICKLPRTL_1	Reserved	021E 0290h	Section 11.1.4.1.37
294h	ASRC_ICKPRTH_1	Reserved	021E 0294h	Section 11.1.4.1.38
298h	ASRC_ICKZCNT_1	Input Clockzone Count 1 Register	021E 0298h	Section 11.1.4.1.39
29Ch	ASRC_ICKZCTRL_1	Input Clockzone Control 1 Register	021E 029Ch	Section 11.1.4.1.40
2A0h	ASRC_OCKGENSTL_1	Reserved	021E 02A0h	Section 11.1.4.1.41
2A4h	ASRC_OCKGENSTH_1	Reserved	021E 02A4h	Section 11.1.4.1.42

Table 11-11. ASRC Configuration and Status Registers (continued)

Offset	Acronym	Register Name	ASRC_0_CFG Physical Address	Section
2A8h	ASRC_OCKGENRTL_1	Reserved	021E 02A8h	Section 11.1.4.1.43
2ACh	ASRC_OCKGENRTH_1	Reserved	021E 02ACh	Section 11.1.4.1.44
2B0h	ASRC_OCKLPRTL_1	Reserved	021E 02B0h	Section 11.1.4.1.45
2B4h	ASRC_OCKLPRTH_1	Reserved	021E 02B4h	Section 11.1.4.1.46
2B8h	ASRC_OCKLPSTL_1	Reserved	021E 02B8h	Section 11.1.4.1.47
2BCh	ASRC_OCKLPSTH_1	Reserved	021E 02BCh	Section 11.1.4.1.48
2C0h	ASRC_OCKZCNT_1	Output Clockzone Count 1 Register	021E 02C0h	Section 11.1.4.1.49
2C4h	ASRC_OCKZCTRL_1	Output Clockzone Control 1 Register	021E 02C4h	Section 11.1.4.1.50
2C8h	ASRC_ICKLPORTL_1	Reserved	021E 02C8h	Section 11.1.4.1.51
2CCh	ASRC_ICKLPORTH_1	Reserved	021E 02CCh	Section 11.1.4.1.52
2D0h	ASRC_OCKLPORTL_1	Reserved	021E 02D0h	Section 11.1.4.1.53
2D4h	ASRC_OCKLPORTH_1	Reserved	021E 02D4h	Section 11.1.4.1.54
300h	ASRC_ICKGENSTL_2	Reserved	021E 0300h	Section 11.1.4.1.33
304h	ASRC_ICKGENSTH_2	Reserved	021E 0304h	Section 11.1.4.1.34
308h	ASRC_ICKGENRTL_2	Reserved	021E 0308h	Section 11.1.4.1.35
30Ch	ASRC_ICKGENRTH_2	Reserved	021E 030Ch	Section 11.1.4.1.36
310h	ASRC_ICKLPRTL_2	Reserved	021E 0310h	Section 11.1.4.1.37
314h	ASRC_ICKPRTH_2	Reserved	021E 0314h	Section 11.1.4.1.38
318h	ASRC_ICKZCNT_2	Input Clockzone Count 2 Register	021E 0318h	Section 11.1.4.1.39
31Ch	ASRC_ICKZCTRL_2	Input Clockzone Control 2 Register	021E 031Ch	Section 11.1.4.1.40
320h	ASRC_OCKGENSTL_2	Reserved	021E 0320h	Section 11.1.4.1.41
324h	ASRC_OCKGENSTH_2	Reserved	021E 0324h	Section 11.1.4.1.42
328h	ASRC_OCKGENRTL_2	Reserved	021E 0328h	Section 11.1.4.1.43
32Ch	ASRC_OCKGENRTH_2	Reserved	021E 032Ch	Section 11.1.4.1.44
330h	ASRC_OCKLPRTL_2	Reserved	021E 0330h	Section 11.1.4.1.45
334h	ASRC_OCKLPRTH_2	Reserved	021E 0334h	Section 11.1.4.1.46
338h	ASRC_OCKLPSTL_2	Reserved	021E 0338h	Section 11.1.4.1.47

Table 11-11. ASRC Configuration and Status Registers (continued)

Offset	Acronym	Register Name	ASRC_0_CFG Physical Address	Section
33Ch	ASRC_OCKLPSTH_2	Reserved	021E 033Ch	Section 11.1.4.1.48
340h	ASRC_OCKZCNT_2	Output Clockzone Count 2 Register	021E 0340h	Section 11.1.4.1.49
344h	ASRC_OCKZCTRL_2	Output Clockzone Control 2 Register	021E 0344h	Section 11.1.4.1.50
348h	ASRC_ICKLPORTL_2	Reserved	021E 0348h	Section 11.1.4.1.51
34Ch	ASRC_ICKLPORTH_2	Reserved	021E 034Ch	Section 11.1.4.1.52
350h	ASRC_OCKLPORTL_2	Reserved	021E 0350h	Section 11.1.4.1.53
354h	ASRC_OCKLPORTH_2	Reserved	021E 0354h	Section 11.1.4.1.54
380h	ASRC_ICKGENSTL_3	Reserved	021E 0380h	Section 11.1.4.1.33
384h	ASRC_ICKGENSTH_3	Reserved	021E 0384h	Section 11.1.4.1.34
388h	ASRC_ICKGENRTL_3	Reserved	021E 0388h	Section 11.1.4.1.35
38Ch	ASRC_ICKGENRTH_3	Reserved	021E 038Ch	Section 11.1.4.1.36
390h	ASRC_ICKLPRTL_3	Reserved	021E 0390h	Section 11.1.4.1.37
394h	ASRC_ICKPRTH_3	Reserved	021E 0394h	Section 11.1.4.1.38
398h	ASRC_ICKZCNT_3	Input Clockzone Count 3 Register	021E 0398h	Section 11.1.4.1.39
39Ch	ASRC_ICKZCTRL_3	Input Clockzone Control 3 Register	021E 039Ch	Section 11.1.4.1.40
3A0h	ASRC_OCKGENSTL_3	Reserved	021E 03A0h	Section 11.1.4.1.41
3A4h	ASRC_OCKGENSTH_3	Reserved	021E 03A4h	Section 11.1.4.1.42
3A8h	ASRC_OCKGENRTL_3	Reserved	021E 03A8h	Section 11.1.4.1.43
3ACh	ASRC_OCKGENRTH_3	Reserved	021E 03ACh	Section 11.1.4.1.44
3B0h	ASRC_OCKLPRTL_3	Reserved	021E 03B0h	Section 11.1.4.1.45
3B4h	ASRC_OCKLPRTH_3	Reserved	021E 03B4h	Section 11.1.4.1.46
3B8h	ASRC_OCKLPSTL_3	Reserved	021E 03B8h	Section 11.1.4.1.47
3BCh	ASRC_OCKLPSTH_3	Reserved	021E 03BCh	Section 11.1.4.1.48
3C0h	ASRC_OCKZCNT_3	Output Clockzone Count 3 Register	021E 03C0h	Section 11.1.4.1.49
3C4h	ASRC_OCKZCTRL_3	Output Clockzone Control 3 Register	021E 03C4h	Section 11.1.4.1.50
3C8h	ASRC_ICKLPORTL_3	Reserved	021E 03C8h	Section 11.1.4.1.51
3CCh	ASRC_ICKLPORTH_3	Reserved	021E 03CCh	Section 11.1.4.1.52

Table 11-11. ASRC Configuration and Status Registers (continued)

Offset	Acronym	Register Name	ASRC_0_CFG Physical Address	Section
3D0h	ASRC_OCKLPORTL_3	Reserved	021E 03D0h	Section 11.1.4.1.53
3D4h	ASRC_OCKLPORTH_3	Reserved	021E 03D4h	Section 11.1.4.1.54

11.1.4.1.1 ASRC_PID Register (Offset = 0h) [reset = 64000900h]

ASRC_PID is shown in Figure 11-11 and described in Table 11-13.

Table 11-12. ASRC_PID Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-11. ASRC_PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-64000900h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-13. ASRC_PID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	64000900h	TI internal data. Identifies revision of peripheral.

Table 11-14. Register Call Summary for ASRC_PID

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_PID Register (Offset = 0h) [reset = 64000900h]: [0]

11.1.4.1.2 ASRC_SYSCONFIG Register (Offset = 10h) [reset = 0h]

ASRC_SYSCONFIG is shown in Figure 11-12 and described in Table 11-16.

Table 11-15. ASRC_SYSCONFIG Instances

Instance	Physical Address
ASRC_0_CFG	021E 0010h

Figure 11-12. ASRC_SYSCONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						DATA_FORMAT_DISABLE	SOFTRESET
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-16. ASRC_SYSCONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Always read as 0
1	DATA_FORMAT_DISABLE	R/W	0h	If this bit is enabled, data formatting needs to be done outside
0	SOFTRESET	R/W	0h	Write 1'b1 for reset assertion and 1'b1 for reset deassertion.

Table 11-17. Register Call Summary for ASRC_SYSCONFIG

ASRC Functional Description • : [2] • ASRC Data Interface and Formats: [0] • : [2]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_SYSCONFIG Register (Offset = 10h) [reset = 0h]: [0]

11.1.4.1.3 ASRC_IRQEOI Register (Offset = 20h) [reset = 0h]

ASRC_IRQEOI is shown in Figure 11-13 and described in Table 11-19.

Table 11-18. ASRC_IRQEOI Instances

Instance	Physical Address
ASRC_0_CFG	021E 0020h

Figure 11-13. ASRC_IRQEOI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								EOI_VECTOR							
R-0h								R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-19. ASRC_IRQEOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Always read as 0
7-0	EOI_VECTOR	R/W	0h	Number associated with the ipgenericirq for intr output. Write 0x0: Write to input_intr Write 0x1: Write to output_intr Write 0x2: Write to ingroup_intr Write 0x3: Write to outgroup_intr Write 0x4: Write to error_intr Any other write value is ignored. Always read as 0.

Table 11-20. Register Call Summary for ASRC_IRQEOI

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_IRQEOI Register (Offset = 20h) [reset = 0h]: [0]

11.1.4.1.4 ASRC_IFIRQRAW Register (Offset = 24h) [reset = 0h]

ASRC_IFIRQRAW is shown in Figure 11-14 and described in Table 11-22.

Table 11-21. ASRC_IFIRQRAW Instances

Instance	Physical Address
ASRC_0_CFG	021E 0024h

Figure 11-14. ASRC_IFIRQRAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CHANNEL_15_INPUT_FIFO_THRESHOLD_RAW	CHANNEL_14_INPUT_FIFO_THRESHOLD_RAW	CHANNEL_13_INPUT_FIFO_THRESHOLD_RAW	CHANNEL_12_INPUT_FIFO_THRESHOLD_RAW	CHANNEL_11_INPUT_FIFO_THRESHOLD_RAW	CHANNEL_10_INPUT_FIFO_THRESHOLD_RAW	CHANNEL_9_INPUT_FIFO_THRESHOLD_RAW	CHANNEL_8_INPUT_FIFO_THRESHOLD_RAW
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
CHANNEL_7_INPUT_FIFO_THRESHOLD_RAW	CHANNEL_6_INPUT_FIFO_THRESHOLD_RAW	CHANNEL_5_INPUT_FIFO_THRESHOLD_RAW	CHANNEL_4_INPUT_FIFO_THRESHOLD_RAW	CHANNEL_3_INPUT_FIFO_THRESHOLD_RAW	CHANNEL_2_INPUT_FIFO_THRESHOLD_RAW	CHANNEL_1_INPUT_FIFO_THRESHOLD_RAW	CHANNEL_0_INPUT_FIFO_THRESHOLD_RAW
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-22. ASRC_IFIRQRAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15	CHANNEL_15_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 15 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
14	CHANNEL_14_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 14 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
13	CHANNEL_13_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 13 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
12	CHANNEL_12_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 12 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
11	CHANNEL_11_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 11 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
10	CHANNEL_10_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 10 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.

Table 11-22. ASRC_IFIRQRAW Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	CHANNEL_9_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 9 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
8	CHANNEL_8_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 8 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
7	CHANNEL_7_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 7 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
6	CHANNEL_6_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 6 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
5	CHANNEL_5_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 5 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
4	CHANNEL_4_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 4 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
3	CHANNEL_3_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 3 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
2	CHANNEL_2_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 2 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
1	CHANNEL_1_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 1 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
0	CHANNEL_0_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 0 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.

Table 11-23. Register Call Summary for ASRC_IFIRQRAW

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_IFIRQRAW Register (Offset = 24h) [reset = 0h]: [0]

11.1.4.1.5 ASRC_IFIRQENSTS Register (Offset = 28h) [reset = 0h]

ASRC_IFIRQENSTS is shown in Figure 11-15 and described in Table 11-25.

Table 11-24. ASRC_IFIRQENSTS Instances

Instance	Physical Address
ASRC_0_CFG	021E 0028h

Figure 11-15. ASRC_IFIRQENSTS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CHANNEL_15_INPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_14_INPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_13_INPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_12_INPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_11_INPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_10_INPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_9_INPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_8_INPUT_FIFO_THRESHOLD_ENABLED
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
CHANNEL_7_INPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_6_INPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_5_INPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_4_INPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_3_INPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_2_INPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_1_INPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_0_INPUT_FIFO_THRESHOLD_ENABLED
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-25. ASRC_IFIRQENSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15	CHANNEL_15_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 15 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect
14	CHANNEL_14_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 14 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect
13	CHANNEL_13_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 13 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect
12	CHANNEL_12_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 12 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect
11	CHANNEL_11_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 11 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect
10	CHANNEL_10_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 10 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect

Table 11-25. ASRC_IFIRQENSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	CHANNEL_9_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 9 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect
8	CHANNEL_8_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 8 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect
7	CHANNEL_7_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 7 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect
6	CHANNEL_6_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 6 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect
5	CHANNEL_5_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 5 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect
4	CHANNEL_4_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 4 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect
3	CHANNEL_3_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 3 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect
2	CHANNEL_2_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 2 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect
1	CHANNEL_1_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 1 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect
0	CHANNEL_0_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 0 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Input FIFO for this Channel is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect

Table 11-26. Register Call Summary for ASRC_IFIRQENSTS

ASRC Functional Description • ASRC Interrupts and Data Transfer Model: [0] • Input FIFO Interrupts: [0][1]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_IFIRQENSTS Register (Offset = 28h) [reset = 0h]: [0]

11.1.4.1.6 ASRC_IFIRQENSET Register (Offset = 2Ch) [reset = 0h]

ASRC_IFIRQENSET is shown in Figure 11-16 and described in Table 11-28.

Table 11-27. ASRC_IFIRQENSET Instances

Instance	Physical Address
ASRC_0_CFG	021E 002Ch

Figure 11-16. ASRC_IFIRQENSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CHANNEL_15_INPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_14_INPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_13_INPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_12_INPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_11_INPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_10_INPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_9_INPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_8_INPUT_FIFO_THRESHOLD_ENABLE
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
CHANNEL_7_INPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_6_INPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_5_INPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_4_INPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_3_INPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_2_INPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_1_INPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_0_INPUT_FIFO_THRESHOLD_ENABLE
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-28. ASRC_IFIRQENSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15	CHANNEL_15_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 15 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect
14	CHANNEL_14_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 14 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect
13	CHANNEL_13_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 13 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect
12	CHANNEL_12_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 12 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect
11	CHANNEL_11_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 11 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect
10	CHANNEL_10_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 10 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect
9	CHANNEL_9_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 9 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect

Table 11-28. ASRC_IFIRQENSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	CHANNEL_8_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 8 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect
7	CHANNEL_7_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 7 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect
6	CHANNEL_6_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 6 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect
5	CHANNEL_5_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 5 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect
4	CHANNEL_4_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 4 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect
3	CHANNEL_3_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 3 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect
2	CHANNEL_2_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 2 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect
1	CHANNEL_1_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 1 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect
0	CHANNEL_0_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 0 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect

Table 11-29. Register Call Summary for ASRC_IFIRQENSET

ASRC Functional Description • ASRC Interrupts and Data Transfer Model: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_IFIRQENSET Register (Offset = 2Ch) [reset = 0h]: [0]

11.1.4.1.7 ASRC_IFIRQENCLR Register (Offset = 30h) [reset = 0h]

ASRC_IFIRQENCLR is shown in Figure 11-17 and described in Table 11-31.

Table 11-30. ASRC_IFIRQENCLR Instances

Instance	Physical Address
ASRC_0_CFG	021E 0030h

Figure 11-17. ASRC_IFIRQENCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CHANNEL_15_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_14_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_13_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_12_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_11_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_10_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_9_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_8_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
CHANNEL_7_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_6_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_5_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_4_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_3_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_2_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_1_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_0_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-31. ASRC_IFIRQENCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15	CHANNEL_15_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 15 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect
14	CHANNEL_14_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 14 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect
13	CHANNEL_13_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 13 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect
12	CHANNEL_12_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 12 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect
11	CHANNEL_11_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 11 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect
10	CHANNEL_10_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 10 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect
9	CHANNEL_9_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 9 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect

Table 11-31. ASRC_IFIRQENCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	CHANNEL_8_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 8 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect
7	CHANNEL_7_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 7 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect
6	CHANNEL_6_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 6 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect
5	CHANNEL_5_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 5 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect
4	CHANNEL_4_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 4 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect
3	CHANNEL_3_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 3 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect
2	CHANNEL_2_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 2 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect
1	CHANNEL_1_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 1 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect
0	CHANNEL_0_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 0 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect

Table 11-32. Register Call Summary for ASRC_IFIRQENCLR

ASRC Functional Description • ASRC Interrupts and Data Transfer Model: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_IFIRQENCLR Register (Offset = 30h) [reset = 0h]: [0]

11.1.4.1.8 ASRC_OFIRQRAW Register (Offset = 34h) [reset = 0h]

ASRC_OFIRQRAW is shown in Figure 11-18 and described in Table 11-34.

Table 11-33. ASRC_OFIRQRAW Instances

Instance	Physical Address
ASRC_0_CFG	021E 0034h

Figure 11-18. ASRC_OFIRQRAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CHANNEL_15_OUTPUT_FIFO_THRESHOLD_RAW	CHANNEL_14_OUTPUT_FIFO_THRESHOLD_RAW	CHANNEL_13_OUTPUT_FIFO_THRESHOLD_RAW	CHANNEL_12_OUTPUT_FIFO_THRESHOLD_RAW	CHANNEL_11_OUTPUT_FIFO_THRESHOLD_RAW	CHANNEL_10_OUTPUT_FIFO_THRESHOLD_RAW	CHANNEL_9_OUTPUT_FIFO_THRESHOLD_RAW	CHANNEL_8_OUTPUT_FIFO_THRESHOLD_RAW
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
CHANNEL_7_OUTPUT_FIFO_THRESHOLD_RAW	CHANNEL_6_OUTPUT_FIFO_THRESHOLD_RAW	CHANNEL_5_OUTPUT_FIFO_THRESHOLD_RAW	CHANNEL_4_OUTPUT_FIFO_THRESHOLD_RAW	CHANNEL_3_OUTPUT_FIFO_THRESHOLD_RAW	CHANNEL_2_OUTPUT_FIFO_THRESHOLD_RAW	CHANNEL_1_OUTPUT_FIFO_THRESHOLD_RAW	CHANNEL_0_OUTPUT_FIFO_THRESHOLD_RAW
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-34. ASRC_OFIRQRAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15	CHANNEL_15_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 15 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect
14	CHANNEL_14_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 14 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect
13	CHANNEL_13_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 13 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect
12	CHANNEL_12_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 12 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect
11	CHANNEL_11_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 11 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect
10	CHANNEL_10_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 10 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect

Table 11-34. ASRC_OFIRQRAW Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	CHANNEL_9_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 9 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect
8	CHANNEL_8_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 8 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect
7	CHANNEL_7_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 7 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect
6	CHANNEL_6_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 6 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect
5	CHANNEL_5_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 5 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect
4	CHANNEL_4_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 4 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect
3	CHANNEL_3_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 3 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect
2	CHANNEL_2_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 2 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect
1	CHANNEL_1_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 1 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect
0	CHANNEL_0_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Channel 0 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect

Table 11-35. Register Call Summary for ASRC_OFIRQRAW

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OFIRQRAW Register (Offset = 34h) [reset = 0h]: [0]

11.1.4.1.9 ASRC_OFIRQENSTS Register (Offset = 38h) [reset = 0h]

ASRC_OFIRQENSTS is shown in Figure 11-19 and described in Table 11-37.

Table 11-36. ASRC_OFIRQENSTS Instances

Instance	Physical Address
ASRC_0_CFG	021E 0038h

Figure 11-19. ASRC_OFIRQENSTS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CHANNEL_15_OUTPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_14_OUTPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_13_OUTPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_12_OUTPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_11_OUTPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_10_OUTPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_9_OUTPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_8_OUTPUT_FIFO_THRESHOLD_ENABLED
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
CHANNEL_7_OUTPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_6_OUTPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_5_OUTPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_4_OUTPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_3_OUTPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_2_OUTPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_1_OUTPUT_FIFO_THRESHOLD_ENABLED	CHANNEL_0_OUTPUT_FIFO_THRESHOLD_ENABLED
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-37. ASRC_OFIRQENSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15	CHANNEL_15_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 15 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
14	CHANNEL_14_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 14 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
13	CHANNEL_13_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 13 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
12	CHANNEL_12_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 12 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
11	CHANNEL_11_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 11 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
10	CHANNEL_10_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 10 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.

Table 11-37. ASRC_OFIRQENSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	CHANNEL_9_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 9 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
8	CHANNEL_8_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 8 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
7	CHANNEL_7_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 7 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
6	CHANNEL_6_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 6 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
5	CHANNEL_5_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 5 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
4	CHANNEL_4_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 4 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
3	CHANNEL_3_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 3 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
2	CHANNEL_2_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 2 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
1	CHANNEL_1_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 1 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
0	CHANNEL_0_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Channel 0 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when the Output FIFO for this Channel is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.

Table 11-38. Register Call Summary for ASRC_OFIRQENSTS

ASRC Functional Description • ASRC Interrupts and Data Transfer Model: [0] • Output FIFO Interrupts: [0][1]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OFIRQENSTS Register (Offset = 38h) [reset = 0h]: [0]

11.1.4.1.10 ASRC_OFIRQENSET Register (Offset = 3Ch) [reset = 0h]

ASRC_OFIRQENSET is shown in Figure 11-20 and described in Table 11-40.

Table 11-39. ASRC_OFIRQENSET Instances

Instance	Physical Address
ASRC_0_CFG	021E 003Ch

Figure 11-20. ASRC_OFIRQENSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CHANNEL_15_OUTPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_14_OUTPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_13_OUTPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_12_OUTPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_11_OUTPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_10_OUTPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_9_OUTPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_8_OUTPUT_FIFO_THRESHOLD_ENABLE
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
CHANNEL_7_OUTPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_6_OUTPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_5_OUTPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_4_OUTPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_3_OUTPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_2_OUTPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_1_OUTPUT_FIFO_THRESHOLD_ENABLE	CHANNEL_0_OUTPUT_FIFO_THRESHOLD_ENABLE
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-40. ASRC_OFIRQENSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15	CHANNEL_15_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 15 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
14	CHANNEL_14_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 14 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
13	CHANNEL_13_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 13 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
12	CHANNEL_12_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 12 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
11	CHANNEL_11_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 11 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
10	CHANNEL_10_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 10 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
9	CHANNEL_9_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 9 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.

Table 11-40. ASRC_OFIRQENSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	CHANNEL_8_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 8 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
7	CHANNEL_7_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 7 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
6	CHANNEL_6_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 6 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
5	CHANNEL_5_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 5 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
4	CHANNEL_4_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 4 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
3	CHANNEL_3_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 3 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
2	CHANNEL_2_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 2 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
1	CHANNEL_1_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 1 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
0	CHANNEL_0_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Channel 0 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.

Table 11-41. Register Call Summary for ASRC_OFIRQENSET

ASRC Functional Description • ASRC Interrupts and Data Transfer Model: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OFIRQENSET Register (Offset = 3Ch) [reset = 0h]: [0]

11.1.4.1.11 ASRC_OFIRQENCLR Register (Offset = 40h) [reset = 0h]

ASRC_OFIRQENCLR is shown in Figure 11-21 and described in Table 11-43.

Table 11-42. ASRC_OFIRQENCLR Instances

Instance	Physical Address
ASRC_0_CFG	021E 0040h

Figure 11-21. ASRC_OFIRQENCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CHANNEL_15_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_14_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_13_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_12_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_11_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_10_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_9_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_8_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
CHANNEL_7_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_6_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_5_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_4_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_3_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_2_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_1_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	CHANNEL_0_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-43. ASRC_OFIRQENCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15	CHANNEL_15_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 15 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
14	CHANNEL_14_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 14 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
13	CHANNEL_13_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 13 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
12	CHANNEL_12_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 12 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
11	CHANNEL_11_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 11 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
10	CHANNEL_10_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 10 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
9	CHANNEL_9_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 9 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.

Table 11-43. ASRC_OFIRQENCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	CHANNEL_8_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 8 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
7	CHANNEL_7_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 7 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
6	CHANNEL_6_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 6 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
5	CHANNEL_5_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 5 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
4	CHANNEL_4_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 4 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
3	CHANNEL_3_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 3 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
2	CHANNEL_2_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 2 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
1	CHANNEL_1_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 1 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
0	CHANNEL_0_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Channel 0 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.

Table 11-44. Register Call Summary for ASRC_OFIRQENCLR

ASRC Functional Description • ASRC Interrupts and Data Transfer Model: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OFIRQENCLR Register (Offset = 40h) [reset = 0h]: [0]

11.1.4.1.12 ASRC_IGIRQRAW Register (Offset = 44h) [reset = 0h]

ASRC_IGIRQRAW is shown in Figure 11-22 and described in Table 11-46.

Table 11-45. ASRC_IGIRQRAW Instances

Instance	Physical Address
ASRC_0_CFG	021E 0044h

Figure 11-22. ASRC_IGIRQRAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				GROUP_3_INPUT_FIFO_THRESHOLD_RAW	GROUP_2_INPUT_FIFO_THRESHOLD_RAW	GROUP_1_INPUT_FIFO_THRESHOLD_RAW	GROUP_0_INPUT_FIFO_THRESHOLD_RAW
R-0h				R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-46. ASRC_IGIRQRAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Always read as 0
3	GROUP_3_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Group 3 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when all the Input FIFOs for this Group is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
2	GROUP_2_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Group 2 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when all the Input FIFOs for this Group is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
1	GROUP_1_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Group 1 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when all the Input FIFOs for this Group is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
0	GROUP_0_INPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Group 0 Input FIFO Threshold Interrupt. Read indicates raw status. This fires when all the Input FIFOs for this Group is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.

Table 11-47. Register Call Summary for ASRC_IGIRQRAW

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_IGIRQRAW Register (Offset = 44h) [reset = 0h]: [0]

11.1.4.1.13 ASRC_IGIRQENSTS Register (Offset = 48h) [reset = 0h]

ASRC_IGIRQENSTS is shown in Figure 11-23 and described in Table 11-49.

Table 11-48. ASRC_IGIRQENSTS Instances

Instance	Physical Address
ASRC_0_CFG	021E 0048h

Figure 11-23. ASRC_IGIRQENSTS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				GROUP_3_INPUT_FIFO_THRESHOLD_ENABLED	GROUP_2_INPUT_FIFO_THRESHOLD_ENABLED	GROUP_1_INPUT_FIFO_THRESHOLD_ENABLED	GROUP_0_INPUT_FIFO_THRESHOLD_ENABLED
R-0h				R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-49. ASRC_IGIRQENSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Always read as 0
3	GROUP_3_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Group 3 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when all the Input FIFOs for this Group is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
2	GROUP_2_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Group 2 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when all the Input FIFOs for this Group is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
1	GROUP_1_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Group 1 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when all the Input FIFOs for this Group is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
0	GROUP_0_INPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Group 0 Input FIFO Threshold Interrupt. Read indicates enabled status. This fires when all the Input FIFOs for this Group is below the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.

Table 11-50. Register Call Summary for ASRC_IGIRQENSTS

ASRC Functional Description • ASRC Interrupts and Data Transfer Model: [0] • Group Interrupts: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_IGIRQENSTS Register (Offset = 48h) [reset = 0h]: [0]

11.1.4.1.14 ASRC_IGIRQENSET Register (Offset = 4Ch) [reset = 0h]

ASRC_IGIRQENSET is shown in Figure 11-24 and described in Table 11-52.

Table 11-51. ASRC_IGIRQENSET Instances

Instance	Physical Address
ASRC_0_CFG	021E 004Ch

Figure 11-24. ASRC_IGIRQENSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				GROUP_3_INPUT_FIFO_THRESHOLD_ENABLE	GROUP_2_INPUT_FIFO_THRESHOLD_ENABLE	GROUP_1_INPUT_FIFO_THRESHOLD_ENABLE	GROUP_0_INPUT_FIFO_THRESHOLD_ENABLE
R-0h				R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-52. ASRC_IGIRQENSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Always read as 0
3	GROUP_3_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Group 3 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
2	GROUP_2_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Group 2 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
1	GROUP_1_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Group 1 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
0	GROUP_0_INPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Group 0 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.

Table 11-53. Register Call Summary for ASRC_IGIRQENSET

ASRC Functional Description • ASRC Interrupts and Data Transfer Model: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_IGIRQENSET Register (Offset = 4Ch) [reset = 0h]: [0]

11.1.4.1.15 ASRC_IGIRQENCLR Register (Offset = 50h) [reset = 0h]

ASRC_IGIRQENCLR is shown in Figure 11-25 and described in Table 11-55.

Table 11-54. ASRC_IGIRQENCLR Instances

Instance	Physical Address
ASRC_0_CFG	021E 0050h

Figure 11-25. ASRC_IGIRQENCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				GROUP_3_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	GROUP_2_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	GROUP_1_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	GROUP_0_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR
R-0h				R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-55. ASRC_IGIRQENCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Always read as 0
3	GROUP_3_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Group 3 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
2	GROUP_2_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Group 2 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
1	GROUP_1_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Group 1 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
0	GROUP_0_INPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Group 0 Input FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.

Table 11-56. Register Call Summary for ASRC_IGIRQENCLR

ASRC Functional Description • ASRC Interrupts and Data Transfer Model: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_IGIRQENCLR Register (Offset = 50h) [reset = 0h]: [0]

11.1.4.1.16 ASRC_OGIRQRAW Register (Offset = 54h) [reset = 0h]

ASRC_OGIRQRAW is shown in Figure 11-26 and described in Table 11-58.

Table 11-57. ASRC_OGIRQRAW Instances

Instance	Physical Address
ASRC_0_CFG	021E 0054h

Figure 11-26. ASRC_OGIRQRAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				GROUP_3_OUTPUT_FIFO_THRESHOLD_RAW	GROUP_2_OUTPUT_FIFO_THRESHOLD_RAW	GROUP_1_OUTPUT_FIFO_THRESHOLD_RAW	GROUP_0_OUTPUT_FIFO_THRESHOLD_RAW
R-0h				R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-58. ASRC_OGIRQRAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Always read as 0
3	GROUP_3_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Group 3 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when all the Output FIFOs for this Group is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
2	GROUP_2_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Group 2 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when all the Output FIFOs for this Group is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
1	GROUP_1_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Group 1 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when all the Output FIFOs for this Group is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.
0	GROUP_0_OUTPUT_FIFO_THRESHOLD_RAW	R/W1S	0h	Group 0 Output FIFO Threshold Interrupt. Read indicates raw status. This fires when all the Output FIFOs for this Group is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will set status. Writing 0 has no effect.

Table 11-59. Register Call Summary for ASRC_OGIRQRAW

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OGIRQRAW Register (Offset = 54h) [reset = 0h]: [0]

11.1.4.1.17 ASRC_OGIRQENSTS Register (Offset = 58h) [reset = 0h]

ASRC_OGIRQENSTS is shown in Figure 11-27 and described in Table 11-61.

Table 11-60. ASRC_OGIRQENSTS Instances

Instance	Physical Address
ASRC_0_CFG	021E 0058h

Figure 11-27. ASRC_OGIRQENSTS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				GROUP_3_OUTPUT_FIFO_THRESHOLD_ENABLED	GROUP_2_OUTPUT_FIFO_THRESHOLD_ENABLED	GROUP_1_OUTPUT_FIFO_THRESHOLD_ENABLED	GROUP_0_OUTPUT_FIFO_THRESHOLD_ENABLED
R-0h				R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-61. ASRC_OGIRQENSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Always read as 0
3	GROUP_3_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Group 3 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when all the Output FIFOs for this Group is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
2	GROUP_2_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Group 2 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when all the Output FIFOs for this Group is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
1	GROUP_1_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Group 1 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when all the Output FIFOs for this Group is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.
0	GROUP_0_OUTPUT_FIFO_THRESHOLD_ENABLED	R/W1C	0h	Group 0 Output FIFO Threshold Interrupt. Read indicates enabled status. This fires when all the Output FIFOs for this Group is above the level set in the Configuration register. 0 = inactive. 1 = active. Writing 1 will clear status. Writing 0 has no effect.

Table 11-62. Register Call Summary for ASRC_OGIRQENSTS

ASRC Functional Description • ASRC Interrupts and Data Transfer Model: [0] • Group Interrupts: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OGIRQENSTS Register (Offset = 58h) [reset = 0h]: [0]

11.1.4.1.18 ASRC_OGIRQENSET Register (Offset = 5Ch) [reset = 0h]

ASRC_OGIRQENSET is shown in Figure 11-28 and described in Table 11-64.

Table 11-63. ASRC_OGIRQENSET Instances

Instance	Physical Address
ASRC_0_CFG	021E 005Ch

Figure 11-28. ASRC_OGIRQENSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				GROUP_3_OUTPUT_FIFO_THRESHOLD_ENABLE	GROUP_2_OUTPUT_FIFO_THRESHOLD_ENABLE	GROUP_1_OUTPUT_FIFO_THRESHOLD_ENABLE	GROUP_0_OUTPUT_FIFO_THRESHOLD_ENABLE
R-0h				R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-64. ASRC_OGIRQENSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Always read as 0
3	GROUP_3_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Group 3 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
2	GROUP_2_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Group 2 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
1	GROUP_1_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Group 1 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.
0	GROUP_0_OUTPUT_FIFO_THRESHOLD_ENABLE	R/W1S	0h	Group 0 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will set enable. Writing 0 has no effect.

Table 11-65. Register Call Summary for ASRC_OGIRQENSET

ASRC Functional Description • ASRC Interrupts and Data Transfer Model: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OGIRQENSET Register (Offset = 5Ch) [reset = 0h]: [0]

11.1.4.1.19 ASRC_OGIRQENCLR Register (Offset = 60h) [reset = 0h]

ASRC_OGIRQENCLR is shown in Figure 11-29 and described in Table 11-67.

Table 11-66. ASRC_OGIRQENCLR Instances

Instance	Physical Address
ASRC_0_CFG	021E 0060h

Figure 11-29. ASRC_OGIRQENCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				GROUP_3_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	GROUP_2_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	GROUP_1_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	GROUP_0_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR
R-0h				R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-67. ASRC_OGIRQENCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Always read as 0
3	GROUP_3_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Group 3 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
2	GROUP_2_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Group 2 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
1	GROUP_1_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Group 1 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.
0	GROUP_0_OUTPUT_FIFO_THRESHOLD_ENABLE_CLEAR	R/W1C	0h	Group 0 Output FIFO Threshold Interrupt. Read indicates interrupt enable. 0 = inactive. 1 = active. Writing 1 will clear enable. Writing 0 has no effect.

Table 11-68. Register Call Summary for ASRC_OGIRQENCLR

ASRC Functional Description • ASRC Interrupts and Data Transfer Model: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OGIRQENCLR Register (Offset = 60h) [reset = 0h]: [0]

11.1.4.1.20 ASRC_ERIRQRAW Register (Offset = 64h) [reset = 0h]

ASRC_ERIRQRAW is shown in Figure 11-30 and described in Table 11-70.

Table 11-69. ASRC_ERIRQRAW Instances

Instance	Physical Address
ASRC_0_CFG	021E 0064h

Figure 11-30. ASRC_ERIRQRAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CHANNEL_15_ERROR_RAW	CHANNEL_14_ERROR_RAW	CHANNEL_13_ERROR_RAW	CHANNEL_12_ERROR_RAW	CHANNEL_11_ERROR_RAW	CHANNEL_10_ERROR_RAW	CHANNEL_9_ERROR_RAW	CHANNEL_8_ERROR_RAW
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
CHANNEL_7_ERROR_RAW	CHANNEL_6_ERROR_RAW	CHANNEL_5_ERROR_RAW	CHANNEL_4_ERROR_RAW	CHANNEL_3_ERROR_RAW	CHANNEL_2_ERROR_RAW	CHANNEL_1_ERROR_RAW	CHANNEL_0_ERROR_RAW
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-70. ASRC_ERIRQRAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15	CHANNEL_15_ERROR_RAW	R/W1S	0h	Channel 15 FIFOs Error Interrupt Read indicates raw status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will set status Writing 0 has no effect
14	CHANNEL_14_ERROR_RAW	R/W1S	0h	Channel 14 FIFOs Error Interrupt Read indicates raw status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will set status Writing 0 has no effect
13	CHANNEL_13_ERROR_RAW	R/W1S	0h	Channel 13 FIFOs Error Interrupt Read indicates raw status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will set status Writing 0 has no effect
12	CHANNEL_12_ERROR_RAW	R/W1S	0h	Channel 12 FIFOs Error Interrupt Read indicates raw status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will set status Writing 0 has no effect
11	CHANNEL_11_ERROR_RAW	R/W1S	0h	Channel 11 FIFOs Error Interrupt Read indicates raw status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will set status Writing 0 has no effect
10	CHANNEL_10_ERROR_RAW	R/W1S	0h	Channel 10 FIFOs Error Interrupt Read indicates raw status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will set status Writing 0 has no effect
9	CHANNEL_9_ERROR_RAW	R/W1S	0h	Channel 9 FIFOs Error Interrupt Read indicates raw status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will set status Writing 0 has no effect
8	CHANNEL_8_ERROR_RAW	R/W1S	0h	Channel 8 FIFOs Error Interrupt Read indicates raw status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will set status Writing 0 has no effect
7	CHANNEL_7_ERROR_RAW	R/W1S	0h	Channel 7 FIFOs Error Interrupt Read indicates raw status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will set status Writing 0 has no effect

Table 11-70. ASRC_ERIRQRAW Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	CHANNEL_6_ERROR_RAW	R/W1S	0h	Channel 6 FIFOs Error Interrupt Read indicates raw status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will set status Writing 0 has no effect
5	CHANNEL_5_ERROR_RAW	R/W1S	0h	Channel 5 FIFOs Error Interrupt Read indicates raw status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will set status Writing 0 has no effect
4	CHANNEL_4_ERROR_RAW	R/W1S	0h	Channel 4 FIFOs Error Interrupt Read indicates raw status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will set status Writing 0 has no effect
3	CHANNEL_3_ERROR_RAW	R/W1S	0h	Channel 3 FIFOs Error Interrupt Read indicates raw status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will set status Writing 0 has no effect
2	CHANNEL_2_ERROR_RAW	R/W1S	0h	Channel 2 FIFOs Error Interrupt Read indicates raw status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will set status Writing 0 has no effect
1	CHANNEL_1_ERROR_RAW	R/W1S	0h	Channel 1 FIFOs Error Interrupt Read indicates raw status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will set status Writing 0 has no effect
0	CHANNEL_0_ERROR_RAW	R/W1S	0h	Channel 0 FIFOs Error Interrupt Read indicates raw status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will set status Writing 0 has no effect

Table 11-71. Register Call Summary for ASRC_ERIRQRAW

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_ERIRQRAW Register (Offset = 64h) [reset = 0h]: [0]

11.1.4.1.21 ASRC_ERIRQENSTS Register (Offset = 68h) [reset = 0h]

ASRC_ERIRQENSTS is shown in Figure 11-31 and described in Table 11-73.

Table 11-72. ASRC_ERIRQENSTS Instances

Instance	Physical Address
ASRC_0_CFG	021E 0068h

Figure 11-31. ASRC_ERIRQENSTS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CHANNEL_15_ERROR_ENABLED	CHANNEL_14_ERROR_ENABLED	CHANNEL_13_ERROR_ENABLED	CHANNEL_12_ERROR_ENABLED	CHANNEL_11_ERROR_ENABLED	CHANNEL_10_ERROR_ENABLED	CHANNEL_9_ERROR_ENABLED	CHANNEL_8_ERROR_ENABLED
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
CHANNEL_7_ERROR_ENABLED	CHANNEL_6_ERROR_ENABLED	CHANNEL_5_ERROR_ENABLED	CHANNEL_4_ERROR_ENABLED	CHANNEL_3_ERROR_ENABLED	CHANNEL_2_ERROR_ENABLED	CHANNEL_1_ERROR_ENABLED	CHANNEL_0_ERROR_ENABLED
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-73. ASRC_ERIRQENSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15	CHANNEL_15_ERROR_ENABLED	R/W1C	0h	Channel 15 FIFOs Error Interrupt Read indicates enabled status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will clear status Writing 0 has no effect
14	CHANNEL_14_ERROR_ENABLED	R/W1C	0h	Channel 14 FIFOs Error Interrupt Read indicates enabled status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will clear status Writing 0 has no effect
13	CHANNEL_13_ERROR_ENABLED	R/W1C	0h	Channel 13 FIFOs Error Interrupt Read indicates enabled status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will clear status Writing 0 has no effect
12	CHANNEL_12_ERROR_ENABLED	R/W1C	0h	Channel 12 FIFOs Error Interrupt Read indicates enabled status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will clear status Writing 0 has no effect
11	CHANNEL_11_ERROR_ENABLED	R/W1C	0h	Channel 11 FIFOs Error Interrupt Read indicates enabled status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will clear status Writing 0 has no effect
10	CHANNEL_10_ERROR_ENABLED	R/W1C	0h	Channel 10 FIFOs Error Interrupt Read indicates enabled status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will clear status Writing 0 has no effect

Table 11-73. ASRC_ERIRQENSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	CHANNEL_9_ERROR_ENABLED	R/W1C	0h	Channel 9 FIFOs Error Interrupt Read indicates enabled status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will clear status Writing 0 has no effect
8	CHANNEL_8_ERROR_ENABLED	R/W1C	0h	Channel 8 FIFOs Error Interrupt Read indicates enabled status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will clear status Writing 0 has no effect
7	CHANNEL_7_ERROR_ENABLED	R/W1C	0h	Channel 7 FIFOs Error Interrupt Read indicates enabled status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will clear status Writing 0 has no effect
6	CHANNEL_6_ERROR_ENABLED	R/W1C	0h	Channel 6 FIFOs Error Interrupt Read indicates enabled status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will clear status Writing 0 has no effect
5	CHANNEL_5_ERROR_ENABLED	R/W1C	0h	Channel 5 FIFOs Error Interrupt Read indicates enabled status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will clear status Writing 0 has no effect
4	CHANNEL_4_ERROR_ENABLED	R/W1C	0h	Channel 4 FIFOs Error Interrupt Read indicates enabled status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will clear status Writing 0 has no effect
3	CHANNEL_3_ERROR_ENABLED	R/W1C	0h	Channel 3 FIFOs Error Interrupt Read indicates enabled status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will clear status Writing 0 has no effect
2	CHANNEL_2_ERROR_ENABLED	R/W1C	0h	Channel 2 FIFOs Error Interrupt Read indicates enabled status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will clear status Writing 0 has no effect
1	CHANNEL_1_ERROR_ENABLED	R/W1C	0h	Channel 1 FIFOs Error Interrupt Read indicates enabled status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will clear status Writing 0 has no effect
0	CHANNEL_0_ERROR_ENABLED	R/W1C	0h	Channel 0 FIFOs Error Interrupt Read indicates enabled status. This fires when the FIFOs for this Channel isUnderflowed/Overflowed 0 = inactive 1 = active Writing 1 will clear status Writing 0 has no effect

Table 11-74. Register Call Summary for ASRC_ERIRQENSTS

ASRC Functional Description • ASRC Interrupts and Data Transfer Model: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_ERIRQENSTS Register (Offset = 68h) [reset = 0h]: [0]

11.1.4.1.22 ASRC_ERIRQENSET Register (Offset = 6Ch) [reset = 0h]

ASRC_ERIRQENSET is shown in Figure 11-32 and described in Table 11-76.

Table 11-75. ASRC_ERIRQENSET Instances

Instance	Physical Address
ASRC_0_CFG	021E 006Ch

Figure 11-32. ASRC_ERIRQENSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CHANNEL_15_ERROR_ENABLE	CHANNEL_14_ERROR_ENABLE	CHANNEL_13_ERROR_ENABLE	CHANNEL_12_ERROR_ENABLE	CHANNEL_11_ERROR_ENABLE	CHANNEL_10_ERROR_ENABLE	CHANNEL_9_ERROR_ENABLE	CHANNEL_8_ERROR_ENABLE
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
CHANNEL_7_ERROR_ENABLE	CHANNEL_6_ERROR_ENABLE	CHANNEL_5_ERROR_ENABLE	CHANNEL_4_ERROR_ENABLE	CHANNEL_3_ERROR_ENABLE	CHANNEL_2_ERROR_ENABLE	CHANNEL_1_ERROR_ENABLE	CHANNEL_0_ERROR_ENABLE
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-76. ASRC_ERIRQENSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15	CHANNEL_15_ERROR_ENABLE	R/W1S	0h	Channel 15 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will set enable Writing 0 has no effect
14	CHANNEL_14_ERROR_ENABLE	R/W1S	0h	Channel 14 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will set enable Writing 0 has no effect
13	CHANNEL_13_ERROR_ENABLE	R/W1S	0h	Channel 13 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will set enable Writing 0 has no effect
12	CHANNEL_12_ERROR_ENABLE	R/W1S	0h	Channel 12 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will set enable Writing 0 has no effect
11	CHANNEL_11_ERROR_ENABLE	R/W1S	0h	Channel 11 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will set enable Writing 0 has no effect
10	CHANNEL_10_ERROR_ENABLE	R/W1S	0h	Channel 10 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will set enable Writing 0 has no effect
9	CHANNEL_9_ERROR_ENABLE	R/W1S	0h	Channel 9 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will set enable Writing 0 has no effect
8	CHANNEL_8_ERROR_ENABLE	R/W1S	0h	Channel 8 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will set enable Writing 0 has no effect
7	CHANNEL_7_ERROR_ENABLE	R/W1S	0h	Channel 7 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will set enable Writing 0 has no effect
6	CHANNEL_6_ERROR_ENABLE	R/W1S	0h	Channel 6 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will set enable Writing 0 has no effect
5	CHANNEL_5_ERROR_ENABLE	R/W1S	0h	Channel 5 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will set enable Writing 0 has no effect

Table 11-76. ASRC_ERIRQENSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	CHANNEL_4_ERROR_ENABLE	R/W1S	0h	Channel 4 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will set enable Writing 0 has no effect
3	CHANNEL_3_ERROR_ENABLE	R/W1S	0h	Channel 3 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will set enable Writing 0 has no effect
2	CHANNEL_2_ERROR_ENABLE	R/W1S	0h	Channel 2 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will set enable Writing 0 has no effect
1	CHANNEL_1_ERROR_ENABLE	R/W1S	0h	Channel 1 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will set enable Writing 0 has no effect
0	CHANNEL_0_ERROR_ENABLE	R/W1S	0h	Channel 10 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will set enable Writing 0 has no effect

Table 11-77. Register Call Summary for ASRC_ERIRQENSET

ASRC Functional Description • ASRC Interrupts and Data Transfer Model: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_ERIRQENSET Register (Offset = 6Ch) [reset = 0h]: [0]

11.1.4.1.23 ASRC_ERIRQENCLR Register (Offset = 70h) [reset = 0h]

ASRC_ERIRQENCLR is shown in Figure 11-33 and described in Table 11-79.

Table 11-78. ASRC_ERIRQENCLR Instances

Instance	Physical Address
ASRC_0_CFG	021E 0070h

Figure 11-33. ASRC_ERIRQENCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CHANNEL_15_ERROR_ENABLE_CLEAR	CHANNEL_14_ERROR_ENABLE_CLEAR	CHANNEL_13_ERROR_ENABLE_CLEAR	CHANNEL_12_ERROR_ENABLE_CLEAR	CHANNEL_11_ERROR_ENABLE_CLEAR	CHANNEL_10_ERROR_ENABLE_CLEAR	CHANNEL_9_ERROR_ENABLE_CLEAR	CHANNEL_8_ERROR_ENABLE_CLEAR
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
CHANNEL_7_ERROR_ENABLE_CLEAR	CHANNEL_6_ERROR_ENABLE_CLEAR	CHANNEL_5_ERROR_ENABLE_CLEAR	CHANNEL_4_ERROR_ENABLE_CLEAR	CHANNEL_3_ERROR_ENABLE_CLEAR	CHANNEL_2_ERROR_ENABLE_CLEAR	CHANNEL_1_ERROR_ENABLE_CLEAR	CHANNEL_0_ERROR_ENABLE_CLEAR
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-79. ASRC_ERIRQENCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15	CHANNEL_15_ERROR_ENABLE_CLEAR	R/W1C	0h	Channel 15 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will clear enable Writing 0 has no effect
14	CHANNEL_14_ERROR_ENABLE_CLEAR	R/W1C	0h	Channel 14 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will clear enable Writing 0 has no effect
13	CHANNEL_13_ERROR_ENABLE_CLEAR	R/W1C	0h	Channel 13 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will clear enable Writing 0 has no effect
12	CHANNEL_12_ERROR_ENABLE_CLEAR	R/W1C	0h	Channel 12 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will clear enable Writing 0 has no effect
11	CHANNEL_11_ERROR_ENABLE_CLEAR	R/W1C	0h	Channel 11 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will clear enable Writing 0 has no effect
10	CHANNEL_10_ERROR_ENABLE_CLEAR	R/W1C	0h	Channel 10 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will clear enable Writing 0 has no effect
9	CHANNEL_9_ERROR_ENABLE_CLEAR	R/W1C	0h	Channel 9 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will clear enable Writing 0 has no effect
8	CHANNEL_8_ERROR_ENABLE_CLEAR	R/W1C	0h	Channel 8 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will clear enable Writing 0 has no effect
7	CHANNEL_7_ERROR_ENABLE_CLEAR	R/W1C	0h	Channel 7 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will clear enable Writing 0 has no effect

Table 11-79. ASRC_ERIRQENCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	CHANNEL_6_ERROR_ENABLE_CLEAR	R/W1C	0h	Channel 6 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will clear enable Writing 0 has no effect
5	CHANNEL_5_ERROR_ENABLE_CLEAR	R/W1C	0h	Channel 5 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will clear enable Writing 0 has no effect
4	CHANNEL_4_ERROR_ENABLE_CLEAR	R/W1C	0h	Channel 4 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will clear enable Writing 0 has no effect
3	CHANNEL_3_ERROR_ENABLE_CLEAR	R/W1C	0h	Channel 3 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will clear enable Writing 0 has no effect
2	CHANNEL_2_ERROR_ENABLE_CLEAR	R/W1C	0h	Channel 2 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will clear enable Writing 0 has no effect
1	CHANNEL_1_ERROR_ENABLE_CLEAR	R/W1C	0h	Channel 1 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will clear enable Writing 0 has no effect
0	CHANNEL_0_ERROR_ENABLE_CLEAR	R/W1C	0h	Channel 0 FIFOs Error Interrupt Read indicates interrupt enable. 0 = inactive 1 = active Writing 1 will clear enable Writing 0 has no effect

Table 11-80. Register Call Summary for ASRC_ERIRQENCLR

ASRC Functional Description • ASRC Interrupts and Data Transfer Model: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_ERIRQENCLR Register (Offset = 70h) [reset = 0h]: [0]

11.1.4.1.24 ASRC_IGRPSEL_0 to ASRC_IGRPSEL_3 Register (Offset = 74h to 80h) [reset = 0h]

ASRC_IGRPSEL_0 to ASRC_IGRPSEL_3 is shown in Figure 11-34 and described in Table 11-82.

Table 11-81. ASRC_IGRPSEL_0 to ASRC_IGRPSEL_3 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0074h to 021E 0080h

Figure 11-34. ASRC_IGRPSEL_0 to ASRC_IGRPSEL_3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CHANNEL_15_GROUP_ENABLE	CHANNEL_14_GROUP_ENABLE	CHANNEL_13_GROUP_ENABLE	CHANNEL_12_GROUP_ENABLE	CHANNEL_11_GROUP_ENABLE	CHANNEL_10_GROUP_ENABLE	CHANNEL_9_GROUP_ENABLE	CHANNEL_8_GROUP_ENABLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CHANNEL_7_GROUP_ENABLE	CHANNEL_6_GROUP_ENABLE	CHANNEL_5_GROUP_ENABLE	CHANNEL_4_GROUP_ENABLE	CHANNEL_3_GROUP_ENABLE	CHANNEL_2_GROUP_ENABLE	CHANNEL_1_GROUP_ENABLE	CHANNEL_0_GROUP_ENABLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-82. ASRC_IGRPSEL_0 to ASRC_IGRPSEL_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15	CHANNEL_15_GROUP_ENABLE	R/W	0h	Input Channel 15 Group Enable indicates if the Input Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Input Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
14	CHANNEL_14_GROUP_ENABLE	R/W	0h	Input Channel 14 Group Enable indicates if the Input Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Input Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
13	CHANNEL_13_GROUP_ENABLE	R/W	0h	Input Channel 13 Group Enable indicates if the Input Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Input Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
12	CHANNEL_12_GROUP_ENABLE	R/W	0h	Input Channel 12 Group Enable indicates if the Input Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Input Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
11	CHANNEL_11_GROUP_ENABLE	R/W	0h	Input Channel 11 Group Enable indicates if the Input Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Input Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable

Table 11-82. ASRC_IGRPSEL_0 to ASRC_IGRPSEL_3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	CHANNEL_10_GROUP_ENABLE	R/W	0h	Input Channel 10 Group Enable indicates if the Input Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Input Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
9	CHANNEL_9_GROUP_ENABLE	R/W	0h	Input Channel 9 Group Enable indicates if the Input Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Input Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
8	CHANNEL_8_GROUP_ENABLE	R/W	0h	Input Channel 8 Group Enable indicates if the Input Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Input Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
7	CHANNEL_7_GROUP_ENABLE	R/W	0h	Input Channel 7 Group Enable indicates if the Input Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Input Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
6	CHANNEL_6_GROUP_ENABLE	R/W	0h	Input Channel 6 Group Enable indicates if the Input Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Input Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
5	CHANNEL_5_GROUP_ENABLE	R/W	0h	Input Channel 5 Group Enable indicates if the Input Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Input Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
4	CHANNEL_4_GROUP_ENABLE	R/W	0h	Input Channel 4 Group Enable indicates if the Input Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Input Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
3	CHANNEL_3_GROUP_ENABLE	R/W	0h	Input Channel 3 Group Enable indicates if the Input Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Input Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
2	CHANNEL_2_GROUP_ENABLE	R/W	0h	Input Channel 2 Group Enable indicates if the Input Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Input Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
1	CHANNEL_1_GROUP_ENABLE	R/W	0h	Input Channel 1 Group Enable indicates if the Input Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Input Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
0	CHANNEL_0_GROUP_ENABLE	R/W	0h	Input Channel 0 Group Enable indicates if the Input Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Input Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable

Table 11-83. Register Call Summary for ASRC_IGRPSEL_0

ASRC Functional Description • Group Mode Configuration Sequence: [0] • ASRC Programming Ranges and Restrictions: [0] • Group Interrupts: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_IGRPSEL_0 to ASRC_IGRPSEL_3 Register (Offset = 74h to 80h) [reset = 0h]: [0]

11.1.4.1.25 ASRC_OGRPSEL_0 to ASRC_OGRPSEL_3 Register (Offset = 94h to A0h) [reset = 0h]

ASRC_OGRPSEL_0 to ASRC_OGRPSEL_3 is shown in Figure 11-35 and described in Table 11-85.

Table 11-84. ASRC_OGRPSEL_0 to ASRC_OGRPSEL_3 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0094h to 021E 00A0h

Figure 11-35. ASRC_OGRPSEL_0 to ASRC_OGRPSEL_3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CHANNEL_15_GROUP_ENABLE	CHANNEL_14_GROUP_ENABLE	CHANNEL_13_GROUP_ENABLE	CHANNEL_12_GROUP_ENABLE	CHANNEL_11_GROUP_ENABLE	CHANNEL_10_GROUP_ENABLE	CHANNEL_9_GROUP_ENABLE	CHANNEL_8_GROUP_ENABLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CHANNEL_7_GROUP_ENABLE	CHANNEL_6_GROUP_ENABLE	CHANNEL_5_GROUP_ENABLE	CHANNEL_4_GROUP_ENABLE	CHANNEL_3_GROUP_ENABLE	CHANNEL_2_GROUP_ENABLE	CHANNEL_1_GROUP_ENABLE	CHANNEL_0_GROUP_ENABLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-85. ASRC_OGRPSEL_0 to ASRC_OGRPSEL_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15	CHANNEL_15_GROUP_ENABLE	R/W	0h	Output Channel 15 Group Enable indicates if the Output Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Output Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
14	CHANNEL_14_GROUP_ENABLE	R/W	0h	Output Channel 14 Group Enable indicates if the Output Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Output Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
13	CHANNEL_13_GROUP_ENABLE	R/W	0h	Output Channel 13 Group Enable indicates if the Output Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Output Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
12	CHANNEL_12_GROUP_ENABLE	R/W	0h	Output Channel 12 Group Enable indicates if the Output Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Output Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
11	CHANNEL_11_GROUP_ENABLE	R/W	0h	Output Channel 11 Group Enable indicates if the Output Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Output Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable

Table 11-85. ASRC_OGRPSEL_0 to ASRC_OGRPSEL_3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	CHANNEL_10_GROUP_ENABLE	R/W	0h	Output Channel 10 Group Enable indicates if the Output Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Output Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
9	CHANNEL_9_GROUP_ENABLE	R/W	0h	Output Channel 9 Group Enable indicates if the Output Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Output Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
8	CHANNEL_8_GROUP_ENABLE	R/W	0h	Output Channel 8 Group Enable indicates if the Output Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Output Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
7	CHANNEL_7_GROUP_ENABLE	R/W	0h	Output Channel 7 Group Enable indicates if the Output Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Output Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
6	CHANNEL_6_GROUP_ENABLE	R/W	0h	Output Channel 6 Group Enable indicates if the Output Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Output Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
5	CHANNEL_5_GROUP_ENABLE	R/W	0h	Output Channel 5 Group Enable indicates if the Output Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Output Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
4	CHANNEL_4_GROUP_ENABLE	R/W	0h	Output Channel 4 Group Enable indicates if the Output Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Output Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
3	CHANNEL_3_GROUP_ENABLE	R/W	0h	Output Channel 3 Group Enable indicates if the Output Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Output Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
2	CHANNEL_2_GROUP_ENABLE	R/W	0h	Output Channel 2 Group Enable indicates if the Output Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Output Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
1	CHANNEL_1_GROUP_ENABLE	R/W	0h	Output Channel 1 Group Enable indicates if the Output Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Output Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable
0	CHANNEL_0_GROUP_ENABLE	R/W	0h	Output Channel 0 Group Enable indicates if the Output Channel is part of the group0 for the group settings and interrupt event generation. If this bit is set then the group event and interrupt will not fire until the threshold condition for this Output Channel is met. 0 = inactive 1 = active Writing 1 will enable Writing 0 will disable

Table 11-86. Register Call Summary for ASRC_OGRPSEL_0

ASRC Functional Description • Group Mode Configuration Sequence: [0] • ASRC Programming Ranges and Restrictions: [0] • Group Interrupts: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OGRPSEL_0 to ASRC_OGRPSEL_3 Register (Offset = 94h to A0h) [reset = 0h]: [0]

11.1.4.1.26 ASRC_ICKDIV Register (Offset = 400h) [reset = 10001h]

ASRC_ICKDIV is shown in Figure 11-36 and described in Table 11-88.

Table 11-87. ASRC_ICKDIV Instances

Instance	Physical Address
ASRC_0_CFG	021E 0400h

Figure 11-36. ASRC_ICKDIV Register

31	30	29	28	27	26	25	24
RESERVED	CLOCK_ZONE 1_DIVIDE_EN	CLOCK_ZONE1_DIVIDE					
R-0h	R/W-0h	R/W-1h					
23	22	21	20	19	18	17	16
CLOCK_ZONE1_DIVIDE							
R/W-1h							
15	14	13	12	11	10	9	8
RESERVED	CLOCK_ZONE 0_DIVIDE_EN	CLOCK_ZONE0_DIVIDE					
R-0h	R/W-0h	R/W-1h					
7	6	5	4	3	2	1	0
CLOCK_ZONE0_DIVIDE							
R/W-1h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-88. ASRC_ICKDIV Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Always read as 0
30	CLOCK_ZONE1_DIVIDE_EN	R/W	0h	Clock zone 1 divider enable. 1'b1:Divider logic enable. 1'b0:Divider logic disable.
29-16	CLOCK_ZONE1_DIVIDE	R/W	1h	This is the setting for input clock zone1 sync signal division value
15	RESERVED	R	0h	Always read as 0
14	CLOCK_ZONE0_DIVIDE_EN	R/W	0h	Clock zone 0 divider enable. 1'b1:Divider logic enable. 1'b0:Divider logic disable.
13-0	CLOCK_ZONE0_DIVIDE	R/W	1h	This is the setting for input clock zone0 sync signal division value

Table 11-89. Register Call Summary for ASRC_ICKDIV

ASRC Functional Description • Group Mode Configuration Sequence: [0][1][2][3] • Stream Mode Configuration Sequence: [0][1][2][3] • ASRC Clock Configuration: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_ICKDIV Register (Offset = 400h) [reset = 10001h]: [0]

11.1.4.1.27 ASRC_OCKDIV Register (Offset = 404h) [reset = 10001h]

ASRC_OCKDIV is shown in Figure 11-37 and described in Table 11-91.

Table 11-90. ASRC_OCKDIV Instances

Instance	Physical Address
ASRC_0_CFG	021E 0404h

Figure 11-37. ASRC_OCKDIV Register

31	30	29	28	27	26	25	24
RESERVED	CLOCK_ZONE 1_DIVIDE_EN	CLOCK_ZONE1_DIVIDE					
R-0h	R/W-0h	R/W-1h					
23	22	21	20	19	18	17	16
CLOCK_ZONE1_DIVIDE							
R/W-1h							
15	14	13	12	11	10	9	8
RESERVED	CLOCK_ZONE 0_DIVIDE_EN	CLOCK_ZONE0_DIVIDE					
R-0h	R/W-0h	R/W-1h					
7	6	5	4	3	2	1	0
CLOCK_ZONE0_DIVIDE							
R/W-1h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-91. ASRC_OCKDIV Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Always read as 0
30	CLOCK_ZONE1_DIVIDE_EN	R/W	0h	Clock zone 1 divider enable. 1'b1:Divider logic enable. 1'b0:Divider logic disable.
29-16	CLOCK_ZONE1_DIVIDE	R/W	1h	This is the setting for output clock zone1 sync signal division value
15	RESERVED	R	0h	Always read as 0
14	CLOCK_ZONE0_DIVIDE_EN	R/W	0h	Clock zone 0 divider enable. 1'b1:Divider logic enable. 1'b0:Divider logic disable.
13-0	CLOCK_ZONE0_DIVIDE	R/W	1h	This is the setting for output clock zone0 sync signal division value

Table 11-92. Register Call Summary for ASRC_OCKDIV

ASRC Functional Description • Group Mode Configuration Sequence: [0][1][2][3] • Stream Mode Configuration Sequence: [0][1][2][3] • ASRC Clock Configuration: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OCKDIV Register (Offset = 404h) [reset = 10001h]: [0]

11.1.4.1.28 ASRC_SRCFFCTRL_0 Register (Offset = 100h + [i * 10h]) [reset = 10001h]

ASRC_SRCFFCTRL_0 is shown in Figure 11-38 and described in Table 11-94.

Table 11-93. ASRC_SRCFFCTRL_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-38. ASRC_SRCFFCTRL_0 Register

31	30	29	28	27	26	25	24
RESERVED				R_CHANNEL_OUTFIFO_UNDERFLOW	L_CHANNEL_OUTFIFO_UNDERFLOW	R_CHANNEL_OUTFIFO_OVERFLOW	L_CHANNEL_OUTFIFO_OVERFLOW
R-0h				R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
OUTFIFO_THRESHOLD							
R/W-1h							
15	14	13	12	11	10	9	8
RESERVED				R_CHANNEL_INFIFO_UNDERFLOW	L_CHANNEL_INFIFO_UNDERFLOW	R_CHANNEL_INFIFO_OVERFLOW	L_CHANNEL_INFIFO_OVERFLOW
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
INFIFO_THRESHOLD							
R/W-1h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-94. ASRC_SRCFFCTRL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Always read as 0
27	R_CHANNEL_OUTFIFO_UNDERFLOW	R	0h	This signal is set to 1 when Right Channel OUTFIFO for SRC0 is underflowed
26	L_CHANNEL_OUTFIFO_UNDERFLOW	R	0h	This signal is set to 1 when Left Channel OUTFIFO for SRC0 is underflowed
25	R_CHANNEL_OUTFIFO_OVERFLOW	R	0h	This signal is set to 1 when Right Channel OUTFIFO for SRC0 is overflowed
24	L_CHANNEL_OUTFIFO_OVERFLOW	R	0h	This signal is set to 1 when Left Channel OUTFIFO for SRC0 is overflowed
23-16	OUTFIFO_THRESHOLD	R/W	1h	This is the number of samples that must be available for the interrupt and SRC0 OUTFIFOs event to fire. 1-32: 1-32 samples
15-12	RESERVED	R	0h	Reserved
11	R_CHANNEL_INFIFO_UNDERFLOW	R	0h	This signal is set to 1 when Right Channel INFIFO for SRC0 is underflowed
10	L_CHANNEL_INFIFO_UNDERFLOW	R	0h	This signal is set to 1 when Left Channel INFIFO for SRC0 is underflowed
9	R_CHANNEL_INFIFO_OVERFLOW	R	0h	This signal is set to 1 when Right Channel INFIFO for SRC0 is overflowed
8	L_CHANNEL_INFIFO_OVERFLOW	R	0h	This signal is set to 1 when Left Channel INFIFO for SRC0 is overflowed
7-0	INFIFO_THRESHOLD	R/W	1h	This is the number of samples that must be available for the interrupt and SRC0 INFIFOs event to fire. 1-32: 1-32 samples

Table 11-95. Register Call Summary for ASRC_SRCFFCTRL_0

ASRC Functional Description • Error Interrupts: [0] • ASRC DMA Events and Data Transfer Model: [0][1] • Stream Mode DMA Events: [0][1] • ASRC Programming Ranges and Restrictions: [0] • Output FIFO Interrupts: [0] • ASRC Data Rate and Traffic Behavior: [0] • Input FIFO Interrupts: [0] • Stream Mode Configuration Sequence: [0][1][2][3]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_SRCFFCTRL_0 Register (Offset = 100h + [i * 10h]) [reset = 10001h]: [0]

11.1.4.1.29 ASRC_SRCCTRL_0 Register (Offset = 104h + [i * 10h]) [reset = 0h]

ASRC_SRCCTRL_0 is shown in Figure 11-39 and described in Table 11-97.

Table 11-96. ASRC_SRCCTRL_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-39. ASRC_SRCCTRL_0 Register

31	30	29	28	27	26	25	24
CHANNEL_ENABLE		OUTPUT_WORD_LENGTH		GROUP_DELAY		DE_EMPHASIS_MODE	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
ATTENUATION							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED					DIRECT_DOWN_SAMPLE	MUTE	DITHER_ENABLE
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INPUT_WORD_LENGTH		OUTPUT_CLOCK_ZONE_SELECT			INPUT_CLOCK_ZONE_SELECT		
R/W-0h		R/W-0h			R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-97. ASRC_SRCCTRL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	CHANNEL_ENABLE	R/W	0h	Setting this enables the Channels of SRC0. The output FIFOs for the stream must be enabled before the input FIFOs. 00: No channel enabled 01: Only Right Channel enabled 10: Only Left Channel enabled 11: Both channels enabled Clearing this register will clear the data path.
29-28	OUTPUT_WORD_LENGTH	R/W	0h	These bits select the word length for the SRC0 output data. The word length reduction is performed by utilizing triangular PDF dithering. 0: 24 Bits (Default) 1: 20 Bits 2: 18 Bits 3: 16 Bits
27-26	GROUP_DELAY	R/W	0h	These bits select the interpolation filter group delay by configuring the number of samples which are pre-buffered prior to the re-sampler function. 0: 64 Samples (Default) 1: 32 Samples 2: 16 Samples 3: 8 Samples
25-24	DE_EMPHASIS_MODE	R/W	0h	These bits are utilized to enable or disable the digital de-emphasis filter manually. The de-emphasis filter is intended to process 50/15 s pre-emphasized audio material at the following input sampling rates. 0: De-Emphasis Disabled (Default) 1: De-Emphasis Enabled for fS = 48kHz 2: De-Emphasis Enabled for fS = 44.1kHz 3: De-Emphasis Enabled for fS = 32kHz
23-16	ATTENUATION	R/W	0h	These bits are utilized to configure the SRC digital output attenuation for the stream Output Attenuation (dB) = N 0.5, where N = Attenuation[7:0]DEC.
15-11	RESERVED	R	0h	Always read as 0
10	DIRECT_DOWN_SAMPLE	R/W	0h	This bit selects the mode of the decimation function, either true decimation filter or direct down-sampling without filtering. DDN Decimation Function 0 Decimation Filter (Default) 1 Direct Down Sampling Note: Direct down-sampling should only be used when the output sampling rate is higher than the input sampling rate. When the output sampling rate is equal to or lower than the input sampling rate, the Decimation Filter must be used in order to avoid aliasing.

Table 11-97. ASRC_SRCCTRL_0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	MUTE	R/W	0h	This bit enables or disables the SRC output soft mute function. 0: Mute Disabled (Default) 1: Mute enabled; output data set to all zeros.
8	DITHER_ENABLE	R/W	0h	This bit enables or disables the SRC filter s dithering. 0: Dithering Disabled (Default) 1: Dithering enabled
7-6	INPUT_WORD_LENGTH	R/W	0h	These bits select the word length for the SRC0 input data. 0: 24 Bits (Default) 1: 20 Bits 2: 18 Bits 3: 16 Bits
5-3	OUTPUT_CLOCK_ZONE_SELECT	R/W	0h	This selects the output clock zone for the stream0. It can select between the 4 external audio clock zones. Based on this number rate and stamp is passed to the Stream0 Sample rate converter from respective clock recovery loop. 00: This selects the Clock Recovery loop 0. 01: This selects the Clock Recovery loop 1. 10: This selects the Clock Recovery loop 2. 11: This selects the Clock Recovery loop 3. In this SoC other values are not valid.
2-0	INPUT_CLOCK_ZONE_SELECT	R/W	0h	This selects the input clock zone for the stream0. It can select between the 4 external audio clock zones. Based on this number rate and stamp is passed to the Stream0 Sample rate converter from respective clock recovery loop. 000: This selects the Clock Recovery loop 0. 001: This selects the Clock Recovery loop 1. 010: This selects the Clock Recovery loop 2. 011: This selects the Clock Recovery loop 3. In this SoC other values are not valid.

Table 11-98. Register Call Summary for ASRC_SRCCTRL_0

ASRC Functional Description • Stream Mode Configuration Sequence: [0][1][2][3][4][5][6][7][8][9][10]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_SRCCTRL_0 Register (Offset = 104h + [i * 10h]) [reset = 0h]: [0]

11.1.4.1.30 ASRC_SRCSTS_0 Register (Offset = 108h + [i * 10h]) [reset = 0h]

ASRC_SRCSTS_0 is shown in Figure 11-40 and described in Table 11-100.

Table 11-99. ASRC_SRCSTS_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-40. ASRC_SRCSTS_0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							UPSAMPLE
R-0h							R-0h
15	14	13	12	11	10	9	8
RATE_RATIO							
R-0h							
7	6	5	4	3	2	1	0
RATE_RATIO							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-100. ASRC_SRCSTS_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Always read as 0
16	UPSAMPLE	R	0h	This signal becomes high when Output sampling rate is greater than Input sampling ratio
15-0	RATE_RATIO	R	0h	Input to Output sampling ratio for SRC0

Table 11-101. Register Call Summary for ASRC_SRCSTS_0

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_SRCSTS_0 Register (Offset = 108h + [i * 10h]) [reset = 0h]: [0]

11.1.4.1.31 ASRC_GFFCTRL_0 Register (Offset = 180h + [i * 8h]) [reset = 10001h]

ASRC_GFFCTRL_0 is shown in Figure 11-41 and described in Table 11-103.

Table 11-102. ASRC_GFFCTRL_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-41. ASRC_GFFCTRL_0 Register

31	30	29	28	27	26	25	24
RESERVED				R_CHANNEL_OUTFIFO_UNDERFLOW	L_CHANNEL_OUTFIFO_UNDERFLOW	R_CHANNEL_OUTFIFO_OVERFLOW	L_CHANNEL_OUTFIFO_OVERFLOW
R-0h				R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
OUTFIFO_THRESHOLD							
R/W-1h							
15	14	13	12	11	10	9	8
RESERVED				R_CHANNEL_INFIFO_UNDERFLOW	L_CHANNEL_INFIFO_UNDERFLOW	R_CHANNEL_INFIFO_OVERFLOW	L_CHANNEL_INFIFO_OVERFLOW
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
INFIFO_THRESHOLD							
R/W-1h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-103. ASRC_GFFCTRL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Always read as 0
27	R_CHANNEL_OUTFIFO_UNDERFLOW	R	0h	This signal is set to 1 when any of the Right Channel OUTFIFO for Group0 SRCs is underflowed
26	L_CHANNEL_OUTFIFO_UNDERFLOW	R	0h	This signal is set to 1 when any of the Left Channel OUTFIFO for Group0 SRCs is underflowed
25	R_CHANNEL_OUTFIFO_OVERFLOW	R	0h	This signal is set to 1 when any of the Right Channel OUTFIFO for Group0 SRCs is overflowed
24	L_CHANNEL_OUTFIFO_OVERFLOW	R	0h	This signal is set to 1 when any of the Left Channel OUTFIFO for Group0 SRCs is overflowed
23-16	OUTFIFO_THRESHOLD	R/W	1h	This is the number of samples that must be available for the interrupt and Group0 SRCs OUTFIFOs event to fire.
15-12	RESERVED	R	0h	Always read as 0
11	R_CHANNEL_INFIFO_UNDERFLOW	R	0h	This signal is set to 1 when any of the Right Channel INFIFO for Group0 SRCs is underflowed
10	L_CHANNEL_INFIFO_UNDERFLOW	R	0h	This signal is set to 1 when any of the Left Channel INFIFO for Group0 SRCs is underflowed
9	R_CHANNEL_INFIFO_OVERFLOW	R	0h	This signal is set to 1 when any of the Right Channel INFIFO for Group0 SRCs is overflowed
8	L_CHANNEL_INFIFO_OVERFLOW	R	0h	This signal is set to 1 when any of the Left Channel INFIFO for Group0 SRCs is overflowed
7-0	INFIFO_THRESHOLD	R/W	1h	This is the number of samples that must be available for the interrupt and Group0 SRCs INFIFOs event to fire.

Table 11-104. Register Call Summary for ASRC_GFFCTRL_0

ASRC Functional Description • Group Mode Configuration Sequence: [0][1][2][3] • ASRC Programming Ranges and Restrictions: [0] • Group Mode DMA Events: [0][1]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_GFFCTRL_0 Register (Offset = 180h + [i * 8h]) [reset = 10001h]: [0]

11.1.4.1.32 ASRC_GSRCCTRL_0 Register (Offset = 184h + [i * 8h]) [reset = 0h]

ASRC_GSRCCTRL_0 is shown in Figure 11-42 and described in Table 11-106.

Table 11-105. ASRC_GSRCCTRL_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-42. ASRC_GSRCCTRL_0 Register

31	30	29	28	27	26	25	24
CHANNEL_ENABLE		OUTPUT_WORD_LENGTH		GROUP_DELAY		DE_EMPHASIS_MODE	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
ATTENUATION							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED					DIRECT_DOWN_SAMPLE	MUTE	DITHER_ENABLE
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INPUT_WORD_LENGTH		OUTPUT_CLOCK_ZONE_SELECT			INPUT_CLOCK_ZONE_SELECT		
R/W-0h		R/W-0h			R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-106. ASRC_GSRCCTRL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	CHANNEL_ENABLE	R/W	0h	Setting this enables the Channels of Group0 SRCs. The output FIFOs for the stream must be enabled before the input FIFOs. 00: No channel enabled 01: Only Right Channel enabled 10: Only Left Channel enabled 11: Both channels enabled Clearing this register will clear the data path
29-28	OUTPUT_WORD_LENGTH	R/W	0h	These bits select the word length for the Group0 SRC output data. The word length reduction is performed by utilizing triangular PDF dithering. 0: 24 Bits (Default) 1: 20 Bits 2: 18 Bits 3: 16 Bits
27-26	GROUP_DELAY	R/W	0h	These bits select the interpolation filter group delay by configuring the number of samples which are pre-buffered prior to the re-sampler function. 0: 64 Samples (Default) 1: 32 Samples 2: 16 Samples 3: 8 Samples
25-24	DE_EMPHASIS_MODE	R/W	0h	These bits are utilized to enable or disable the digital de-emphasis filter manually. The de-emphasis filter is intended to process 50/15 s pre-emphasized audio material at the following input sampling rates. 0: De-Emphasis Disabled (Default) 1: De-Emphasis Enabled for fS = 48kHz 2: De-Emphasis Enabled for fS = 44.1kHz 3: De-Emphasis Enabled for fS = 32kHz
23-16	ATTENUATION	R/W	0h	These bits are utilized to configure the Group0 SRC digital output attenuation for the stream Output Attenuation (dB) = N 0.5, where N = Attenuation[7:0]DEC.
15-11	RESERVED	R	0h	Always read as 0
10	DIRECT_DOWN_SAMPLE	R/W	0h	This bit selects the mode of the decimation function, either true decimation filter or direct down-sampling without filtering. DDN Decimation Function 0 Decimation Filter (Default) 1 Direct Down Sampling Note: Direct down-sampling should only be used when the output sampling rate is higher than the input sampling rate. When the output sampling rate is equal to or lower than the input sampling rate, the Decimation Filter must be used in order to avoid aliasing.

Table 11-106. ASRC_GSRCCTRL_0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	MUTE	R/W	0h	This bit enables or disables the Group0 SRC output soft mute function. 0: Mute Disabled (Default) 1: Mute enabled; output data set to all zeros.
8	DITHER_ENABLE	R/W	0h	This bit enables or disables the Group0 SRC filter s dithering. 0: Dithering Disabled (Default) 1: Dithering enabled;
7-6	INPUT_WORD_LENGTH	R/W	0h	These bits select the word length for the SRC0 input data. 0: 24 Bits (Default) 1: 20 Bits 2: 18 Bits 3: 16 Bits
5-3	OUTPUT_CLOCK_ZONE_SELECT	R/W	0h	This selects the output clock zone for the Group0 SRCs. It can select between the 4 external audio clock zones. Based on this number rate and stamp is passed to the Stream0 Sample rate converter from respective clock recovery loop. 00: This selects the Clock Recovery loop 0. 01: This selects the Clock Recovery loop 1. 10: This selects the Clock Recovery loop 2. 11: This selects the Clock Recovery loop 3. In this SoC other values are not valid.
2-0	INPUT_CLOCK_ZONE_SELECT	R/W	0h	This selects the input clock zone for the Group0 SRCs. It can select between the 4 external audio clock zones. Based on this number rate and stamp is passed to the Stream0 Sample rate converter from respective clock recovery loop. 000: This selects the Clock Recovery loop 0. 001: This selects the Clock Recovery loop 1. 010: This selects the Clock Recovery loop 2. 011: This selects the Clock Recovery loop 3. In this SoC other values are not valid.

Table 11-107. Register Call Summary for ASRC_GSRCCTRL_0

ASRC Functional Description • Group Mode Configuration Sequence: [0][1][2][3][4][5][6][7][8][9][10]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_GSRCCTRL_0 Register (Offset = 184h + [i * 8h]) [reset = 0h]: [0]

11.1.4.1.33 ASRC_ICKGENSTL_0 Register (Offset = 200h + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_ICKGENSTL_0 is shown in Figure 11-43 and described in Table 11-109.

Table 11-108. ASRC_ICKGENSTL_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-43. ASRC_ICKGENSTL_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STAMP_INT_LO								FRACTIONAL_STAMP							
R/W-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRACTIONAL_STAMP															
R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-109. ASRC_ICKGENSTL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	STAMP_INT_LO	R/W	0h	This is the lower 8 bits of the integer multiple of the next timestamp for the clock source. This register will be updated by adding the Clock Generator Rate to the Stamp value whenever the Free Running Counter passes the Stamp value.
23-0	FRACTIONAL_STAMP	R/W	0h	This is the fractional portion of the next timestamp for the clock source. This register will be updated by adding the Clock Generator Rate to the Stamp value whenever the Free Running Counter passes the Stamp value.

Table 11-110. Register Call Summary for ASRC_ICKGENSTL_0

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_ICKGENSTL_0 Register (Offset = 200h + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.34 ASRC_ICKGENSTH_0 Register (Offset = 204h + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_ICKGENSTH_0 is shown in Figure 11-44 and described in Table 11-112.

Table 11-111. ASRC_ICKGENSTH_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-44. ASRC_ICKGENSTH_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																STAMP_INT_HI															
R-0h																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-112. ASRC_ICKGENSTH_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15-0	STAMP_INT_HI	R/W	0h	This is the upper 16 bits of the integer multiple of the next timestamp for the clock source. This register will be updated by adding the Clock Generator Rate to the Stamp value whenever the Free Running Counter passes the Stamp value.

Table 11-113. Register Call Summary for ASRC_ICKGENSTH_0

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_ICKGENSTH_0 Register (Offset = 204h + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.35 ASRC_ICKGENRTL_0 Register (Offset = 208h + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_ICKGENRTL_0 is shown in Figure 11-45 and described in Table 11-115.

Table 11-114. ASRC_ICKGENRTL_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-45. ASRC_ICKGENRTL_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RATE_INT_LO								FRACTIONAL_STAMP							
R/W-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRACTIONAL_STAMP															
R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-115. ASRC_ICKGENRTL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RATE_INT_LO	R/W	0h	This is the lower 8 bits of the integer multiple of the rate for the clock source. This register will be added with the Clock Generator Stamp to update the Stamp value whenever the Free Running Counter passes the Stamp value.
23-0	FRACTIONAL_STAMP	R/W	0h	This is the fractional portion of the rate for the clock source.

Table 11-116. Register Call Summary for ASRC_ICKGENRTL_0

ASRC Functional Description
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_ICKGENRTL_0 Register (Offset = 208h + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.36 ASRC_ICKGENRTH_0 Register (Offset = 20Ch + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_ICKGENRTH_0 is shown in Figure 11-46 and described in Table 11-118.

Table 11-117. ASRC_ICKGENRTH_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-46. ASRC_ICKGENRTH_0 Register

31	30	29	28	27	26	25	24
INPUT_CLK_GEN_EN	RESERVED						
R/W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RATE_INT_HI							
R/W-0h							
7	6	5	4	3	2	1	0
RATE_INT_HI							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-118. ASRC_ICKGENRTH_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	INPUT_CLK_GEN_EN	R/W	0h	This signal enables the Input Clock Generator 0 1: Enable 0: Disable
30-16	RESERVED	R	0h	Always read as 0
15-0	RATE_INT_HI	R/W	0h	This is the lower 8 bits of the integer multiple of the rate for the clock source. This register will be added with the Clock Generator Stamp to update the Stamp value whenever the Free Running Counter passes the Stamp value.

Table 11-119. Register Call Summary for ASRC_ICKGENRTH_0

ASRC Functional Description
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_ICKGENRTH_0 Register (Offset = 20Ch + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.37 ASRC_ICKLPRTL_0 Register (Offset = 210h + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_ICKLPRTL_0 is shown in Figure 11-47 and described in Table 11-121.

Table 11-120. ASRC_ICKLPRTL_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-47. ASRC_ICKLPRTL_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
_0								FRACTIONAL_STAMP																							
R-0h								R-0h																							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-121. ASRC_ICKLPRTL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	_0	R	0h	This is the lower 8 bits of the integer multiple of the rate for the intput clock recovery loop 0.
23-0	FRACTIONAL_STAMP	R	0h	This is the fractional portion of the rate for the intput clock recovery loop 0

Table 11-122. Register Call Summary for ASRC_ICKLPRTL_0

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_ICKLPRTL_0 Register (Offset = 210h + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.38 ASRC_ICKPRTH_0 Register (Offset = 214h + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_ICKPRTH_0 is shown in Figure 11-48 and described in Table 11-124.

Table 11-123. ASRC_ICKPRTH_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-48. ASRC_ICKPRTH_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RATE_INT_HI															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-124. ASRC_ICKPRTH_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15-0	RATE_INT_HI	R	0h	This is the upper 16 bits of the integer multiple of the rate for the intput clock recovery loop 0.

Table 11-125. Register Call Summary for ASRC_ICKPRTH_0

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_ICKPRTH_0 Register (Offset = 214h + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.39 ASRC_ICKZCNT_0 Register (Offset = 218h + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_ICKZCNT_0 is shown in Figure 11-49 and described in Table 11-127.

Table 11-126. ASRC_ICKZCNT_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-49. ASRC_ICKZCNT_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								INPUT_CLOCK_ZONE_COUNT																									
R-0h								R/W-0h																							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-127. ASRC_ICKZCNT_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Always read as 0
23-0	INPUT_CLOCK_ZONE_COUNT	R/W	0h	This gives the value of free running counter at every posedge of sync signal.

Table 11-128. Register Call Summary for ASRC_ICKZCNT_0

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_ICKZCNT_0 Register (Offset = 218h + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.40 ASRC_ICKZCTRL_0 Register (Offset = 21Ch + [i * 80h]) [reset = 0h]

ASRC_ICKZCTRL_0 is shown in Figure 11-50 and described in Table 11-130.

Table 11-129. ASRC_ICKZCTRL_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-50. ASRC_ICKZCTRL_0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					OVERRIDE_SETTLE_VALUE	OVERRIDE_SETTLE	LOOP_SETUP
R-0h					R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
LOOP_SETUP							SETTLE
R/W-0h							R-0h
7	6	5	4	3	2	1	0
LOOP_STATE				INPUT_CLOCK_ZONE_CLOCK_SOURCE_SELECT			
R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-130. ASRC_ICKZCTRL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Always read as 0
18	OVERRIDE_SETTLE_VALUE	R/W	0h	Value written to this field will be used as settle if override settle signal is set.
17	OVERRIDE_SETTLE	R/W	0h	This signal override the settle of input clock zone 0 clock recovery loop. 1'b1: value written to override settle value will be used. 1'b0: settle from clock recovery will be used.
16-9	LOOP_SETUP	R/W	0h	This is the setting for input clock recovery loop for zone 0 8 h00: Slow loop 8 h40: Medium1 loop 8 h80: Medium2 loop 8 hC0: Slow loop
8	SETTLE	R	0h	This signals sets to 1 when clock recovery loop is settled
7-4	LOOP_STATE	R	0h	Loop debug State
3-0	INPUT_CLOCK_ZONE_CLOCK_SOURCE_SELECT	R/W	0h	This selects the 8x1 and 2x1 mux structure for input clock zone 0. 1xxx: Clock comes from output clock gen 0 0000-0111: Selects rxsync signal from 0-7

Table 11-131. Register Call Summary for ASRC_ICKZCTRL_0

ASRC Functional Description • Stream Mode Configuration Sequence: [0][1][2] • Group Mode Configuration Sequence: [0][1][2] • ASRC Clock Configuration: [0] • ASRC Programming Ranges and Restrictions: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_ICKZCTRL_0 Register (Offset = 21Ch + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.41 ASRC_OCKGENSTL_0 Register (Offset = 220h + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_OCKGENSTL_0 is shown in Figure 11-51 and described in Table 11-133.

Table 11-132. ASRC_OCKGENSTL_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-51. ASRC_OCKGENSTL_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STAMP_INT_LO								FRACTIONAL_STAMP							
R/W-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRACTIONAL_STAMP															
R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-133. ASRC_OCKGENSTL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	STAMP_INT_LO	R/W	0h	This is the lower 8 bits of the integer multiple of the next timestamp for the clock source. This register will be updated by adding the Clock Generator Rate to the Stamp value whenever the Free Running Counter passes the Stamp value.
23-0	FRACTIONAL_STAMP	R/W	0h	This is the fractional portion of the next timestamp for the clock source. This register will be updated by adding the Clock Generator Rate to the Stamp value whenever the Free Running Counter passes the Stamp value.

Table 11-134. Register Call Summary for ASRC_OCKGENSTL_0

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OCKGENSTL_0 Register (Offset = 220h + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.42 ASRC_OCKGENSTH_0 Register (Offset = 224h + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_OCKGENSTH_0 is shown in Figure 11-52 and described in Table 11-136.

Table 11-135. ASRC_OCKGENSTH_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-52. ASRC_OCKGENSTH_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																STAMP_INT_HI															
R-0h																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-136. ASRC_OCKGENSTH_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15-0	STAMP_INT_HI	R/W	0h	This is the upper 16 bits of the integer multiple of the next timestamp for the clock source. This register will be updated by adding the Clock Generator Rate to the Stamp value whenever the Free Running Counter passes the Stamp value.

Table 11-137. Register Call Summary for ASRC_OCKGENSTH_0

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OCKGENSTH_0 Register (Offset = 224h + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.43 ASRC_OCKGENRTL_0 Register (Offset = 228h + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_OCKGENRTL_0 is shown in Figure 11-53 and described in Table 11-139.

Table 11-138. ASRC_OCKGENRTL_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-53. ASRC_OCKGENRTL_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RATE_INT_LO								FRACTIONAL_STAMP							
R/W-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRACTIONAL_STAMP															
R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-139. ASRC_OCKGENRTL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RATE_INT_LO	R/W	0h	This is the lower 8 bits of the integer multiple of the rate for the clock source. This register will be added with the Clock Generator Stamp to update the Stamp value whenever the Free Running Counter passes the Stamp value.
23-0	FRACTIONAL_STAMP	R/W	0h	This is the fractional portion of the rate for the clock source.

Table 11-140. Register Call Summary for ASRC_OCKGENRTL_0

ASRC Functional Description
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OCKGENRTL_0 Register (Offset = 228h + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.44 ASRC_OCKGENRTH_0 Register (Offset = 22Ch + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_OCKGENRTH_0 is shown in Figure 11-54 and described in Table 11-142.

Table 11-141. ASRC_OCKGENRTH_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-54. ASRC_OCKGENRTH_0 Register

31	30	29	28	27	26	25	24
OUTPUT_CLOCK_GEN_EN	RESERVED						
R/W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RATE_INT_HI							
R/W-0h							
7	6	5	4	3	2	1	0
RATE_INT_HI							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-142. ASRC_OCKGENRTH_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	OUTPUT_CLOCK_GEN_EN	R/W	0h	This signal enables the Output Clock Generator 0 1: Enable 0: Disable
30-16	RESERVED	R	0h	Always read as 0
15-0	RATE_INT_HI	R/W	0h	This is the lower 8 bits of the integer multiple of the rate for the clock source. This register will be added with the Clock Generator Stamp to update the Stamp value whenever the Free Running Counter passes the Stamp value.

Table 11-143. Register Call Summary for ASRC_OCKGENRTH_0

ASRC Functional Description
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OCKGENRTH_0 Register (Offset = 22Ch + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.45 ASRC_OCKLPRTL_0 Register (Offset = 230h + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_OCKLPRTL_0 is shown in Figure 11-55 and described in Table 11-145.

Table 11-144. ASRC_OCKLPRTL_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-55. ASRC_OCKLPRTL_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RATE_INT_LO								FRACTIONAL_STAMP							
R-0h								R-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRACTIONAL_STAMP															
R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-145. ASRC_OCKLPRTL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RATE_INT_LO	R	0h	This is the lower 8 bits of the integer multiple of the rate for the outtput clock recovery loop 0.
23-0	FRACTIONAL_STAMP	R	0h	This is the fractional portion of the rate for the outtput clock recovery loop 0.

Table 11-146. Register Call Summary for ASRC_OCKLPRTL_0

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OCKLPRTL_0 Register (Offset = 230h + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.46 ASRC_OCKLPRTH_0 Register (Offset = 234h + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_OCKLPRTH_0 is shown in Figure 11-56 and described in Table 11-148.

Table 11-147. ASRC_OCKLPRTH_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-56. ASRC_OCKLPRTH_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RATE_INT_HI															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-148. ASRC_OCKLPRTH_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15-0	RATE_INT_HI	R	0h	This is the Upper 16 bits of the integer multiple of the rate for the outtput clock recovery loop 0.

Table 11-149. Register Call Summary for ASRC_OCKLPRTH_0

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OCKLPRTH_0 Register (Offset = 234h + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.47 ASRC_OCKLPSTL_0 Register (Offset = 238h + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_OCKLPSTL_0 is shown in Figure 11-57 and described in Table 11-151.

Table 11-150. ASRC_OCKLPSTL_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-57. ASRC_OCKLPSTL_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STAMP_INT_LO								FRACTIONAL_STAMP							
R-0h								R-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRACTIONAL_STAMP															
R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-151. ASRC_OCKLPSTL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	STAMP_INT_LO	R	0h	This is the lower 8 bits of the integer multiple of the next timestamp for the clock source. This register will be updated when output fifo event occurs
23-0	FRACTIONAL_STAMP	R	0h	This is the fractional portion of the next timestamp for the clock source. This register will be updated when output fifo event occurs

Table 11-152. Register Call Summary for ASRC_OCKLPSTL_0

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OCKLPSTL_0 Register (Offset = 238h + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.48 ASRC_OCKLPSTH_0 Register (Offset = 23Ch + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_OCKLPSTH_0 is shown in Figure 11-58 and described in Table 11-154.

Table 11-153. ASRC_OCKLPSTH_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-58. ASRC_OCKLPSTH_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																STAMP_INT_HI															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-154. ASRC_OCKLPSTH_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Always read as 0
15-0	STAMP_INT_HI	R	0h	This is the upper 16 bits of the integer multiple of the next timestamp for the clock recovery loop. This register will be updated when output fifo event occurs

Table 11-155. Register Call Summary for ASRC_OCKLPSTH_0

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OCKLPSTH_0 Register (Offset = 23Ch + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.49 ASRC_OCKZCNT_0 Register (Offset = 240h + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_OCKZCNT_0 is shown in Figure 11-59 and described in Table 11-157.

Table 11-156. ASRC_OCKZCNT_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-59. ASRC_OCKZCNT_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								OUTPUT_CLOCK_ZONE_COUNT																									
R-0h								R/W-0h																							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-157. ASRC_OCKZCNT_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Always read as 0
23-0	OUTPUT_CLOCK_ZONE_COUNT	R/W	0h	This gives the value of free running counter at every posedge of sync signal

Table 11-158. Register Call Summary for ASRC_OCKZCNT_0

ASRC Functional Description • ASRC Clock Configuration: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OCKZCNT_0 Register (Offset = 240h + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.50 ASRC_OCKZCTRL_0 Register (Offset = 244h + [i * 80h]) [reset = 0h]

ASRC_OCKZCTRL_0 is shown in Figure 11-60 and described in Table 11-160.

Table 11-159. ASRC_OCKZCTRL_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-60. ASRC_OCKZCTRL_0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					OVERRIDE_SETTLE_VALUE	OVERRIDE_SETTLE	LOOP_SETUP
R-0h					R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
LOOP_SETUP							SETTLE
R/W-0h							R-0h
7	6	5	4	3	2	1	0
LOOP_STATE				OUTPUT_CLOCK_ZONE_CLOCK_SOURCE_SELECT			
R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-160. ASRC_OCKZCTRL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Always read as 0
18	OVERRIDE_SETTLE_VALUE	R/W	0h	Value written to this field will be used as settle if override settle signal is set.
17	OVERRIDE_SETTLE	R/W	0h	This signal override the settle of output clock zone 0 clock recovery loop. 1'b1: value written to override settle value will be used. 1'b0: settle from clock recovery will be used.
16-9	LOOP_SETUP	R/W	0h	This is the setting for input clock recovery loop for zone 0 8 h00: Slow loop 8 h40: Medium1 loop 8 h80: Medium2 loop 8 hC0: Slow loop
8	SETTLE	R	0h	This signals sets to 1 when clock recovery loop is settled
7-4	LOOP_STATE	R	0h	Loop debug state
3-0	OUTPUT_CLOCK_ZONE_CLOCK_SOURCE_SELECT	R/W	0h	This selects the 8x1 and 2x1 mux structure for output clock zone 0. 1xxx: Clock comes from output clock gen 0 0000-0111: Selects txsync signal from 0-7

Table 11-161. Register Call Summary for ASRC_OCKZCTRL_0

ASRC Functional Description • Group Mode Configuration Sequence: [0][1][2] • Stream Mode Configuration Sequence: [0][1][2] • ASRC Programming Ranges and Restrictions: [0]
ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OCKZCTRL_0 Register (Offset = 244h + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.51 ASRC_ICKLPORTL_0 Register (Offset = 248h + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_ICKLPORTL_0 is shown in Figure 11-61 and described in Table 11-163.

Table 11-162. ASRC_ICKLPORTL_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-61. ASRC_ICKLPORTL_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RATE_INT_LO								FRACTIONAL_STAMP							
R/W-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRACTIONAL_STAMP															
R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-163. ASRC_ICKLPORTL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RATE_INT_LO	R/W	0h	This is the lower 8 bits of the integer multiple of the rate for the override clock recovery loop 0.
23-0	FRACTIONAL_STAMP	R/W	0h	This is the fractional portion of the rate for the override clock recovery loop 0.

Table 11-164. Register Call Summary for ASRC_ICKLPORTL_0

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_ICKLPORTL_0 Register (Offset = 248h + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.52 ASRC_ICKLPORTH_0 Register (Offset = 24Ch + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_ICKLPORTH_0 is shown in Figure 11-62 and described in Table 11-166.

Table 11-165. ASRC_ICKLPORTH_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-62. ASRC_ICKLPORTH_0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							OVERRIDE_RATE
R-0h							R/W-0h
15	14	13	12	11	10	9	8
RATE_INT_HI							
R/W-0h							
7	6	5	4	3	2	1	0
RATE_INT_HI							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-166. ASRC_ICKLPORTH_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Always read as 0
16	OVERRIDE_RATE	R/W	0h	This is the upper 16 bits of the integer multiple of the rate for the override clock recovery loop 0.
15-0	RATE_INT_HI	R/W	0h	This is the upper 16 bits of the integer multiple of the rate for the override clock recovery loop 0.

Table 11-167. Register Call Summary for ASRC_ICKLPORTH_0

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_ICKLPORTH_0 Register (Offset = 24Ch + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.53 ASRC_OCKLPORTL_0 Register (Offset = 250h + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_OCKLPORTL_0 is shown in Figure 11-63 and described in Table 11-169.

Table 11-168. ASRC_OCKLPORTL_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-63. ASRC_OCKLPORTL_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RATE_INT_LO								FRACTIONAL_STAMP							
R/W-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRACTIONAL_STAMP															
R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-169. ASRC_OCKLPORTL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RATE_INT_LO	R/W	0h	This is the lower 8 bits of the integer multiple of the rate for the output override clock recovery loop 0.
23-0	FRACTIONAL_STAMP	R/W	0h	This is the fractional portion of the rate for the output override clock recovery loop 0.

Table 11-170. Register Call Summary for ASRC_OCKLPORTL_0

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OCKLPORTL_0 Register (Offset = 250h + [i * 80h]) [reset = 0h]: [0]

11.1.4.1.54 ASRC_OCKLPORTH_0 Register (Offset = 254h + [i * 80h]) [reset = 0h]

NOTE: The functions controlled by this register are not supported in this family of devices.

ASRC_OCKLPORTH_0 is shown in Figure 11-64 and described in Table 11-172.

Table 11-171. ASRC_OCKLPORTH_0 Instances

Instance	Physical Address
ASRC_0_CFG	021E 0000h

Figure 11-64. ASRC_OCKLPORTH_0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							OVERRIDE_RATE
R-0h							R/W-0h
15	14	13	12	11	10	9	8
RATE_INT_HI							
R/W-0h							
7	6	5	4	3	2	1	0
RATE_INT_HI							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-172. ASRC_OCKLPORTH_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Always read as 0
16	OVERRIDE_RATE	R/W	0h	This is the upper 16 bits of the integer multiple of the rate for the output override clock recovery loop 0.
15-0	RATE_INT_HI	R/W	0h	This is the Upper 16 bits of the integer multiple of the rate for the output override clock recovery loop 0.

Table 11-173. Register Call Summary for ASRC_OCKLPORTH_0

ASRC Registers • ASRC Configuration and Status Registers: [0]
ASRC Configuration and Status Registers • ASRC_OCKLPORTH_0 Register (Offset = 254h + [i * 80h]) [reset = 0h]: [0]

11.1.4.2 ASRC Group Data Registers

Table 11-175 lists the memory-mapped registers for the ASRC Group Data. All register offset addresses not listed in Table 11-175 should be considered as reserved locations and the register contents should not be modified.

Table 11-174. ASRC Group Data Instances

Instance	Physical Address
ASRC_0_S0	2010 4000h

Table 11-175. ASRC Group Data Registers

Offset	Acronym	Register Name	ASRC_0_S0 Physical Address	Section
0h	ASRC_GIFDATAL_0	Group Input FIFO Data L0 Register	2010 4000h	Section 11.1.4.2.1
4h	ASRC_GIFDATAR_0	Group Input FIFO Data R0 Register	2010 4004h	Section 11.1.4.2.2
8h	ASRC_GIFDATAL_1	Group Input FIFO Data L1 Register	2010 4008h	Section 11.1.4.2.1
Ch	ASRC_GIFDATAR_1	Group Input FIFO Data R1 Register	2010 400Ch	Section 11.1.4.2.2
10h	ASRC_GIFDATAL_2	Group Input FIFO Data L2 Register	2010 4010h	Section 11.1.4.2.1
14h	ASRC_GIFDATAR_2	Group Input FIFO Data R2 Register	2010 4014h	Section 11.1.4.2.2
18h	ASRC_GIFDATAL_3	Group Input FIFO Data L3 Register	2010 4018h	Section 11.1.4.2.1
1Ch	ASRC_GIFDATAR_3	Group Input FIFO Data R3 Register	2010 401Ch	Section 11.1.4.2.2
20h	ASRC_GIFDATAL_4	Group Input FIFO Data L4 Register	2010 4020h	Section 11.1.4.2.1
24h	ASRC_GIFDATAR_4	Group Input FIFO Data R4 Register	2010 4024h	Section 11.1.4.2.2
28h	ASRC_GIFDATAL_5	Group Input FIFO Data L5 Register	2010 4028h	Section 11.1.4.2.1
2Ch	ASRC_GIFDATAR_5	Group Input FIFO Data R5 Register	2010 402Ch	Section 11.1.4.2.2
30h	ASRC_GIFDATAL_6	Group Input FIFO Data L6 Register	2010 4030h	Section 11.1.4.2.1
34h	ASRC_GIFDATAR_6	Group Input FIFO Data R6 Register	2010 4034h	Section 11.1.4.2.2
38h	ASRC_GIFDATAL_7	Group Input FIFO Data L7 Register	2010 4038h	Section 11.1.4.2.1
3Ch	ASRC_GIFDATAR_7	Group Input FIFO Data R7 Register	2010 403Ch	Section 11.1.4.2.2
80h	ASRC_GOFDATAL_0	Group Output FIFO Data L0 Register	2010 4080h	Section 11.1.4.2.3
84h	ASRC_GOFDATAR_0	Group Output FIFO Data R0 Register	2010 4084h	Section 11.1.4.2.4
88h	ASRC_GOFDATAL_1	Group Output FIFO Data L1 Register	2010 4088h	Section 11.1.4.2.3
8Ch	ASRC_GOFDATAR_1	Group Output FIFO Data R1 Register	2010 408Ch	Section 11.1.4.2.4
90h	ASRC_GOFDATAL_2	Group Output FIFO Data L2 Register	2010 4090h	Section 11.1.4.2.3
94h	ASRC_GOFDATAR_2	Group Output FIFO Data R2 Register	2010 4094h	Section 11.1.4.2.4

Table 11-175. ASRC Group Data Registers (continued)

Offset	Acronym	Register Name	ASRC_0_S0 Physical Address	Section
98h	ASRC_GOFDATAL_3	Group Output FIFO Data L3 Register	2010 4098h	Section 11.1.4.2.3
9Ch	ASRC_GOFDATAR_3	Group Output FIFO Data R3 Register	2010 409Ch	Section 11.1.4.2.4
A0h	ASRC_GOFDATAL_4	Group Output FIFO Data L4 Register	2010 40A0h	Section 11.1.4.2.3
A4h	ASRC_GOFDATAR_4	Group Output FIFO Data R4 Register	2010 40A4h	Section 11.1.4.2.4
A8h	ASRC_GOFDATAL_5	Group Output FIFO Data L5 Register	2010 40A8h	Section 11.1.4.2.3
ACh	ASRC_GOFDATAR_5	Group Output FIFO Data R5 Register	2010 40ACh	Section 11.1.4.2.4
B0h	ASRC_GOFDATAL_6	Group Output FIFO Data L6 Register	2010 40B0h	Section 11.1.4.2.3
B4h	ASRC_GOFDATAR_6	Group Output FIFO Data R6 Register	2010 40B4h	Section 11.1.4.2.4
B8h	ASRC_GOFDATAL_7	Group Output FIFO Data L7 Register	2010 40B8h	Section 11.1.4.2.3
BCh	ASRC_GOFDATAR_7	Group Output FIFO Data R7 Register	2010 40BCh	Section 11.1.4.2.4

11.1.4.2.1 ASRC_GIFDATAL_0 Register (Offset = 0h + [i * 8h]) [reset = 0h]

ASRC_GIFDATAL_0 is shown in Figure 11-65 and described in Table 11-177.

Table 11-176. ASRC_GIFDATAL_0 Instances

Instance	Physical Address
ASRC_0_S0	2010 4000h

Figure 11-65. ASRC_GIFDATAL_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IN_DATA0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-177. ASRC_GIFDATAL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IN_DATA0	R/W	0h	This is where the input samples are written for stream0 left channel

Table 11-178. Register Call Summary for ASRC_GIFDATAL_0

ASRC Group Data Registers • ASRC_GIFDATAL_0 Register (Offset = 0h + [i * 8h]) [reset = 0h]: [0]
ASRC Registers • ASRC Group Data Registers: [0]

11.1.4.2.2 ASRC_GIFDATAR_0 Register (Offset = 4h + [i * 8h]) [reset = 0h]

ASRC_GIFDATAR_0 is shown in Figure 11-66 and described in Table 11-180.

Table 11-179. ASRC_GIFDATAR_0 Instances

Instance	Physical Address
ASRC_0_S0	2010 4000h

Figure 11-66. ASRC_GIFDATAR_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IN_DATA1																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-180. ASRC_GIFDATAR_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IN_DATA1	R/W	0h	This is where the input samples are written for stream0 right channel

Table 11-181. Register Call Summary for ASRC_GIFDATAR_0

ASRC Group Data Registers • ASRC_GIFDATAR_0 Register (Offset = 4h + [i * 8h]) [reset = 0h]: [0]
ASRC Registers • ASRC Group Data Registers: [0]

11.1.4.2.3 ASRC_GOFDATAL_0 Register (Offset = 80h + [i * 8h]) [reset = 0h]

ASRC_GOFDATAL_0 is shown in Figure 11-67 and described in Table 11-183.

Table 11-182. ASRC_GOFDATAL_0 Instances

Instance	Physical Address
ASRC_0_S0	2010 4000h

Figure 11-67. ASRC_GOFDATAL_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUT_DATA0																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-183. ASRC_GOFDATAL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	OUT_DATA0	R	0h	This is from where the output samples are read for stream0 left channel

Table 11-184. Register Call Summary for ASRC_GOFDATAL_0

ASRC Group Data Registers • ASRC_GOFDATAL_0 Register (Offset = 80h + [i * 8h]) [reset = 0h]: [0]
ASRC Functional Description • Group Mode Configuration Sequence: [0]
ASRC Registers • ASRC Group Data Registers: [0]

11.1.4.2.4 ASRC_GOFDATAR_0 Register (Offset = 84h + [i * 8h]) [reset = 0h]

ASRC_GOFDATAR_0 is shown in Figure 11-68 and described in Table 11-186.

Table 11-185. ASRC_GOFDATAR_0 Instances

Instance	Physical Address
ASRC_0_S0	2010 4000h

Figure 11-68. ASRC_GOFDATAR_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUT_DATA1																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-186. ASRC_GOFDATAR_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	OUT_DATA1	R	0h	This is from where the output samples are read for stream0 right channel

Table 11-187. Register Call Summary for ASRC_GOFDATAR_0

ASRC Group Data Registers • ASRC_GOFDATAR_0 Register (Offset = 84h + [i * 8h]) [reset = 0h]: [0]
ASRC Functional Description • Group Mode Configuration Sequence: [0]
ASRC Registers • ASRC Group Data Registers: [0]

11.1.4.3 ASRC Stream Data Registers

Table 11-189 lists the memory-mapped registers for the ASRC Stream Data. All register offset addresses not listed in Table 11-189 should be considered as reserved locations and the register contents should not be modified.

Table 11-188. ASRC Stream Data Instances

Instance	Physical Address
ASRC_0_S1	2010 0000h

Table 11-189. ASRC Stream Data Registers

Offset	Acronym	Register Name	ASRC_0_S1 Physical Address	Section
0h	ASRC_SIFDATAL_0	Stream Input FIFO Data L0 Register	2010 0000h	Section 11.1.4.3.1
100h	ASRC_SIFDATAR_0	Stream Input FIFO Data R0 Register	2010 0100h	Section 11.1.4.3.2
200h	ASRC_SIFDATAL_1	Stream Input FIFO Data L1 Register	2010 0200h	Section 11.1.4.3.1
300h	ASRC_SIFDATAR_1	Stream Input FIFO Data R1 Register	2010 0300h	Section 11.1.4.3.2
400h	ASRC_SIFDATAL_2	Stream Input FIFO Data L2 Register	2010 0400h	Section 11.1.4.3.1
500h	ASRC_SIFDATAR_2	Stream Input FIFO Data R2 Register	2010 0500h	Section 11.1.4.3.2
600h	ASRC_SIFDATAL_3	Stream Input FIFO Data L3 Register	2010 0600h	Section 11.1.4.3.1
700h	ASRC_SIFDATAR_3	Stream Input FIFO Data R3 Register	2010 0700h	Section 11.1.4.3.2
800h	ASRC_SIFDATAL_4	Stream Input FIFO Data L4 Register	2010 0800h	Section 11.1.4.3.1
900h	ASRC_SIFDATAR_4	Stream Input FIFO Data R4 Register	2010 0900h	Section 11.1.4.3.2
A00h	ASRC_SIFDATAL_5	Stream Input FIFO Data L5 Register	2010 0A00h	Section 11.1.4.3.1
B00h	ASRC_SIFDATAR_5	Stream Input FIFO Data R5 Register	2010 0B00h	Section 11.1.4.3.2
C00h	ASRC_SIFDATAL_6	Stream Input FIFO Data L6 Register	2010 0C00h	Section 11.1.4.3.1
D00h	ASRC_SIFDATAR_6	Stream Input FIFO Data R6 Register	2010 0D00h	Section 11.1.4.3.2
E00h	ASRC_SIFDATAL_7	Stream Input FIFO Data L7 Register	2010 0E00h	Section 11.1.4.3.1
F00h	ASRC_SIFDATAR_7	Stream Input FIFO Data R7 Register	2010 0F00h	Section 11.1.4.3.2
1000h	ASRC_SOFDATAL_0	Stream Output FIFO Data L0 Register	2010 1000h	Section 11.1.4.3.3
1100h	ASRC_SOFDATAR_0	Stream Output FIFO Data R0 Register	2010 1100h	Section 11.1.4.3.4
1200h	ASRC_SOFDATAL_1	Stream Output FIFO Data L1 Register	2010 1200h	Section 11.1.4.3.3
1300h	ASRC_SOFDATAR_1	Stream Output FIFO Data R1 Register	2010 1300h	Section 11.1.4.3.4
1400h	ASRC_SOFDATAL_2	Stream Output FIFO Data L2 Register	2010 1400h	Section 11.1.4.3.3
1500h	ASRC_SOFDATAR_2	Stream Output FIFO Data R2 Register	2010 1500h	Section 11.1.4.3.4

Table 11-189. ASRC Stream Data Registers (continued)

Offset	Acronym	Register Name	ASRC_0_S1 Physical Address	Section
1600h	ASRC_SOFDATAL_3	Stream Output FIFO Data L3 Register	2010 1600h	Section 11.1.4.3.3
1700h	ASRC_SOFDATAR_3	Stream Output FIFO Data R3 Register	2010 1700h	Section 11.1.4.3.4
1800h	ASRC_SOFDATAL_4	Stream Output FIFO Data L4 Register	2010 1800h	Section 11.1.4.3.3
1900h	ASRC_SOFDATAR_4	Stream Output FIFO Data R4 Register	2010 1900h	Section 11.1.4.3.4
1A00h	ASRC_SOFDATAL_5	Stream Output FIFO Data L5 Register	2010 1A00h	Section 11.1.4.3.3
1B00h	ASRC_SOFDATAR_5	Stream Output FIFO Data R5 Register	2010 1B00h	Section 11.1.4.3.4
1C00h	ASRC_SOFDATAL_6	Stream Output FIFO Data L6 Register	2010 1C00h	Section 11.1.4.3.3
1D00h	ASRC_SOFDATAR_6	Stream Output FIFO Data R6 Register	2010 1D00h	Section 11.1.4.3.4
1E00h	ASRC_SOFDATAL_7	Stream Output FIFO Data L7 Register	2010 1E00h	Section 11.1.4.3.3
1F00h	ASRC_SOFDATAR_7	Stream Output FIFO Data R7 Register	2010 1F00h	Section 11.1.4.3.4

11.1.4.3.1 ASRC_SIFDATAL_0 Register (Offset = 0h + [i * 200h]) [reset = 0h]

ASRC_SIFDATAL_0 is shown in Figure 11-69 and described in Table 11-191.

Table 11-190. ASRC_SIFDATAL_0 Instances

Instance	Physical Address
ASRC_0_S1	2010 0000h

Figure 11-69. ASRC_SIFDATAL_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IN_DATA0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-191. ASRC_SIFDATAL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IN_DATA0	R/W	0h	This is where the input samples are written for stream0 left channel

Table 11-192. Register Call Summary for ASRC_SIFDATAL_0

ASRC Registers • ASRC Stream Data Registers: [0]
ASRC Stream Data Registers • ASRC_SIFDATAL_0 Register (Offset = 0h + [i * 200h]) [reset = 0h]: [0]

11.1.4.3.2 ASRC_SIFDATAR_0 Register (Offset = 100h + [i * 200h]) [reset = 0h]

ASRC_SIFDATAR_0 is shown in Figure 11-70 and described in Table 11-194.

Table 11-193. ASRC_SIFDATAR_0 Instances

Instance	Physical Address
ASRC_0_S1	2010 0000h

Figure 11-70. ASRC_SIFDATAR_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IN_DATA1																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-194. ASRC_SIFDATAR_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IN_DATA1	R/W	0h	This is where the input samples are written for stream0 right channel

Table 11-195. Register Call Summary for ASRC_SIFDATAR_0

ASRC Registers • ASRC Stream Data Registers: [0]
ASRC Stream Data Registers • ASRC_SIFDATAR_0 Register (Offset = 100h + [i * 200h]) [reset = 0h]: [0]

11.1.4.3.3 ASRC_SOFDATAL_0 Register (Offset = 1000h + [i * 200h]) [reset = 0h]

ASRC_SOFDATAL_0 is shown in Figure 11-71 and described in Table 11-197.

Table 11-196. ASRC_SOFDATAL_0 Instances

Instance	Physical Address
ASRC_0_S1	2010 0000h

Figure 11-71. ASRC_SOFDATAL_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUT_DATA0																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-197. ASRC_SOFDATAL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	OUT_DATA0	R	0h	This is from where the output samples are read for stream0 left channel

Table 11-198. Register Call Summary for ASRC_SOFDATAL_0

ASRC Registers • ASRC Stream Data Registers: [0]
ASRC Stream Data Registers • ASRC_SOFDATAL_0 Register (Offset = 1000h + [i * 200h]) [reset = 0h]: [0]

11.1.4.3.4 ASRC_SOFDATAR_0 Register (Offset = 1100h + [i * 200h]) [reset = 0h]

ASRC_SOFDATAR_0 is shown in Figure 11-72 and described in Table 11-200.

Table 11-199. ASRC_SOFDATAR_0 Instances

Instance	Physical Address
ASRC_0_S1	2010 0000h

Figure 11-72. ASRC_SOFDATAR_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUT_DATA1																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-200. ASRC_SOFDATAR_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	OUT_DATA1	R	0h	This is from where the output samples are read for stream0 right channel

Table 11-201. Register Call Summary for ASRC_SOFDATAR_0

ASRC Registers • ASRC Stream Data Registers: [0]
ASRC Stream Data Registers • ASRC_SOFDATAR_0 Register (Offset = 1100h + [i * 200h]) [reset = 0h]: [0]

11.2 Controller Area Network Interface (DCAN)

This section describes the Controller Area Network (DCAN) module in the device.

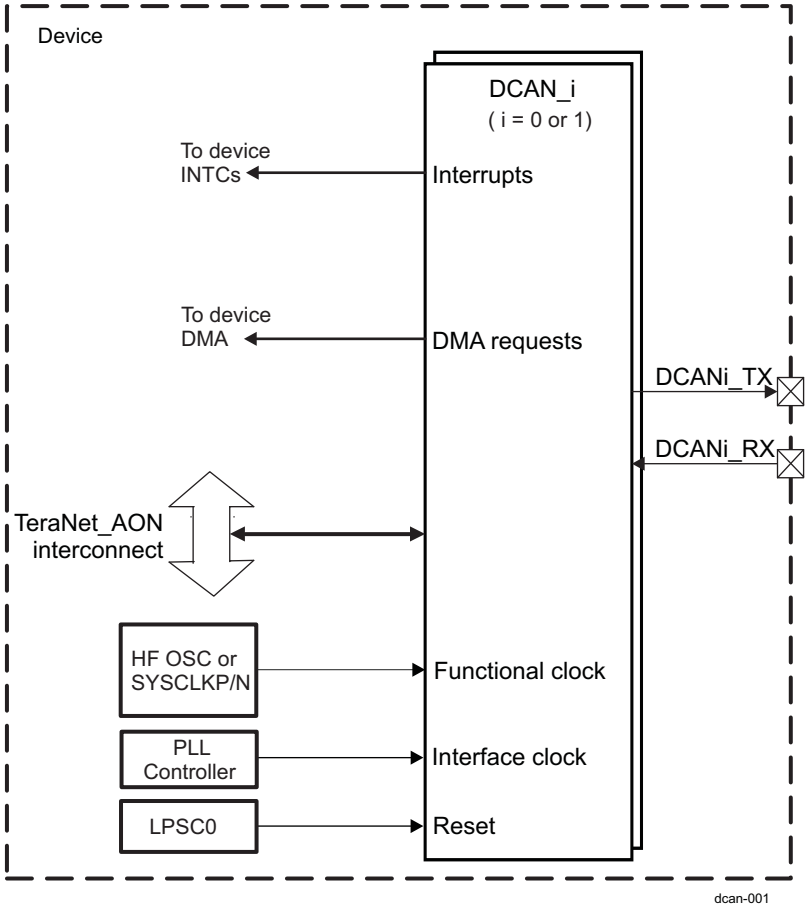
11.2.1 DCAN Overview

Controller Area Network (CAN) is a serial communications protocol which efficiently supports distributed real-time applications. CAN has high immunity to electrical interference and the ability to self-diagnose and repair data errors. In a CAN network, many short messages are broadcast to the entire network, which provides for data consistency in every node of the system.

The device supports two DCAN modules, referred to as DCAN_0 and DCAN_1, connecting to the CAN network through external (for the device) transceivers. The DCAN modules support bit rates up to 1 Mbit/s and are compliant to the CAN 2.0B protocol specification

Figure 11-73 shows the DCAN module highlights.

Figure 11-73. DCAN Overview



11.2.1.1 Features

The DCAN module implements the following features:

- Support for CAN protocol version 2.0 part A, B
- Bit rates up to 1 Mbit/s
- 64 message objects in a dedicated message RAM
- Individual identifier mask for each message object
- Programmable FIFO mode for message objects
- Programmable loop-back modes for self-test operation

- Software module reset
- Suspend mode for debug support
- Automatic bus on after Bus-Off state by a programmable 32-bit timer
- Message RAM single error correction and double error detection mechanism (SECDED)
- Direct access to message RAM during test mode.
- Support for three interrupt lines: Level 0 and Level 1, and a separate ECC interrupt line
- Local power down and wakeup support
- Automatic message RAM initialization
- Support for DMA access

11.2.2 DCAN Environment

CAN network physical layer consists of two-wire differential bus, usually twisted pair, and provides high level of interference immunity. External CAN transceiver IC is needed to access a CAN bus by the DCAN.

Figure 11-74 shows an overview of a typical DCAN application.

Figure 11-74. DCAN Typical Application

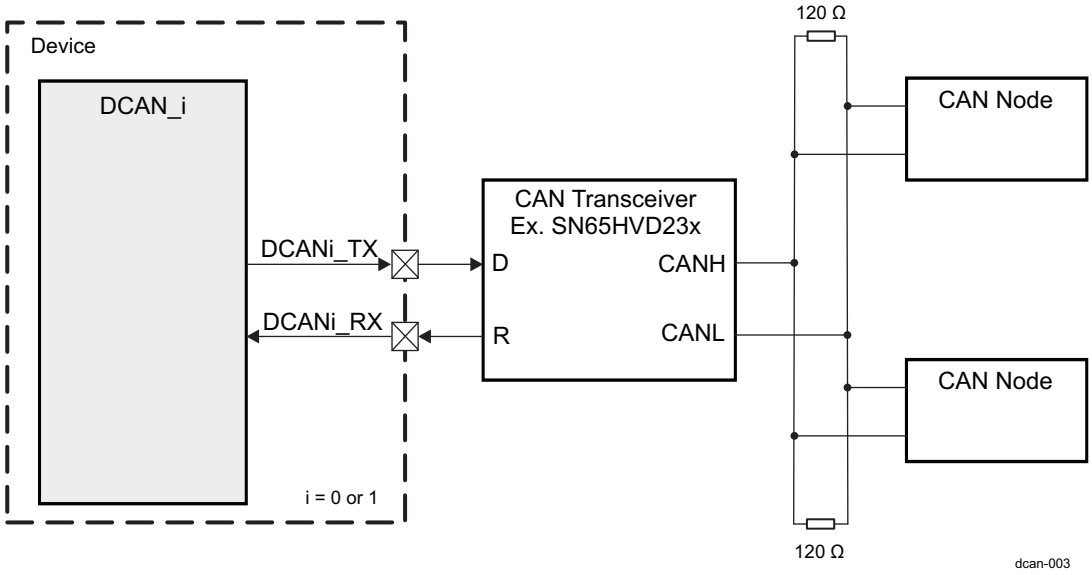


Table 11-202 describes the external signals of the DCAN module.

Table 11-202. DCAN I/O Description

Module Signal	Device Signal	I/O[1]	Description	Value at Reset
CAN_RX	DCANi_RX[2]	I	Serial data input from external CAN tranceiver	HiZ
CAN_TX	DCANi_TX[2]	O	Serial data output to external CAN tranceiver	1

(1) I = Input; O = Output
(2) i = 0 to 1

NOTE: The path from a module pin to device pad(s) is defined at the device I/O logic level. The BOOT_CFG registers assign the specific function to the device pads. For more information on BOOT_CFG settings, see Section 5.1.3.1.1, *Pad Configuration Registers* in Section 5.1, *Control Module (BOOT_CFG)*.

11.2.2.1 CAN Network Basics

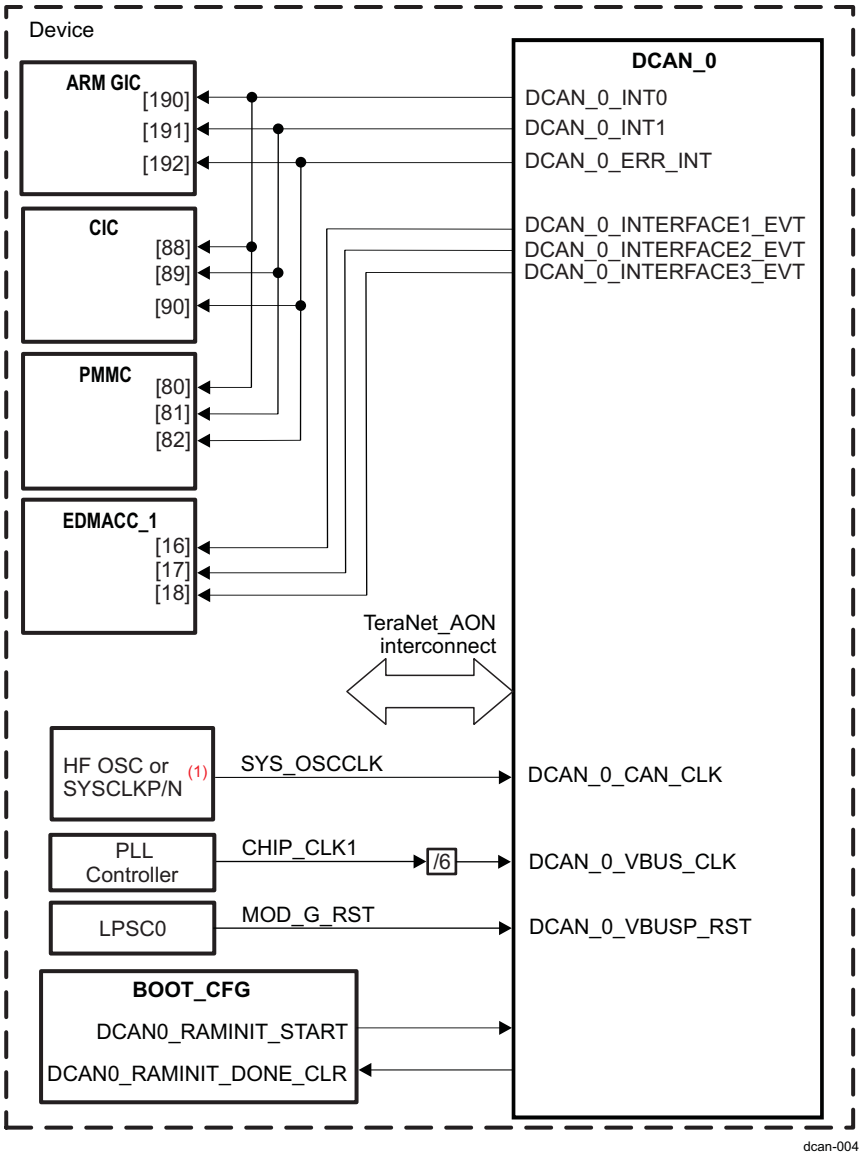
- CAN bus is a 2-wire differential bus using NRZ encoding and has two states:
  - Recessive state (logic 0)
  - Dominant state (logic 1)
- The network is multimaster. When two or more nodes attempt to transmit at the same time, a non-destructive arbitration technique guarantees messages are sent in order of priority and no messages are lost
- The message transmission is multicast. Data messages transmitted are identifier based, not address based
- Content of message is labeled by the identifier that is unique throughout the network (for example, rpm, temperature, position, pressure, and so forth).

- All nodes on network receive the message and each performs an acceptance test on the identifier. If message is relevant, it is processed, otherwise it is ignored
- The unique identifier also determines the priority of the message (the lower the numerical value of the identifier, the higher the priority is)
- Data is transmitted and received using message frames, consisting of:
  - Arbitration field
  - Control field
  - Data field (0 ÷ 8 bytes)
  - CRC field
  - ACK field.

11.2.3 DCAN Integration

Figure 11-75 shows the integration of the DCAN_0 module in the device.

Figure 11-75. DCAN_0 Integration



(1) For more information see Section 5.4, *Clock Management*.

Figure 11-76 shows the integration of the DCAN_1 module in the device.

Figure 11-76. DCAN_1 Integration

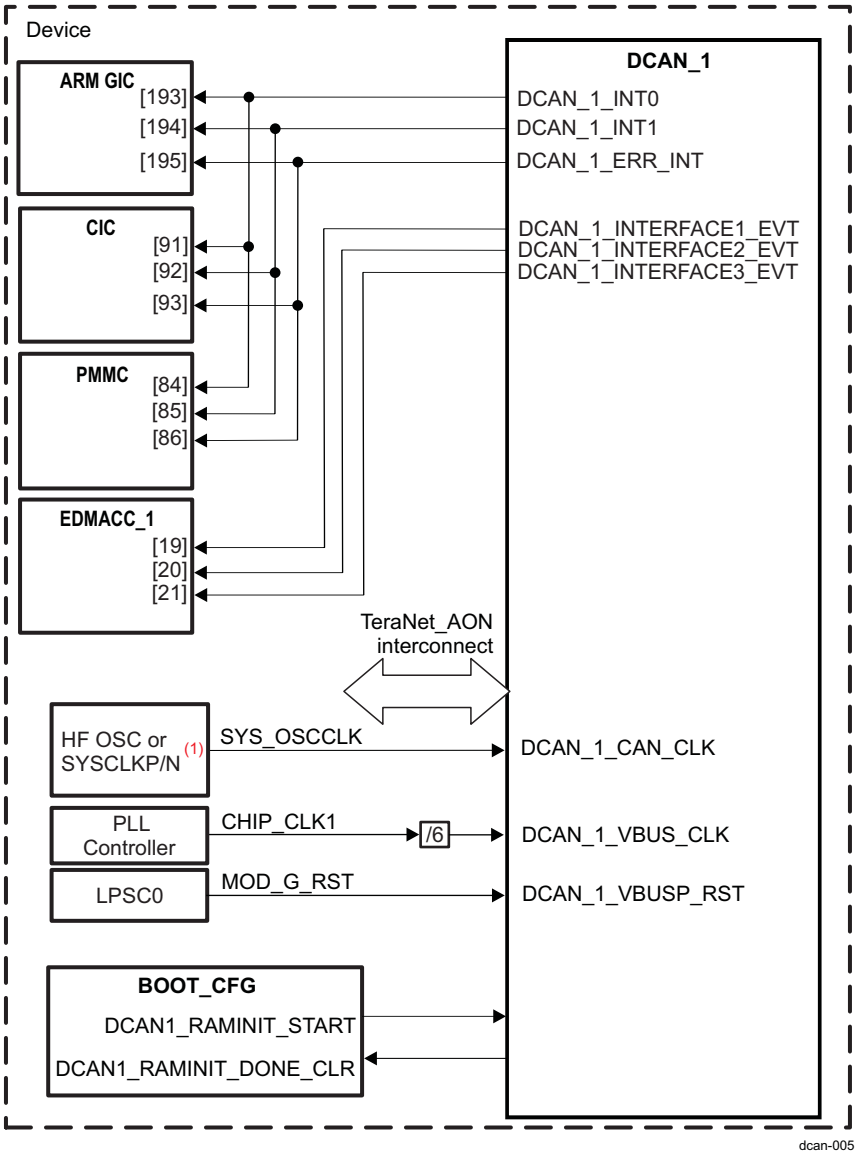


Table 11-3553 through summarize the integration of the DCAN modules in the device.

Table 11-203. DCAN Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
DCAN_0	PD0	LPSC0	TeraNet_AON
DCAN_1	PD0	LPSC0	TeraNet_AON

Table 11-204. DCAN Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
DCAN_0	DCAN_0_VBUS_CLK	CHIP_CLK1 / 6	PLL Controller	DCAN_0 moduleInterface clock.
	DCAN_0_CAN_CLK	SYS_OSCCLK	HF Oscillator or external SYSCLKP/N input pin	DCAN_0 core (CAN_CLK) Functional clock.

Table 11-204. DCAN Clocks and Resets (continued)

DCAN_1	DCAN_1_VBUS_CLK	CHIP_CLK1 / 6	PLL Controller	DCAN_1 module Interface clock.
	DCAN_1_CAN_CLK	SYS_OSCCLK	HF Oscillator or external SYSCLKP/N input pin	DCAN_1 core (CAN_CLK) Functional clock.
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
DCAN_0	DCAN_0_VBUSP_RST	MOD_G_RST	LPSC0	Global asynchronous reset signal to the DCAN_0 module.
DCAN_1	DCAN_1_VBUSP_RST	MOD_G_RST	LPSC0	Global asynchronous reset signal to the DCAN_1 module.

Table 11-205. DCAN Hardware Requests

Interrupt Requests					
Module Instance	**Event Name**	**Mapped To Input Event [Number]**			**Description**
		ARM GIC	**CIC**	**PMMC**	
DCAN_0	DCAN_0_INT0	[190]	[88]	[80]	Error, Status, and Message Objects interrupt
	DCAN_0_INT1	[191]	[89]	[81]	Message Objects interrupt
	DCAN_0_ERR_INT	[192]	[90]	[82]	ECC double bit or single bit error (when correction is disabled) interrupt
DCAN_1	DCAN_1_INT0	[193]	[91]	[84]	Error, Status, and Message Objects interrupt
	DCAN_1_INT1	[194]	[92]	[85]	Message Objects interrupt
	DCAN_1_ERR_INT	[195]	[93]	[86]	ECC double bit or single bit error (when correction is disabled) interrupt
DMA Requests					
Module Instance	**Event Name**	**Mapped To Input Event [Number]**			**Description**
		EDMACC_0		**EDMACC_1**	
DCAN_0	DCAN_0_INTERFACE1_EVT	–		[16]	DMA request for IF1 interface register
	DCAN_0_INTERFACE2_EVT	–		[17]	DMA request for IF2 interface register
	DCAN_0_INTERFACE3_EVT	–		[18]	DMA request for IF3 interface register
DCAN_1	DCAN_1_INTERFACE1_EVT	–		[19]	DMA request for IF1 interface register
	DCAN_1_INTERFACE2_EVT	–		[20]	DMA request for IF2 interface register
	DCAN_1_INTERFACE3_EVT	–		[21]	DMA request for IF3 interface register

NOTE: For the description of the interrupt source, see Section 11.2.4.2, *Interrupt Functionality*.

11.2.4 DCAN Functional Description

The DCAN module performs CAN protocol communication according to ISO 11898-1. The bit rate can be programmed to values up to 1 Mbit/s. Additional transceiver hardware is required for the connection to the physical layer (CAN bus).

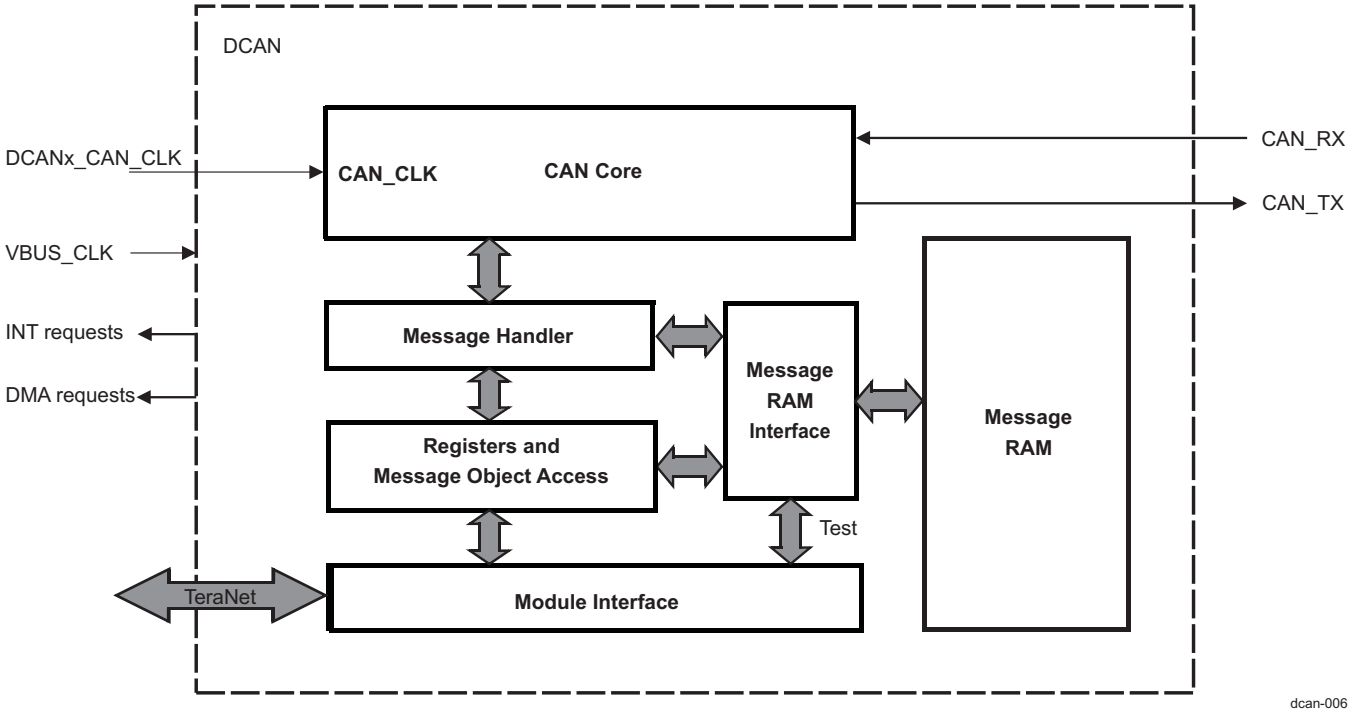
For communication on a CAN network, individual message objects can be configured. The message objects and identifier masks are stored in the message RAM.

All functions concerning the handling of messages are implemented in the message handler. Those functions are acceptance filtering, the transfer of messages between the CAN core and the message RAM, and the handling of transmission requests, as well as the generation of interrupts or DMA requests.

The register set of the DCAN module can be accessed directly via the module interface. These registers are used to control and configure the CAN core and the message handler, and to access the message RAM.

Figure 11-77 shows the DCAN block diagram and its features are described below.

Figure 11-77. DCAN Block Diagram



CAN Core: The CAN core consists of the CAN protocol controller and the Rx/Tx shift register. It handles all ISO 11898-1 protocol functions.

Message Handler: The message handler is a state machine that controls the data transfer between the single-ported message RAM and the CAN core's Rx/Tx shift register. It also handles acceptance filtering and the interrupt/DMA request generation as programmed in the control registers.

Message RAM: The DCAN enables a storage of 64 CAN messages.

Message RAM Interface: Three interface register sets control the MPU read and write accesses to the message RAM. There are two interface registers sets for read and write access, IF1 and IF2, and one interface register set for read access only, IF3. Additional information can be found in Section 11.2.4.8.12, *Reading From a FIFO Buffer*.

The interface registers have the same word-length as the message RAM.

Registers and Message Object Access: Data consistency is ensured by indirect accesses to the message objects. During normal operation, all software and DMA accesses to the message RAM are done through interface registers. In a dedicated test mode, the message RAM is memory mapped and can be directly accessed by either MPU or DMA.

Module Interface: The DCAN module registers are accessed by the user software through a 32-bit peripheral bus interface.

Clocking: Two clocks are provided to the DCAN module: the peripheral synchronous clock (interface clock [VBUS_CLK]) and the peripheral asynchronous clock (functional clock [CAN_CLK]).

11.2.4.1 Module Clocking Requirements

Two clocks are provided to the DCAN module:

- the peripheral synchronous clock (VBUS_CLK) as the general module clock source
- and the peripheral asynchronous clock (CAN_CLK) provided to the CAN core for generating the CAN bit timing.

NOTE: VBUS_CLK must always be higher or equal to CAN_CLK, in order to achieve a stable functionality of the DCAN. Here, also the frequency shift of the modulated VBUS_CLK has to be considered:

$f_{0,\ VBUS\_CLK} \pm \Delta f_{FM,\ VBUS\_CLK} \geq f_{CAN\_CLK}$

11.2.4.2 Interrupt Functionality

Interrupts can be generated on two interrupt lines: INT0 and INT1. These lines can be enabled by setting the DCAN_CTL[1] IE0 and [17] IE1 bits, respectively. The interrupts are level triggered at the chip level.

DCAN provides three groups of interrupt sources: message object interrupts, status change interrupts, and error interrupts (see Figure 11-78, *Error and Status Change Interrupts* and Figure 11-79, *Message Objects Interrupts*).

The source of an interrupt can be determined by the interrupt identifiers DCAN_INT[15-0] INT0ID or bit field [23-16] INT1ID. When no interrupt is pending, the register will hold the value zero.

Each interrupt line remains active until the dedicated field in the interrupt register DCAN_INT[15-0] INT0ID or bit field [23-16] INT1ID again reaches zero (this means the cause of the interrupt is reset), or until IE0/IE1 are reset.

The value 0x8000 in the INT0ID field indicates that an interrupt is pending because the CAN core has updated (not necessarily changed) the Error and Status register (DCAN_ES). This interrupt has the highest priority. The software can update (reset) the status bits [9] WAKEUPPND, [4] RXOK, [3] TXOK and [2-0] LEC by reading DCAN_ES, but a write access of the software will never generate or reset an interrupt.

Values between 1 and the number of the last message object indicates that the source of the interrupt is one of the message objects, INT0ID, respectively INT1ID will point to the pending message interrupt with the highest priority. The Message Object 1 has the highest priority; the last message object has the lowest priority.

An interrupt service routine that reads the message that is the source of the interrupt may read the message and reset the message object's IntPnd at the same time (DCAN_IF1CMD/DCAN_IF2CMD[19] CLRINTPND bit). When IntPnd is cleared, DCAN_INT will point to the next message object with a pending interrupt.

11.2.4.2.1 Message Object Interrupts

Message object interrupts are generated by events from the message objects. They are controlled by the flags IntPnd, TxIE and RxIE that are described in Section 11.2.4.11.1, *Structure of Message Objects*.

Message object interrupts can be routed to either INT0 or INT1 line, controlled by the interrupt multiplexer registers (DCAN_INTMUX12 to DCAN_INTMUX78).

11.2.4.2.2 Status Change Interrupts

The events WAKEUPPND, RXOK, TXOK and LEC in the error and status register (DCAN_ES) belong to the status change interrupts. The status change interrupt group can be enabled by DCAN_CTL[2] SIE bit.

If SIE is set, a status change interrupt will be generated at each CAN frame, independent of bus errors or valid CAN communication, and also independent of the message RAM configuration.

Status change interrupts can only be routed to interrupt line INT0, which has to be enabled by setting DCAN_CTL[1] IE0 = 1.

NOTE: Reading DCAN_ES will clear the WAKEUPPND flag. If in global power-down mode, the WAKEUPPND flag is cleared by such a read access before the DCAN module has been waken up by the system, the DCAN may re-assert the WAKEUPPND flag, and a second interrupt may occur.

11.2.4.2.3 Error Interrupts

The events PER, BOFF and EWARN, monitored in the DCAN_ES register belong to the error interrupts. The error interrupt group can be enabled by setting bit DCAN_CTL[3] EIE = 1.

Error interrupts can only be routed to interrupt line INT0, which has to be enabled by setting DCAN_CTL[1] IE0 = 1.

Figure 11-78. Error and Status Change Interrupts

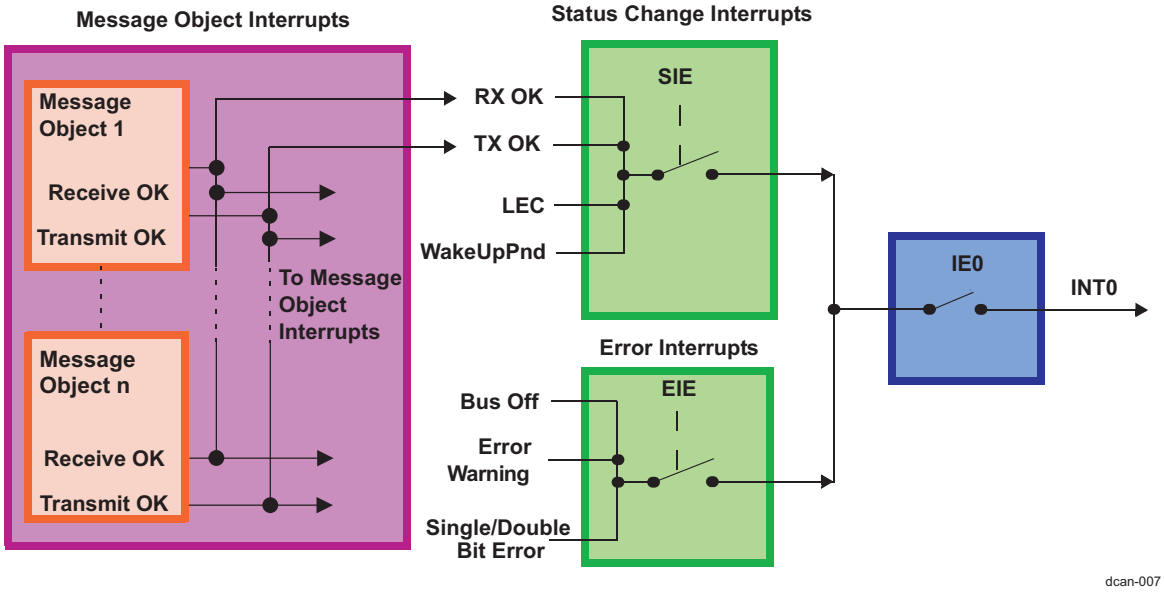
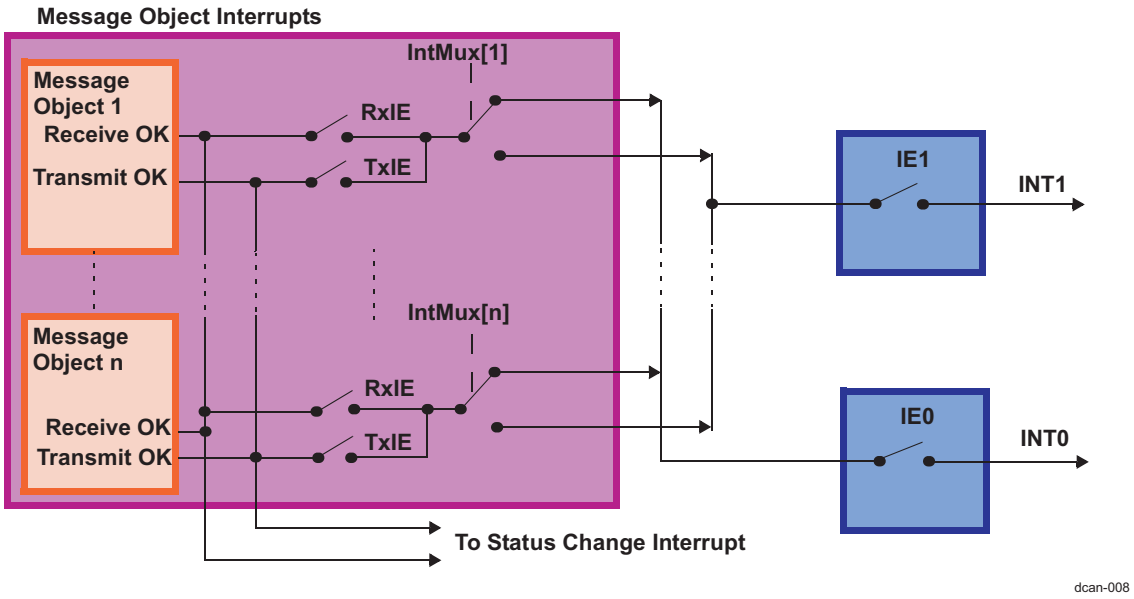


Figure 11-79. Message Objects Interrupts



11.2.4.3 DMA Functionality

DCAN provides three DMA request lines, each indicating new data in one of the three interface register sets IF1, IF2 and IF3.

The update of IF1 and IF2 registers will be initiated by a write access to the IF1 respective IF2 Command Registers (DCAN_IF1CMD, DCAN_IF2CMD).

The IF3 registers content can be automatically updated on reception of CAN messages in message objects which are programmed for automatic IF3 update, see Section 11.2.4.10.2, *IF3 Register Set*.

When a DCAN internal IFx (x = 1 to 3) update is complete, a DMA request will be activated and stays active until the first access to one of the relevant IFx registers. The DMA functionality has to be enabled by setting bit [18] DE0/[19] DE1/[20] DE3 in DCAN_CTL.

11.2.4.4 Local Power-Down Mode

DCAN supports a local power-down mode, which can be controlled within the DCAN registers.

11.2.4.4.1 Entering Local Power-Down Mode

The local power-down mode is requested by setting the DCAN_CTL[24] PDR bit (=1).

DCAN then finishes all transmit requests of the message objects. When all requests are done, DCAN waits until a bus idle state is recognized. Then it will automatically set the DCAN_CTL[0] INIT bit to prevent any further CAN transfers, and it will also set the DCAN_ES[10] PDA bit. With setting the PDA bit, the DCAN module indicates that the local power-down mode has been entered.

During local power-down mode, the internal clocks of the DCAN module are turned off, but there is a wakeup logic (see Section 11.2.4.4.2, *Wakeup From Local Power Down*) that can be active, if enabled. Also, the actual contents of the control registers can be read back.

NOTE: In local low-power mode, the software must not clear the INIT bit while PDR is set. If there are any messages in the message RAM which are configured as transmit messages and the application resets the INIT bit, these messages may get sent.

11.2.4.4.2 Wakeup From Local Power Down

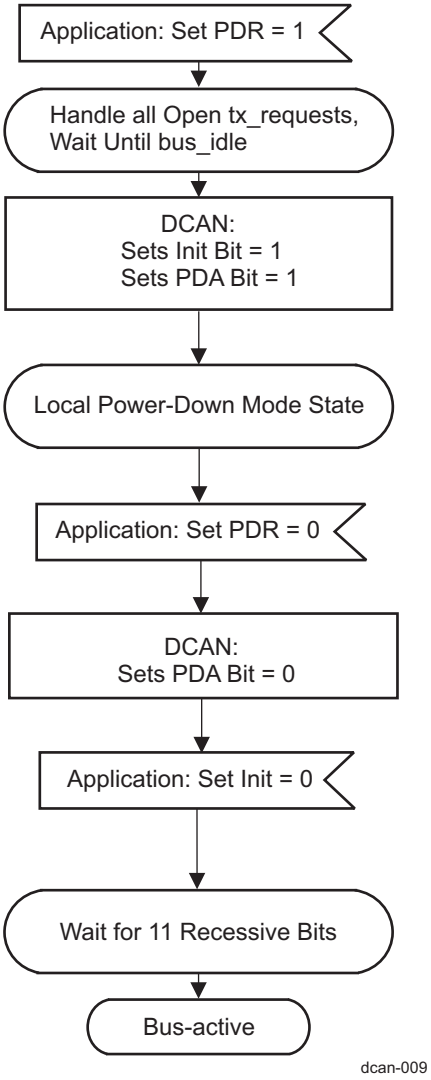
There is one way to wake up the DCAN from local power-down mode:

- The application could wake up the DCAN module manually by clearing the DCAN_CTL[24] PDR bit and then clearing the DCAN_CTL[0] INIT.

After the INIT bit has been cleared, the module waits until it detects 11 consecutive recessive bits on the CAN_RX pin and then goes bus-active again.

Figure 11-80 shows a flow diagram about entering and leaving local power-down mode.

Figure 11-80. Local Power-Down Mode Flow Diagram



11.2.4.5 SECDED Mechanism

The DCAN module provides a single error correction and double error detection (SECDED) mechanism to ensure data integrity of Message RAM data. For each message object (136 bits) in the Message RAM, 9 ECC bits will be calculated.

The ECC bits are stored in a dedicated RAM. They will be generated on write accesses and will be checked on read accesses.

The SECDED functionality can be enabled or disabled by DCAN_CTL[13-10] PMD bit field.

In case of disabled SECDED, the ECC bits will be left unchanged on write access to data area and no correction or check will be done on read access.

If SECDED is enabled, ECC bits will be automatically generated and checked. The ECC bits are memory mapped as described in Section 11.2.4.11.3, *ECC RAM*.

With the ECCMODE field in the DCAN_ECC_CS register the single bit error correction can be enabled or disabled (default: enabled).

NOTE: During RAM initialization, no SECDED check will be done, but if the PMD bit is set, the ECC bits will be generated.

11.2.4.5.1 Behavior on Single Bit Error

If a single bit error is detected with single bit error correction enabled, the correction will be done and the SEFLG in the DCAN_ECC_CS register will be set.

If single bit error correction is disabled and a single bit error is detected then the SEFLG in the DCAN_ECC_CS register and the the PER bit in the Error and Status register (DCAN_ES) will be set. If error interrupts are enabled, also an interrupt would be generated. In order to avoid the transmission of invalid data over the CAN bus, the MsgVal bit of the message object will be reset.

If single bit error correction is disabled and a single bit error is detected then the DCAN_n_ERR_INT signal (where n = 0 or 1) is generating a high pulse for one interface clock period.

The message object number where the single bit error has occurred will be indicated in the DCAN_ECC_SERR Register.

When single bit error correction is disabled the message object data can be read by the host CPU, independently of single bit errors. Thus, the application has to ensure that the read data is valid, for example, by immediately checking the DCAN_ECC_SERR on single bit error.

11.2.4.5.2 Behavior on Double Bit Error

If a double bit error is detected, then the DEFLG in the DCAN_ECC_CS register and the PER bit in DCAN_ES will be set. If error interrupts are enabled, also an interrupt would be generated. In order to avoid the transmission of invalid data over the CAN bus, the MsgVal bit of the message object will be reset. The message object number will be indicated in the DCAN_PERR register.

Additionally the signal DCAN_n_ERR_INT signal (where n = 0 or 1) signalizes — by generating a high pulse for one interface clock period — the double bit error occurrence for the host CPU. The message object data can be read by the host CPU, independently of double bit errors. Thus, the application has to ensure that the read data is valid, for example, by immediately checking the DCAN_PERR register on double bit error interrupt.

11.2.4.5.3 SECDED Testing

Testing of the SECDED mechanism can be implemented by using the diagnostic mode, which is enabled with the DCAN_ECCDIAG register. The following procedure can be used:

1. Disable SECDED using DCAN_CTL register. Enable diagnostic mode using the DCAN_ECCDIAG register.
2. Write to corrupt the data (in RDA mode) or ECC bits.
3. Enable SECDED and read data for which ECC is corrupted (either in RDA mode or via IFx registers).
4. Single bit error or double bit error flag will be set in the diagnostic status register (DCAN_ECCDIAG_STAT) and in the DCAN_ECC_CS register, accordingly. A double bit error or a single bit error with single bit error correction disabled also triggers the PER flag.
5. Disable diagnostic mode.

11.2.4.6 Debug/Suspend Mode

The module supports the usage of an external debug unit by providing functions like pausing DCAN activities and making message RAM content accessible via interconnect interface.

Before entering debug/suspend mode, the DCAN will either wait until a started transmission or reception will be finished and bus idle state is recognized, or immediately interrupt a current transmission or reception. This is depending on bit DCAN_CTL[8] IDS in the CAN control register.

Afterwards, the DCAN enters debug/suspend mode, indicated by DCAN_CTL[16] INITDBG flag.

During debug/suspend mode, all DCAN registers can be accessed. Reading reserved bits will return '0'. Writing to reserved bits will have no effect.

Also, the message RAM will be memory mapped. This allows the external debug unit to read the message RAM. For the memory organization, see Section 11.2.4.11.4, *Message RAM Representation in Debug/Suspend Mode*.

NOTE: During debug/suspend mode, the message RAM cannot be accessed via the IFx register sets.

NOTE: Writing to control registers in debug/suspend mode may influence the CAN state machine and further message handling.

For debug support, the auto clear functionality of the following DCAN registers is disabled:

- DCAN_ES register (clear of status flags by read)
- DCAN_IF1CMD/DCAN_IF2CMD command registers (clear of [14] DMAACTIVE flag by read/write)

11.2.4.7 Configuration of Message Objects Description

The whole message RAM should be configured before the end of the initialization, however it is also possible to change the configuration of message objects during CAN communication.

The CAN software driver must offer subroutines that:

- Transfer a complete message structure into a message object. (Configuration)
- Transfer the data bytes of a message into a message object and set TxRqst and NewDat. (Start a new transmission)
- Get the data bytes of a message from a message object and clear NewDat (and IntPnd). (Read received data)
- Get the complete message from a message object and clear NewDat (and IntPnd). (Read a received message, including identifier, from a message object with UMask = '1'.)

Parameters of the subroutines are the Message Number and a pointer to a complete message structure or to the data bytes of a message structure.

Two examples of assigning the IFx interface register sets to these subroutines are shown here:

- In the first method, the tasks of the application program that may access the module are divided into two groups. Each group is restricted to the use of one of the interface register sets. The tasks of one group may interrupt tasks of the other group, but not of the same group.
- In the second method, which may be a special case of the first method, there are only two tasks in the application program that access the module. A Read_Message task that uses IF2 or IF3 to get received messages from the message RAM and a Write_Message task that uses IF1 to write messages to be transmitted (or to be configured) into the message RAM. Both tasks may interrupt each other.

11.2.4.7.1 Configuration of a Transmit Object for Data Frames

Table 11-206 shows how a transmit object can be initialized.

Table 11-206. Initialization of a Transmit Object

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	1	0	0	0	appl.	0	appl.	0

The arbitration bits (ID[28-0] and Xtd bit) are given by the application. They define the identifier and type of the outgoing message. If an 11-bit identifier (standard frame) is used (Xtd = '0'), it is programmed to ID[28-18]. In this case, ID[17-0] can be ignored.

The data length and data itself (DLC[3-0] and Data0-7) are given by the application. TxRqst and RmtEn should not be set before the data is valid.

If the TXIE bit is set, the IntPnd bit will be set after a successful transmission of the message object.

If the RmtEn bit is set, a matching received remote frame will cause the TxRqst bit to be set; the remote frame will autonomously be answered by a data frame.

The mask bits (Msk[28-0], UMask, MXtd, and MDir bits) may be used (UMask='1') to allow groups of remote frames with similar identifiers to set the TxRqst bit. The Dir bit should not be masked. For details, see Section 11.2.4.8.8, *Reception of Remote Frames*.

Identifier masking must be disabled (UMask = '0') if no remote frames are allowed to set the TxRqst bit (RmtEn = '0').

11.2.4.7.2 Configuration of a Transmit Object for Remote Frames

It is not necessary to configure transmit objects for the transmission of remote frames. Setting TxRqst for a receive object causes the transmission of a remote frame with the same identifier as the data frame for which this receive object is configured.

11.2.4.7.3 Configuration of a Single Receive Object for Data Frames

Table 11-207 shows how a receive object for data frames can be initialized.

Table 11-207. Initialization of a single Receive Object for Data Frames

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	0	0	0	appl.	0	0	0	0

The arbitration bits (ID[28-0] and Xtd bit) are given by the application. They define the identifier and type of accepted received messages. If an 11-bit Identifier (standard frame) is used (Xtd = '0'), it is programmed to ID[28-18]. In this case, ID[17-0] can be ignored. When a data frame with an 11-bit Identifier is received, ID[17-0] is set to '0'.

The data length code (DLC[3-0]) is given by the application. When the message handler stores a data frame in the message object, it will store the received data length code and eight data bytes. If the data length code is less than 8, the remaining bytes of the message object may be overwritten by non specified values.

The mask bits (Msk[28-0], UMask, MXtd, and MDir bits) may be used (UMask = '1') to allow groups of data frames with similar identifiers to be accepted. The Dir bit should not be masked in typical applications. If some bits of the mask bits are set to 'don't care', the corresponding bits of the arbitration register (DCAN_IF1ARB, DCAN_IF2ARB, and DCAN_IF3ARB) will be overwritten by the bits of the stored data frame.

If the RxIE bit is set, the IntPnd bit will be set when a received data frame is accepted and stored in the message object.

If the TxRqst bit is set, the transmission of a remote frame with the same identifier as actually stored in the arbitration bits will be triggered. The content of the arbitration bits may change if the mask bits are used (UMask = '1') for acceptance filtering.

11.2.4.7.4 Configuration of a Single Receive Object for Remote Frames

Table 11-208 shows how a receive object for remote frames can be initialized.

Table 11-208. Initialization of a Single Receive Object for Remote Frames

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	1	0	0	appl.	0	0	0	0

A receive object for remote frames may be used to monitor remote frames on the CAN bus. The remote frame stored in the receive object will not trigger the transmission of a data frame. Receive objects for remote frames may be expanded to a FIFO buffer (see Section 11.2.4.7.5, *Configuration of a FIFO Buffer*).

UMask must be set to '1.' The mask bits (Msk[28-0], UMask, MXtd, and MDir bits) may be set to 'must-match' or to 'don't care' to allow groups of remote frames with similar identifiers to be accepted. The Dir bit should not be masked in typical applications. For details, see Section 11.2.4.8.8, *Reception of Remote Frames*.

The arbitration bits (ID[28-0] and Xtd bit) may be given by the application. They define the identifier and type of accepted received remote frames. If some bits of the mask bits are set to 'don't care' the corresponding bits of the arbitration bits will be overwritten by the bits of the stored remote frame. If an 11-bit Identifier (standard frame) is used (Xtd = '0'), it is programmed to ID[28-18]. In this case, ID[17-0] can be ignored. When a remote frame with an 11-bit Identifier is received, ID[17-0] will be set to '0.'

The data length code (DLC[3-0]) may be given by the application. When the message handler stores a remote frame in the message object, it will store the received data length code. The data bytes of the message object will remain unchanged.

If the RxIE bit is set, the IntPnd bit will be set when a received remote frame is accepted and stored in the message object.

11.2.4.7.5 Configuration of a FIFO Buffer

With the exception of the EoB bit, the configuration of receive objects belonging to a FIFO buffer is the same as the configuration of a single receive object.

To concatenate multiple message objects to a FIFO buffer, the identifiers and masks (if used) of these message objects have to be programmed to matching values. Due to the implicit priority of the message objects, the message object with the lowest number will be the first message object of the FIFO buffer. The EoB bit of all message objects of a FIFO buffer except the last one have to be programmed to zero. The EoB bits of the last message object of a FIFO Buffer is set to one, configuring it as the end of the block.

11.2.4.8 Message Handling

When initialization is finished, the DCAN module synchronizes itself to the traffic on the CAN bus. It does acceptance filtering on received messages and stores those frames that are accepted into the designated message objects. The application has to update the data of the messages to be transmitted and to enable and request their transmission. The transmission is requested automatically when a matching remote frame is received.

The application may read messages which are received and accepted. Messages that are not read before the next messages is accepted for the same message object will be overwritten.

Messages may be read interrupt-driven or after polling of NewDat.

11.2.4.8.1 Message Handler Overview

The message handler state machine controls the data transfer between the Rx/Tx shift register of the CAN core and the message RAM. It performs the following tasks:

- Data transfer from message RAM to CAN core (messages to be transmitted)
- Data transfer from CAN core to the message RAM (received messages)

- Data transfer from CAN core to the acceptance filtering unit
- Scanning of message RAM for a matching message object (acceptance filtering)
- Scanning the same message object after being changed by IF1/IF2 registers when priority is the same or higher as the message the object found by last scanning
- Handling of TxRqst flags
- Handling of interrupt flags

The message handler registers contains status flags of all message objects grouped into the following topics:

- Transmission Request Flags
- New Data Flags
- Interrupt Pending Flags
- Message Valid Registers.

Instead of collecting above listed status information of each message object via IFx registers separately, these message handler registers provides a fast and easy way to get an overview, for example, about all pending transmission requests.

All message handler registers are read-only.

11.2.4.8.2 Receive/Transmit Priority

The receive/transmit priority for the message objects is attached to the message number, not to the CAN identifier. Message object 1 has the highest priority, while the last implemented message object has the lowest priority. If more than one transmission request is pending, they are serviced due to the priority of the corresponding message object so messages with the highest priority, for example, can be placed in the message objects with the lowest numbers.

The acceptance filtering for received data frames or remote frames is also done in ascending order of message objects, so a frame that has been accepted by a message object cannot be accepted by another message object with a higher message number. The last message object may be configured to accept any data frame or remote frame that was not accepted by any other message object, for nodes that need to log the complete message traffic on the CAN bus.

11.2.4.8.3 Transmission of Messages in Event Driven CAN Communication

If the shift register of the CAN core is ready for loading and if there is no data transfer between the IFx registers and message RAM, the MSGVAL bits in the Message Valid register (DCAN_MSGVAL12 to DCAN_MSGVAL78) and the TXRQST bits in the transmission request register (DCAN_TXRQ12 to DCAN_TXRQ78) are evaluated. The valid message object with the highest priority pending transmission request is loaded into the shift register by the message handler and the transmission is started. The message object's NewDat bit is reset.

After a successful transmission and if no new data was written to the message object (NewDat = '0') since the start of the transmission, the TxRqst bit will be reset. If TxIE is set, IntPnd will be set after a successful transmission. If the DCAN has lost the arbitration or if an error occurred during the transmission, the message will be retransmitted as soon as the CAN bus is free again. If meanwhile the transmission of a message with higher priority has been requested, the messages will be transmitted in the order of their priority.

If automatic retransmission mode is disabled by setting the DCAN_CTL[5] DAR bit, the behavior of bits TXRQST and NEWDAT in the DCAN_IF1CMD/DCAN_IF2CMD register of the interface register set is as follows:

- When a transmission starts, the TxRqst bit of the respective interface register set is reset, while bit NewDat remains set.
- When the transmission has been successfully completed, the NewDat bit is reset.

When a transmission failed (lost arbitration or error), bit NewDat remains set. To restart the transmission, the application has to set TxRqst again.

Received remote frames do not require a receive object. They will automatically trigger the transmission of a data frame, if in the matching transmit object the RmtEn bit is set.

11.2.4.8.4 Updating a Transmit Object

The software may update the data bytes of a transmit object any time via the IF1/IF2 interface registers, neither MSGVAL, nor TXRQST have to be reset before the update.

Even if only a part of the data bytes are to be updated, all four bytes in the corresponding IF1/IF2 Data A register (DCAN_IF1DATA/DCAN_IF2DATA) or IF1/IF2 Data B register (DCAN_IF1DATB/DCAN_IF2DATB) have to be valid before the content of that register is transferred to the message object. Either the software has to write all four bytes into the IF1/IF2 data register or the message object is transferred to the IF1/IF2 data register before the software writes the new data bytes.

When only the data bytes are updated, first 0x87 can be written to bits [23-16] of the IF1/IF2 Command register (DCAN_IF1CMD/DCAN_IF2CMD) and then the number of the message object is written to bits [7-0] MESSAGE_NUMBER of the command register, concurrently updating the data bytes and setting TXRQST with NEWDAT.

To prevent the reset of TxRqst at the end of a transmission that may already be in progress while the data is updated, NewDat has to be set together with TxRqst in event driven CAN communication. For details, see Section 11.2.4.8.3, *Transmission of Messages in Event Driven CAN Communication*.

When NewDat is set together with TxRqst, NewDat will be reset as soon as the new transmission has started.

11.2.4.8.5 Changing a Transmit Object

If the number of implemented message objects is not sufficient to be used as permanent message objects only, the transmit objects may be managed dynamically. The software can write the whole message (arbitration, control, and data) into the interface register. The bits [23-16] of the command register (DCAN_IF1CMD/DCAN_IF2CMD) can be set to 0xB7 for the transfer of the whole message object content into the message object. Neither Dir, nor TxRqst have to be reset before this operation.

If a previously requested transmission of this message object is not completed but already in progress, it will be continued; however, it will not be repeated if it is disturbed.

To only update the data bytes of a message to be transmitted, bits [23-16] of the command register (DCAN_IF1CMD/ DCAN_IF2CMD) should be set to 0x87.

NOTE: After the update of the transmit object, the interface register set will contain a copy of the actual contents of the object, including the part that had not been updated.

11.2.4.8.6 Acceptance Filtering of Received Messages

When the arbitration and control bits (Identifier + IDE + RTR + DLC) of an incoming message are completely shifted into the shift register of the CAN core, the message handler starts the scan of the message RAM for a matching valid message object:

- The acceptance filtering unit is loaded with the arbitration bits from the CAN core shift register.
- Then the arbitration and mask bits (including MsgVal, UMask, NewDat, and EoB) of message object 1 are loaded into the acceptance filtering unit and are compared with the arbitration bits from the shift register. This is repeated for all following message objects until a matching message object is found, or until the end of the message RAM is reached.
- If a match occurs, the scanning is stopped and the message handler proceeds depending on the type of the frame (data frame or remote frame) received.

11.2.4.8.7 Reception of Data Frames

The message handler stores the message from the CAN core shift register into the respective message object in the message RAM. Not only the data bytes, but all arbitration bits and the data length code are stored into the corresponding message object. This ensures that the data bytes stay associated to the identifier even if arbitration mask registers are used.

The NewDat bit is set to indicate that new data (not yet seen by the software) has been received. The software should reset the NewDat bit when it reads the message object. If at the time of the reception the NewDat bit was already set, MsgLst is set to indicate that the previous data (supposedly not seen by the software) is lost. If the RxIE bit is set, the IntPnd bit is set, causing the DCAN_INT to point to this message object.

The TxRqst bit of this message object is reset to prevent the transmission of a remote frame, while the requested data frame has just been received.

11.2.4.8.8 Reception of Remote Frames

When a remote frame is received, three different configurations of the matching message object have to be considered:

- Dir = '1' (direction = transmit), RmtEn = '1', UMask = '1' or '0'

  The TxRqst bit of this message object is set at the reception of a matching remote frame. The rest of the message object remains unchanged.
- Dir = '1' (direction = transmit), RmtEn = '0', UMask = '0'

  The remote frame is ignored, this message object remains unchanged.
- Dir = '1' (direction = transmit), RmtEn = '0', UMask = '1'

  The remote frame is treated similar to a received data frame. At the reception of a matching Remote Frame, the TxRqst bit of this message object is reset. The arbitration and control bits (Identifier + IDE + RTR + DLC) from the shift register are stored in the message object in the message RAM and the NewDat bit of this message object is set. The data bytes of the message object remain unchanged.

11.2.4.8.9 Reading Received Messages

The software may read a received message any time via the IFx interface register. The data consistency is guaranteed by the message handler state machine. Typically the software will write first 0x7F to bits [23-16] and then the number of the message object to bits [7-0] MESSAGE_NUMBER of the command register (DCAN_IF1CMD/ DCAN_IF2CMD). That combination will transfer the entire received message from the message RAM into the interface register set. Additionally, the bits NewDat and IntPnd are cleared in the message RAM (not in the interface register set). The values of these bits in the message control register (DCAN_IF1MCTL/DCAN_IF2MCTL/DCAN_IF3MCTL) always reflect the status before resetting the bits. If the message object uses masks for acceptance filtering, the arbitration bits show which of the different matching messages has been received.

The actual value of NewDat shows whether a new message has been received since last time when this message object was read. The actual value of MsgLst shows whether more than one message has been received since the last time when this message object was read. MsgLst will not be automatically reset.

11.2.4.8.10 Requesting New Data for a Receive Object

By means of a remote frame, the software may request another CAN node to provide new data for a receive object. Setting the TxRqst bit of a receive object will cause the transmission of a remote frame with the identifier of the receive object. This remote frame triggers the other CAN node to start the transmission of the matching data frame. If the matching data frame is received before the remote frame could be transmitted, the TxRqst bit is automatically reset. Setting the TxRqst bit without changing the contents of a message object requires the value 0x84 in bits [23-16] of the command register (DCAN_IF1CMD/ DCAN_IF2CMD).

11.2.4.8.11 Storing Received Messages in FIFO Buffers

Several message objects may be grouped to form one or more FIFO buffers. Each FIFO buffer configured to store received messages with a particular (group of) identifier(s). Arbitration and mask registers of the FIFO buffer's message objects are identical. The end of buffer (EoB) bits of all but the last of the FIFO buffer's message objects are '0'; in the last one the EoB bit is '1.'

Received messages with identifiers matching to a FIFO buffer are stored into a message object of this FIFO buffer, starting with the message object with the lowest message number. When a message is stored into a message object of a FIFO buffer, the NewDat bit of this message object is set. By setting NewDat while EoB is '0', the message object is locked for further write accesses by the message handler until the software has cleared the NewDat bit.

Messages are stored into a FIFO buffer until the last message object of this FIFO buffer is reached. If none of the preceding message objects is released by writing NewDat to '0,' all further messages for this FIFO buffer will be written into the last message object of the FIFO buffer (EoB = '1') and therefore overwrite previous messages in this message object.

11.2.4.8.12 Reading From a FIFO Buffer

Several messages may be accumulated in a set of message objects which are concatenated to form a FIFO buffer before the application program is required (in order to avoid the loss of data) to empty the buffer.

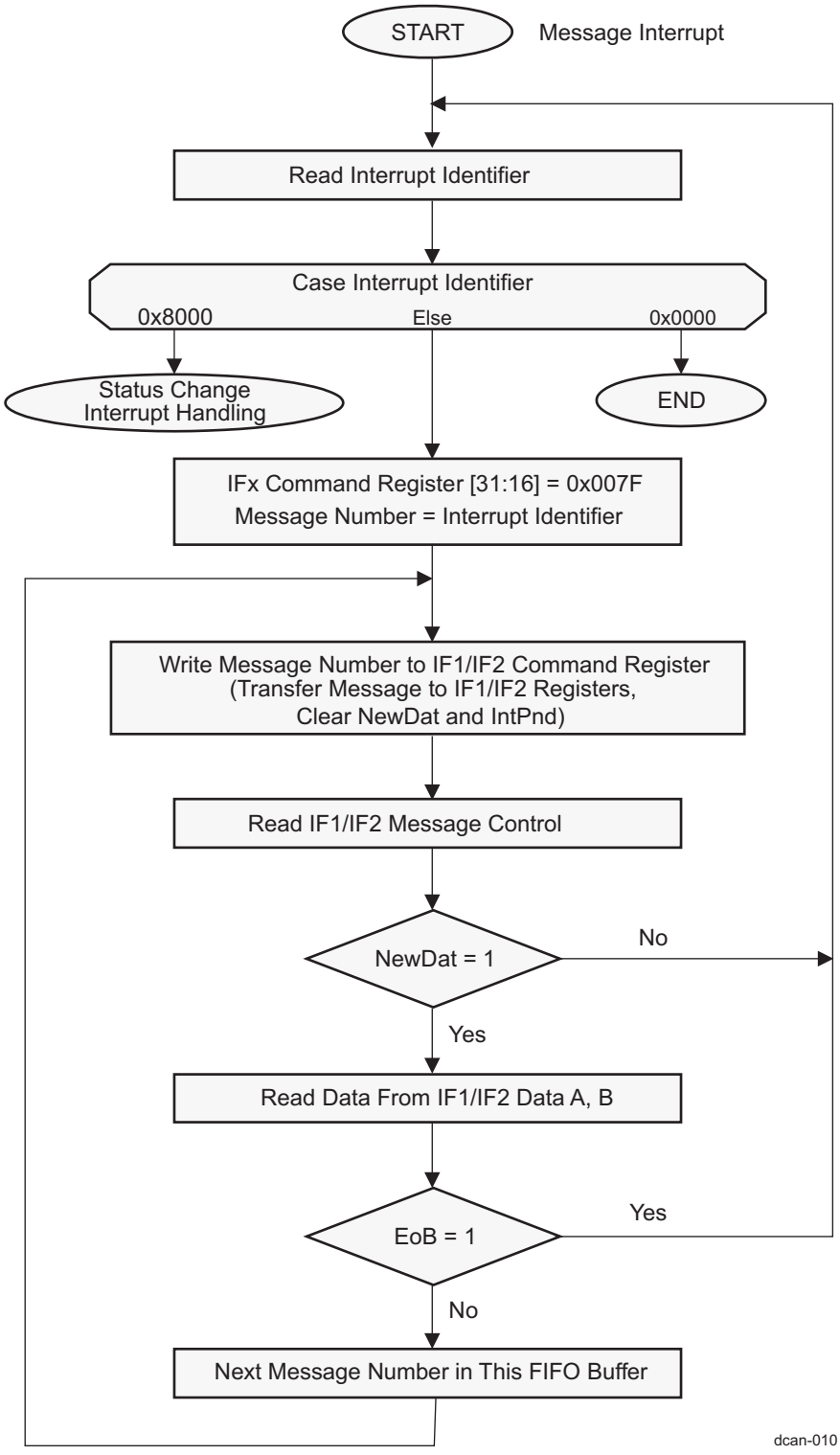
A FIFO buffer of length N will store N–1, plus the last received message since last time it was cleared.

A FIFO buffer is cleared by reading and resetting the NewDat bits of all its message objects, starting at the FIFO Object with the lowest message number. This should be done in a subroutine following the example shown in Figure 11-81.

NOTE: All message objects of a FIFO buffer needs to be read and cleared before the next batch of messages can be stored. Otherwise, true FIFO functionality can not be guaranteed, since the message objects of a partly read buffer will be re-filled according to the normal (descending) priority.

Reading from a FIFO buffer message object and resetting its NewDat bit is handled the same way as reading from a single message object.

Figure 11-81. Software Handling of a FIFO Buffer (Interrupt Driven)



11.2.4.9 CAN Bit Timing

The DCAN supports bit rates between < 1 kBit/s and 1000 kBit/s.

Each member of the CAN network has its own clock generator, typically derived from a crystal oscillator. The bit timing parameters can be configured individually for each CAN node, creating a common bit rate even though the CAN nodes oscillator periods (f_{osc}) may be different.

The frequencies of these oscillators are not absolutely stable. Small variations are caused by changes in temperature or voltage and by deteriorating components. As long as the variations remain inside a specific oscillator tolerance range (df), the CAN nodes are able to compensate for the different bit rates by resynchronizing to the bit stream.

In many cases, the CAN bit synchronization will amend a faulty configuration of the CAN bit timing to such a degree that only occasionally an error frame is generated. In the case of arbitration however, when two or more CAN nodes simultaneously try to transmit a frame, a misplaced sample point may cause one of the transmitters to become error passive.

The analysis of such sporadic errors requires a detailed knowledge of CAN bit synchronization inside CAN node and CAN nodes interaction on the CAN bus.

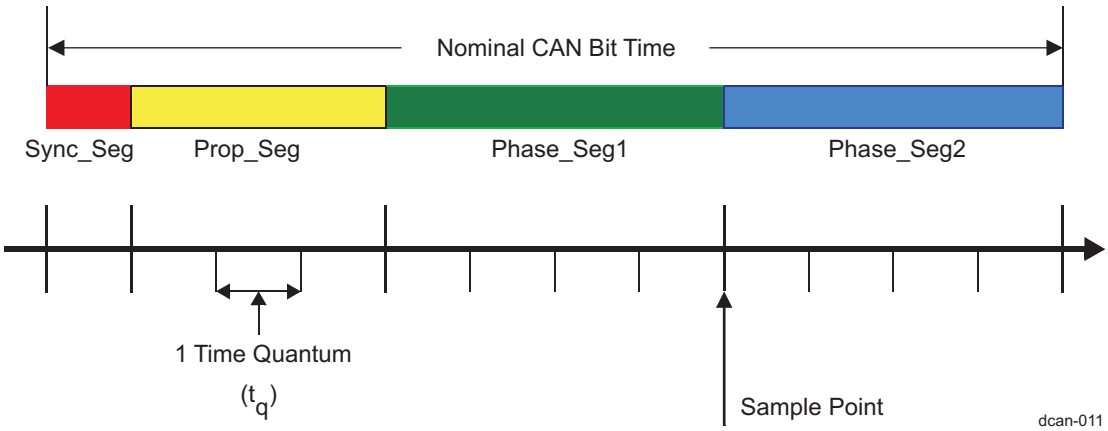
Even if minor errors in the configuration of the CAN bit timing do not result in immediate failure, the performance of CAN network can be reduced significantly.

11.2.4.9.1 Bit Time and Bit Rate

According to the CAN specification, the bit time is divided into four segments (see Figure 11-82):

- Synchronization segment (Sync_Seg)
- Propagation time segment (Prop_Seg)
- Phase buffer segment 1 (Phase_Seg1)
- Phase buffer segment 2 (Phase_Seg2)

Figure 11-82. Bit Timing



Each segment consists of a specific number of time quanta. The length of one time quantum (t_q), which is the basic time unit of the bit time, is given by the CAN_CLK and the baud rate prescalers (BRPE and BRP). With these two baud rate prescalers combined, divider values from 1 to 1024 can be programmed:

t_q = Baud Rate Prescaler / CAN_CLK

Apart from the fixed length of the synchronization segment, these numbers are programmable. Table 11-209 describes the minimum programmable ranges required by the CAN protocol.

A given bit rate may be met by different bit time configurations.

Table 11-209. Parameters of the CAN Bit Time

Parameter	Range	Remark
Sync_Seg	1 t_q (fixed)	Synchronization of bus input to CAN_CLK
Prop_Seg	[1 … 8] t_q	Compensates for the physical delay times
Phase_Seg1	[1 … 8] t_q	May be lengthened temporarily by synchronization

Table 11-209. Parameters of the CAN Bit Time (continued)

Parameter	Range	Remark
Phase_Seg2	[1 … 8] t_q	May be shortened temporarily by synchronization
Synchronization Jump Width (SJW)	[1 … 4] t_q	May not be longer than either Phase Buffer Segment

NOTE: For proper functionality of the CAN network, the physical delay times and the oscillator's tolerance range have to be considered.

11.2.4.9.1.1 Synchronization Segment

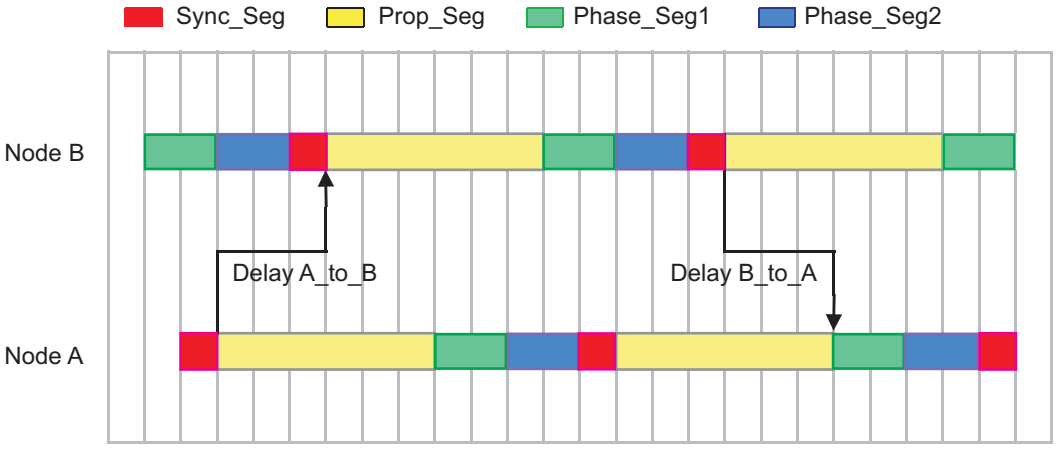
The synchronization segment (Sync_Seg) is the part of the bit time where edges of the CAN bus level are expected to occur. If an edge occurs outside of Sync_Seg, its distance to the Sync_Seg is called the phase error of this edge.

11.2.4.9.1.2 Propagation Time Segment

This part of the bit time is used to compensate physical delay times within the CAN network. These delay times consist of the signal propagation time on the bus and the internal delay time of the CAN nodes.

Any CAN node synchronized to the bit stream on the CAN bus can be out of phase with the transmitter of the bit stream, caused by the signal propagation time between the two nodes. The CAN protocol's nondestructive bitwise arbitration and the dominant acknowledge bit provided by receivers of CAN messages require that a CAN node transmitting a bit stream must also be able to receive dominant bits transmitted by other CAN nodes that are synchronized to that bit stream. The example in Figure 11-83 shows the phase shift and propagation times between two CAN nodes.

Figure 11-83. The Propagation Time Segment



Delay A_to_B ≥ node output delay(A) + bus line delay(AÆB) + node input delay(B)

Prop_Seg ≥ Delay A_to_B + Delay B_to_A

Prop_Seg ≥ 2 [max(node output delay + bus line delay + node input delay)]

dcan-012

In this example, both nodes A and B are transmitters performing an arbitration for the CAN bus. The node A has sent its start of frame bit less than one bit time earlier than node B, therefore node B has synchronized itself to the received edge from recessive to dominant. Since node B has received this edge delay(A_to_B) after it has been transmitted, node B's bit timing segments are shifted with regard to node A. Node B sends an identifier with higher priority and so it will win the arbitration at a specific identifier bit when it transmits a dominant bit while node A transmits a recessive bit. The dominant bit transmitted by node B will arrive at node A after the delay (B_to_A).

Due to oscillator tolerances, the actual position of node A's sample point can be anywhere inside the nominal range of node A's Phase Buffer Segments, so the bit transmitted by node B must arrive at node A before the start of Phase_Seg1. This condition defines the length of Prop_Seg.

If the edge from recessive to dominant transmitted by node B would arrive at node A after the start of Phase_Seg1, it could happen that node A samples a recessive bit instead of a dominant bit, resulting in a bit error and the destruction of the current frame by an error flag.

This error only occurs when two nodes arbitrate for the CAN bus which have oscillators of opposite ends of the tolerance range and are separated by a long bus line; this is an example of a minor error in the bit timing configuration (Prop_Seg too short) that causes sporadic bus errors.

Some CAN implementations provide an optional 3-Sample Mode. The DCAN does not. In this mode, the CAN bus input signal passes a digital low-pass filter, using three samples and a majority logic to determine the valid bit value. This results in an additional input delay of 1 t_q, requiring a longer Prop_Seg.

11.2.4.9.1.3 Phase Buffer Segments and Synchronization

The phase buffer segments (Phase_Seg1 and Phase_Seg2) and the synchronization jump width (SJW) are used to compensate for the oscillator tolerance.

The phase buffer segments surround the sample point and may be lengthened or shortened by synchronization.

The synchronization jump width (SJW) defines how far the resynchronizing mechanism may move the sample point inside the limits defined by the phase buffer segments to compensate for edge phase errors.

Synchronizations occur on edges from recessive to dominant. Their purpose is to control the distance between edges and sample points.

Edges are detected by sampling the actual bus level in each time quantum and comparing it with the bus level at the previous sample point. A synchronization may be done only if a recessive bit was sampled at the previous sample point and if the actual time quantum's bus level is dominant.

An edge is synchronous if it occurs inside of Sync_Seg; otherwise, its distance to the Sync_Seg is the edge phase error, measured in time quanta. If the edge occurs before Sync_Seg, the phase error is negative, else it is positive.

Two types of synchronization exist: hard synchronization and resynchronizing. A hard synchronization is done once at the start of a frame; inside a frame, only resynchronization is possible.

- Hard Synchronization

  After a hard synchronization, the bit time is restarted with the end of Sync_Seg, regardless of the edge phase error. Thus hard synchronization forces the edge which has caused the hard synchronization, to lie within the synchronization segment of the restarted bit time.

- Bit Resynchronizations

  Resynchronization leads to a shortening or lengthening of the Bit time such that the position of the sample point is shifted with regard to the edge.

  When the phase error of the edge which causes resynchronization is positive, Phase_Seg1 is lengthened. If the magnitude of the phase error is less than SJW, Phase_Seg1 is lengthened by the magnitude of the phase error, else it is lengthened by SJW.

  When the phase error of the edge which causes Resynchronization is negative, Phase_Seg2 is shortened. If the magnitude of the phase error is less than SJW, Phase_Seg2 is shortened by the magnitude of the phase error, else it is shortened by SJW.

If the magnitude of the phase error of the edge is less than or equal to the programmed value of SJW, the results of hard synchronization and resynchronization are the same. If the magnitude of the phase error is larger than SJW, the resynchronization cannot compensate the phase error completely, and an error of (phase error — SJW) remains.

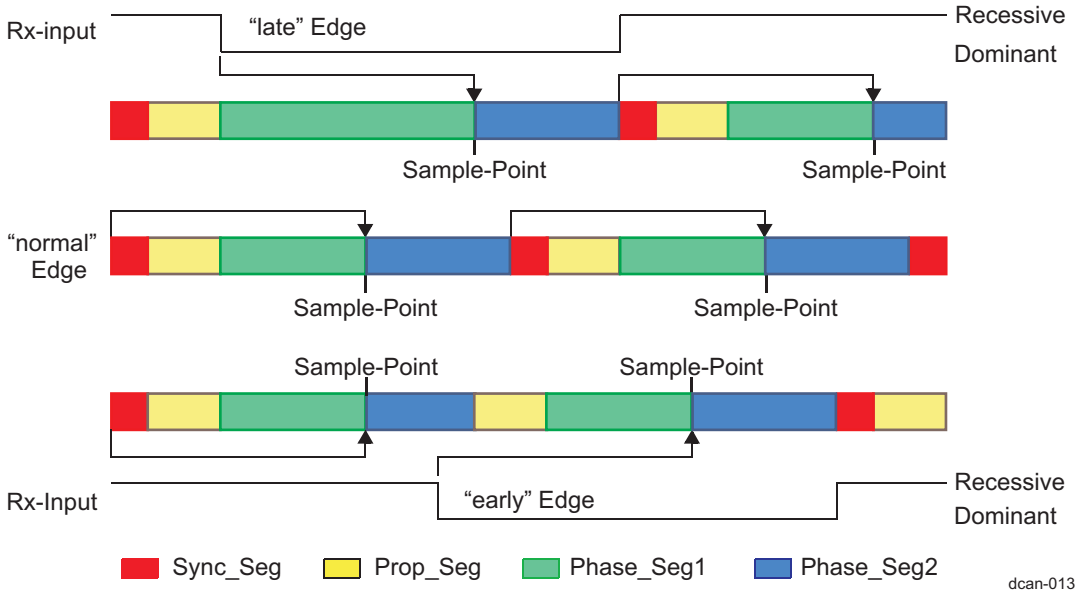
Only one synchronization may be done between two sample points. The synchronizations maintain a minimum distance between edges and sample points, giving the bus level time to stabilize and filtering out spikes that are shorter than (Prop_Seg + Phase_Seg1).

Apart from noise spikes, most synchronizations are caused by arbitration. All nodes synchronize “hard” on the edge transmitted by the “leading” transceiver that started transmitting first, but due to propagation delay times, they cannot become ideally synchronized. The leading transmitter does not necessarily win the arbitration; therefore, the receivers have to synchronize themselves to different transmitters that subsequently take the lead and that are differently synchronized to the previously leading transmitter. The same happens at the acknowledge field, where the transmitter and some of the receivers will have to synchronize to that receiver that takes the lead in the transmission of the dominant acknowledge bit.

Synchronizations after the end of the arbitration will be caused by oscillator tolerance, when the differences in the oscillator’s clock periods of transmitter and receivers sum up during the time between synchronizations (at most ten bits). These summarized differences may not be longer than the SJW, limiting the oscillator’s tolerance range.

Figure 11-84 shows how the phase buffer segments are used to compensate for phase errors. There are three drawings of each two consecutive bit timings. The upper drawing shows the synchronization on a “late” edge, the lower drawing shows the synchronization on an “early” edge, and the middle drawing is the reference without synchronization.

Figure 11-84. Synchronization on Late and Early Edges



In the first example, an edge from recessive to dominant occurs at the end of Prop_Seg. The edge is "late" since it occurs after the Sync_Seg. Reacting to the late edge, Phase_Seg1 is lengthened so that the distance from the edge to the sample point is the same as it would have been from the Sync_Seg to the sample point if no edge had occurred. The phase error of this late edge is less than SJW, so it is fully compensated and the edge from dominant to recessive at the end of the bit, which is one nominal bit time long, occurs in the Sync_Seg.

In the second example, an edge from recessive to dominant occurs during Phase_Seg2. The edge is "early" since it occurs before a Sync_Seg. Reacting to the early edge, Phase_Seg2 is shortened and Sync_Seg is omitted, so that the distance from the edge to the sample point is the same as it would have been from a Sync_Seg to the sample point if no edge had occurred. As in the previous example, the magnitude of this early edge’s phase error is less than SJW, so it is fully compensated.

The phase buffer segments are lengthened or shortened temporarily only; at the next bit time, the segments return to their nominal programmed values.

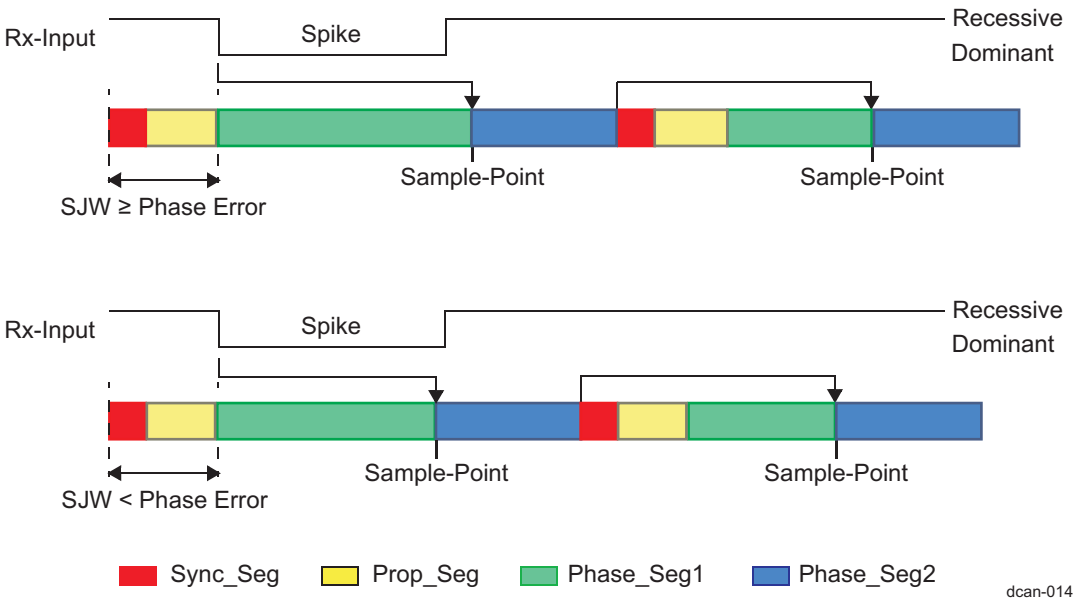
In these examples, the bit timing is seen from the point of view of the CAN implementation’s state machine, where the bit time starts and ends at the sample points. The state machine omits Sync_Seg when synchronizing on an early edge because it cannot subsequently redefine that time quantum of Phase_Seg2 where the edge occurs to be the Sync_Seg.

Figure 11-85 shows how short dominant noise spikes are filtered by synchronizations. In both examples, the spike starts at the end of Prop_Seg and has the length of (Prop_Seg + Phase_Seg1).

In the first example, the synchronization jump width is greater than or equal to the phase error of the spike's edge from recessive to dominant. Therefore the sample point is shifted after the end of the spike; a recessive bus level is sampled.

In the second example, SJW is shorter than the phase error, so the sample point cannot be shifted far enough; the dominant spike is sampled as actual bus level.

Figure 11-85. Filtering of Short Dominant Spikes



11.2.4.9.1.4 Oscillator Tolerance Range

With the introduction of CAN protocol version 1.2, the option to synchronize on edges from dominant to recessive became obsolete. Only edges from recessive to dominant are considered for synchronization. The protocol update to version 2.0 (A and B) had no influence on the oscillator tolerance.

The tolerance range df for an oscillator's frequency f_{osc} around the nominal frequency f_{nom} with:

$$(1 - df) \bullet f_{nom} \le f_{osc} \le (1 + df) \bullet f_{nom}$$

dcan-e015

depends on the proportions of Phase_Seg1, Phase_Seg2, SJW, and the bit time. The maximum tolerance df is the defined by two conditions (both shall be met):

$$\text{I:} \quad df \le \frac{\min(TSeg1, TSeg2)}{2x\left(13x\left(bit\_time - TSeg2\right)\right)}$$

$$\text{II:} \quad df \le \frac{SJW}{20xbit\_time}$$

dcan-e016

It has to be considered that SJW may not be larger than the smaller of the phase buffer segments and that the propagation time segment limits that part of the bit time that may be used for the phase buffer segments.

The combination Prop_Seg = 1 and Phase_Seg1 = Phase_Seg2 = SJW = 4 allows the largest possible oscillator tolerance of 1.58%. This combination with a propagation time segment of only 10% of the bit time is not suitable for short bit times; it can be used for bit rates of up to 125 kBit/s (bit time = 8 µs) with a bus length of 40 m.

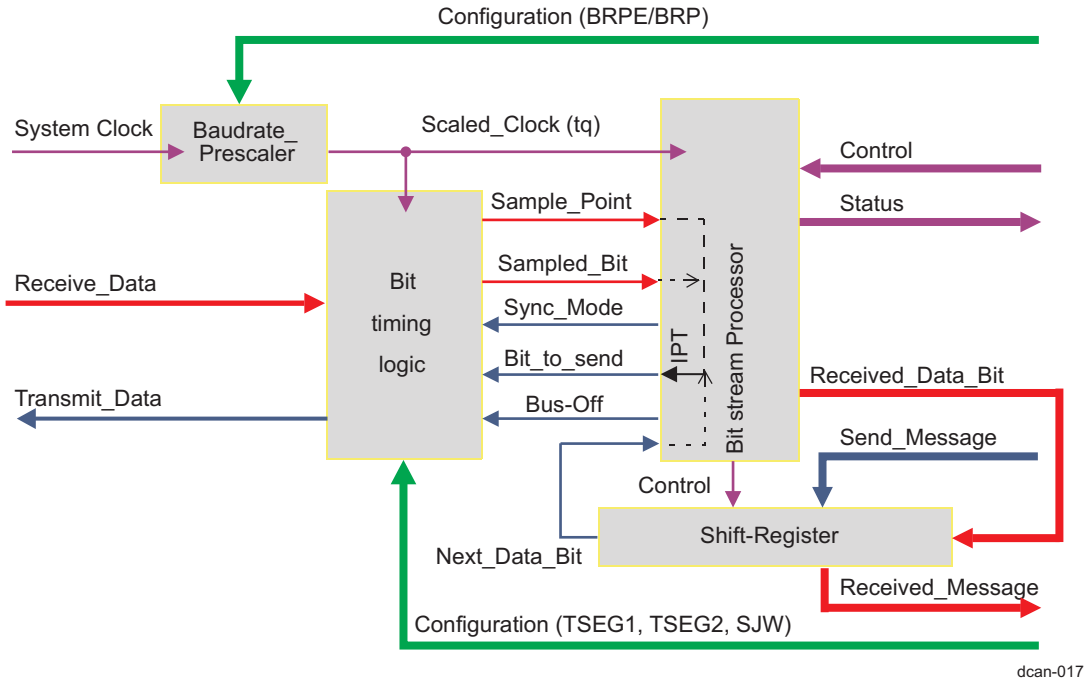
11.2.4.9.2 DCAN Bit Timing Registers

In the DCAN, the bit timing configuration is programmed in DCAN_BTR[14-0], additionally a baud rate prescaler extension of four bits (DCAN_BTR[19-16] BRPE) is provided.

- The sum of Prop_Seg and Phase_Seg1 is set in [11-8] TSEG1
- Phase_Seg2 in [14-12] TSEG2
- Synchronization JumpWidth in [7-6] SJW
- and baud rate prescaler [5-0] BRP (plus [19-16] BRPE).

Figure 11-86 shows the CAN protocol controller.

Figure 11-86. Structure of the CAN Core's CAN Protocol Controller



In DCAN_BTR register, the components TSEG1, TSEG2, SJW and BRP have to be programmed to a numerical value that is one less than its functional value; so instead of values in the range of [1; n], values in the range of [0; n–1] are programmed. That way, for example, SJW (functional range of [1; 4]) is represented by only two bits.

Therefore, the length of the bit time is (programmed values) [TSEG1 + TSEG2 + 3] t_q or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] t_q.

The data in the bit timing register (DCAN_BTR) is the configuration input of the CAN protocol controller. The baud rate prescaler (configured by BRPE/BRP) defines the length of the time quantum (the basic time unit of the bit time); the bit timing logic (configured by TSEG1, TSEG2, and SJW) defines the number of time quanta in the bit time.

The processing of the bit time, the calculation of the position of the sample point, and occasional synchronizations are controlled by the bit timing state machine, which is evaluated once each time quantum. The rest of the CAN protocol controller, the bit stream processor (BSP) state machine, is evaluated once each bit time, at the sample point.

The shift register serializes the messages to be sent and parallelizes received messages. Its loading and shifting is controlled by the BSP. The BSP translates messages into frames and vice versa. It generates and discards the enclosing fixed format bits, inserts and extracts stuff bits, calculates and checks the CRC code, performs the error management, and decides which type of synchronization is to be used. It is evaluated at the sample point and processes the sampled bus input bit. The time after the sample point that is needed to calculate the next bit to be sent (for example, data bit, CRC bit, stuff bit, error flag, or idle) is called the information processing time (IPT), which is 0 tq for the DCAN.

Generally, the IPT is CAN controller-specific, but may not be longer than 2 tq. The IPC length is the lower limit of the programmed length of Phase_Seg2. In case of a synchronization, Phase_Seg2 may be shortened to a value less than IPT, which does not affect bus timing.

11.2.4.9.2.1 Calculation of the Bit Timing Parameters

Usually, the calculation of the bit timing configuration starts with a desired bit rate or bit time. The resulting bit time (1 / Bit rate) must be an integer multiple of the CAN clock period.

The bit time may consist of 8 to 25 time quanta. The length of the time quantum t_q is defined by the baud rate prescaler with t_q = (Baud Rate Prescaler) / CAN_CLK. Several combinations may lead to the desired bit time, allowing iterations of the following steps.

First part of the bit time to be defined is the Prop_Seg. Its length depends on the delay times measured in the system. A maximum bus length as well as a maximum node delay has to be defined for expandible CAN bus systems. The resulting time for Prop_Seg is converted into time quanta (rounded up to the nearest integer multiple of t_q).

The Sync_Seg is 1 t_q long (fixed), leaving (bit time – Prop_Seg – 1) t_q for the two Phase Buffer Segments. If the number of remaining t_q is even, the Phase Buffer Segments have the same length, Phase_Seg2 = Phase_Seg1, else Phase_Seg2 = Phase_Seg1 + 1.

The minimum nominal length of Phase_Seg2 has to be regarded as well. Phase_Seg2 may not be shorter than any CAN controller's Information Processing Time in the network, which is device dependent and can be in the range of [0…2] t_q.

The length of the synchronization jump width is set to its maximum value, which is the minimum of four (4) and Phase_Seg1.

The oscillator tolerance range necessary for the resulting configuration is calculated by the formulas given in Table 11-210.

If more than one configurations are possible to reach a certain Bit rate, it is recommended to choose the configuration which allows the highest oscillator tolerance range.

CAN nodes with different clocks require different configurations to come to the same bit rate. The calculation of the propagation time in the CAN network, based on the nodes with the longest delay times, is done once for the whole network.

The CAN system's oscillator tolerance range is limited by the node with the lowest tolerance range.

The calculation may show that bus length or bit rate have to be decreased or that the oscillator frequencies' stability has to be increased in order to find a protocol compliant configuration of the CAN bit timing.

The resulting configuration is written into the bit timing register (DCAN_BTR):

[14-12] TSEG2 = Phase_Seg2 - 1

[11-8] TSEG1 = Phase_Seg1+ Prop_Seg - 1

[7-6] SJW = SynchronizationJumpWidth - 1

[5-0] BRP = Prescaler - 1

11.2.4.9.2.2 Example for Bit Timing Calculation

In this example, the frequency of CAN_CLK is 20 MHz, BRP is 2, the bit rate is 500 KBit/s.

Table 11-210. Example For Bit Timing

Parameter	Formula	Value	t_q
bit time (500 KBit/s)	1/bit rate, consists of $t_{Sync\text{-}Seg} + t_{TSeg1} + t_{TSeg2}$	2000 ns	20
delay of bus driver		280 ns	-
delay of receiver circuit		29 ns	-
delay of bus line (16 m)	16 × 5.5 ns/m	88 ns	-
t_q	BRP/CAN_CLK	100 ns	1
t_{Sync_Seg}	1 × t_q (fixed)	100 ns	1
t_{Prop_Seg}	INT (2×delays + 1) = 8 × t_q	800 ns	8
t_{Seg1}	$t_{Prop\_Seg} + t_{Phase\_Seg1}$	1400 ns	14

Table 11-210. Example For Bit Timing (continued)

Parameter	Formula	Value	t_q
t_{Seg2}	bit time - ($_{Sync\_Seg}$ + t_{Seg1})	500 ns	5
t_{SJWmax}	MIN (4 × t_q, t_{Phase_Seg1})	400 ns	4

In this example, the bit timing register DCAN_BTR is programmed to:

- BRP = 2 - 1 = 1
- BRPE = 0
- TSEG1 = 14 - 1 = 13 (0xC)
- TSEG2 = 5 - 1 = 4
- SJW = 4 - 1 = 3

11.2.4.10 Message Interface Register Sets

The interface register sets control the software read and write accesses to the message RAM. There are two interface registers sets for read/write access, IF1 and IF2 and one interface register set for read access only, IF3.

Due to the structure of the message RAM, it is not possible to change single bits or bytes of a message object. Instead, always a complete message object in the message RAM is accessed. Therefore the data transfer from the IF1/IF2 registers to the message RAM requires the message handler to perform a read-modify-write cycle: First those parts of the message object that are not to be changed are read from the message RAM into the interface register set, and after the update the whole content of the interface register set is written into the message object.

After the partial write of a message object, those parts of the interface register set that are not selected in DCAN_IF1CMD/DCAN_IF2CMD, will be set to the actual contents of the selected message object.

After the partial read of a message object, those parts of the interface register set that are not selected in DCAN_IF1CMD/DCAN_IF2CMD, will be left unchanged.

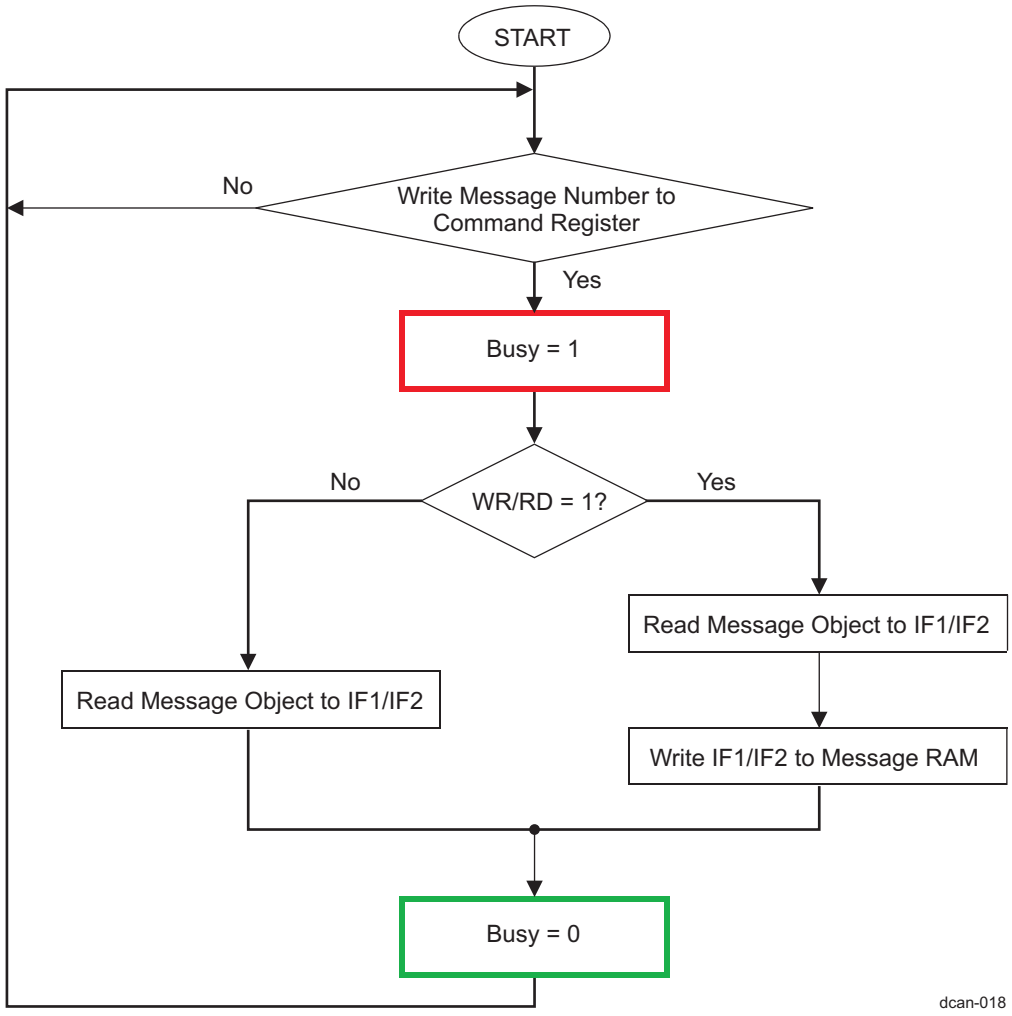
By buffering the data to be transferred, the interface register sets avoid conflicts between concurrent software accesses to the message RAM and CAN message reception and transmission. A complete message object (see Section 11.2.4.11.1, *Structure of Message Objects*) or parts of the message object may be transferred between the message RAM and the IF1/IF2 register set (see Section 11.2.5, *DCAN Registers*) in one single transfer. This transfer, performed in parallel on all selected parts of the message object, guarantees the data consistency of the CAN message.

11.2.4.10.1 Message Interface Register Sets 1 and 2

The IF1 and IF2 register sets control the data transfer to and from the message object. DCAN_IF1CMD/DCAN_IF2CMD address the desired message object in the message RAM and specifies whether a complete message object or only parts should be transferred. The data transfer is initiated by writing the message number to the bits [7-0] MESSAGE_NUMBER.

When the software initiates a data transfer between the IF1/IF2 registers and message RAM, the message handler sets the [15] BUSY bit in respective DCAN_IF1CMD/DCAN_IF2CMD to 1. After the transfer has completed, the BUSY bit is set back to 0 (see Figure 11-87).

Figure 11-87. Data Transfer Between IF1/IF2 Registers and Message RAM



11.2.4.10.2 IF3 Register Set

The IF3 register set can automatically be updated with received message objects without the need to initiate the transfer from message RAM by software. The intention of this feature of IF3 is to provide an interface for the DMA to read packets efficiently. The automatic update functionality can be programmed for each message object (DCAN_IF3UPD12 to DCAN_IF3UPD78).

All valid message objects in message RAM which are configured for automatic update, will be checked for active NewDat flags. If such a message object is found, it will be transferred to the IF3 register (if no previous DMA transfers are ongoing), controlled by IF3 Observation register (DCAN_IF3OBS). If more than one NewDat flag is active, the message object with the lowest number has the highest priority for automatic IF3 update.

The NewDat bit in the message object will be reset by a transfer to IF3.

If DCAN internal IF3 update is complete, a DMA request is generated. The DMA request stays active until first read access to one of the IF3 registers. The DMA functionality has to be enabled by setting bit DCAN_CTL[20] DE3 = 1.

NOTE: The IF3 register set cannot be used for transferring data into message objects.

11.2.4.11 Message RAM

The DCAN message RAM contains message objects. There are up to 64 message objects in the message RAM.

During normal operation, accesses to the message RAM are performed via the interface register sets, and the software cannot directly access the message RAM.

The interface register sets IF1 and IF2 provide indirect read/write access from the software to the message RAM. The IF1 and IF2 register sets can buffer control and user data to be transferred to and from the message objects.

The third interface register set IF3 can be configured to automatically receive control and user data from the message RAM when a message object has been updated after reception of a CAN message. The software does not need to initiate the transfer from message RAM to IF3 register set.

The message handler avoids potential conflicts between concurrent accesses to message RAM and CAN frame reception/transmission.

There are two modes where the message RAM can be directly accessed by the software:

- Debug/Suspend mode (see Section 11.2.4.11.4, *Message RAM Representation in Debug/Suspend Mode*)
- RAM Direct Access (RDA) mode (see Section 11.2.4.11.5, *Message RAM Representation in Direct Access Mode*).

CAUTION

Writes to the DCAN RAM must not be done while it is in low-power mode. If such writes occur, the behavior is undefined and RAM content is corrupted. It has to be ensured in software that the low-power mode is removed from the DCAN RAM before writing to it.

11.2.4.11.1 Structure of Message Objects

Table 11-211 shows the structure of a message object.

The highlighted fields are those parts of the message object which are represented in dedicated registers. For example, the transmit request flags of all message objects are represented in centralized transmit request registers.

Table 11-211. Structure of a Message Object

UMask	Msk[28-0]	MXtd	MDir	EoB	unused	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
MsgVal	ID[28-0]	Xtd	Dir	DLC[3-0]	Data 0	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7

Table 11-212. Message Object Field Descriptions

Field Name	Value	Description
MsgVal		Message valid
	0	The message object is ignored by the message handler.
	1	The message object is to be used by the message handler.
		Note: This bit may be kept at level '1' even when the identifier bits ID[28-0], the control bits Xtd, Dir, or the data length code are changed. It should be reset if the Messages Object is no longer required. The software must reset the MsgVal bit of all unused Messages Objects during the initialization before it resets the INIT bit in the DCAN_CTL register.
UMask		Use acceptance mask
	0	Mask bits (Msk[28-0], MXtd and MDir) are ignored and not used for acceptance filtering.
	1	Mask bits are used for acceptance filtering.
		Note: If the UMask bit is set to one, the message object's mask bits have to be programmed during initialization of the message object before MsgVal is set to one.

Table 11-212. Message Object Field Descriptions (continued)

Field Name	Value	Description
ID[28-0]		Message identifier
	ID[28-0]	29-bit (“extended”) identifier bits
	ID[28-18]	11-bit (“standard”) identifier bits
Msk[28-0]		Identifier mask
	0	The corresponding bit in the message identifier is not used for acceptance filtering (don’t care).
	1	The corresponding bit in the message identifier is used for acceptance filtering.
Xtd		Mask extended identifier
	0	The extended identifier bit (IDE) has no effect on the acceptance filtering.
	1	The extended identifier bit (IDE) is used for acceptance filtering.
		Note: When 11-bit (“standard”) Identifiers are used for a message object, the identifiers of received data frames are written into bits ID[28-18]. For acceptance filtering, only these bits together with mask bits Msk[28-18] are considered.
Dir		Message direction
	0	Direction = receive: On TxRqst, a remote frame with the identifier of this message object is transmitted. On reception of a data frame with matching identifier, the message is stored in this message object.
	1	Direction = transmit: On TxRqst, a data frame is transmitted. On reception of a remote frame with matching identifier, the TxRqst bit of this message object is set (if RmtEn = one).
MDir		Mask message direction
	0	The message direction bit (Dir) has no effect on the acceptance filtering.
	1	The message direction bit (Dir) is used for acceptance filtering.
EOB		End of block
	0	The message object is part of a FIFO Buffer block and is not the last message object of this FIFO Buffer block.
	1	The message object is a single message object or the last message object in a FIFO Buffer Block.
		Note: This bit is used to concatenate multiple message objects to build a FIFO Buffer. For single message objects (not belonging to a FIFO Buffer), this bit must always be set to one.
NewDat		New data
	0	No new data has been written into the data bytes of this message object by the message handler since the last time when this flag was cleared by the software.
	1	The message handler or the software has written new data into the data bytes of this message object.
MsgLst		Message lost (only valid for message objects with direction = receive)
	0	No message was lost since the last time when this bit was reset by the software.
	1	The message handler stored a new message into this message object when NewDat was still set, so the previous message has been overwritten.
RxIE		Receive interrupt enable
	0	IntPnd will not be triggered after the successful reception of a frame.
	1	IntPnd will be triggered after the successful reception of a frame.
TxIE		Transmit interrupt enable
	0	IntPnd will not be triggered after the successful transmission of a frame.
	1	IntPnd will be triggered after the successful transmission of a frame.
IntPnd		Interrupt pending
	0	This message object is not the source of an interrupt.
	1	This message object is the source of an interrupt. The interrupt Identifier in the interrupt register (DCAN_INT) will point to this message object if there is no other interrupt source with higher priority.
RmtEn		Remote enable
	0	At the reception of a remote frame, TxRqst is not changed.
	1	At the reception of a remote frame, TxRqst is set.
TxRqst		Transmit request
	0	This message object is not waiting for a transmission.
	1	The transmission of this message object is requested and is not yet done.

Table 11-212. Message Object Field Descriptions (continued)

Field Name	Value	Description
DLC[3-0]		Data length code
	0 - 8	Data frame has 0 - 8 data bytes.
	9 - 15	Data frame has 8 data bytes.
		Note: The data length code of a message object must be defined to the same value as in the corresponding objects with the same identifier at other nodes. When the message handler stores a data frame, it will write the DLC to the value given by the received message.
Data 0		1st data byte of a CAN data frame
Data 1		2nd data byte of a CAN data frame
Data 2		3rd data byte of a CAN data frame
Data 3		4th data byte of a CAN data frame
Data 4		5th data byte of a CAN data frame
Data 5		6th data byte of a CAN data frame
Data 6		7th data byte of a CAN data frame
Data 7		8th data byte of a CAN data frame
		Note: Byte Data 0 is the first data byte shifted into the shift register of the CAN core during a reception, byte Data 7 is the last. When the message handler stores a data frame, it will write all the eight data bytes into a message object. If the data length code is less than 8, the remaining bytes of the message object may be overwritten by undefined values.

11.2.4.11.2 Addressing Message Objects in RAM

The starting location of a particular message object in RAM is:
Message RAM base address + (message object number) × 20h

That is, message object 1 starts at offset 0020h; message object 2 starts at offset 0040h, and so forth.

NOTE: Because 0 is not a valid message object number, at offset 0000h is not located message object 0, but the last implemented: 64.

The base address for DCAN_0_DATA RAM is 0260 4000h and DCAN_1_DATA RAM is 0260 8000h.

Message object number 1 has the highest priority.

Table 11-213. Message RAM Addressing in Debug/Suspend and RDA Modes

Message Object Number	Offset	Word Number	Debug/Suspend Mode, see Section 11.2.4.11.4	RDA Mode, see Section 11.2.4.11.5
64 (last implemented)	0000h	1	Reserved	Data Bytes 4-7
	0004h	2	MXtd, MDir, Mask	Data Bytes 0-3
	0008h	3	Xtd, Dir, ID	ID[27-0], DLC
	000Ch	4	Ctrl	Mask, Xtd, Dir, ID[28]
	0010h	5	Data Bytes 3-0	Reserved, Ctrl, MXtd, MDir
	0014h	6	Data Bytes 7-4	-
1	0020h	1	Reserved	Data Bytes 4-7
	0024h	2	MXtd, MDir, Mask	Data Bytes 0-3
	0028h	3	Xtd, Dir, ID	ID[27-0], DLC
	002Ch	4	Ctrl	Mask, Xtd, Dir, ID[28]
	0030h	5	Data Bytes 3-0	Reserved, Ctrl, MXtd, MDir
	0034h	6	Data Bytes 7-4	-

Table 11-213. Message RAM Addressing in Debug/Suspend and RDA Modes (continued)

Message Object Number	Offset	Word Number	Debug/Suspend Mode, see Section 11.2.4.11.4	RDA Mode, see Section 11.2.4.11.5
2	0040h	1	Reserved	Data Bytes 4-7
	0044h	2	MXtd, MDir, Mask	Data Bytes 0-3
	0048h	3	Xtd, Dir, ID	ID[27-0], DLC
	004Ch	4	Ctrl	Mask, Xtd, Dir, ID[28]
	0050h	5	Data Bytes 3-0	Reserved, Ctrl, MXtd, MDir
	0054h	6	Data Bytes 7-4	-
…	…	…	…	…
63	07E0h	1	Reserved	Data Bytes 4-7
	07E4h	2	MXtd, MDir, Mask	Data Bytes 0-3
	07E8h	3	Xtd, Dir, ID	ID[27-0], DLC
	07ECh	4	Ctrl	Mask, Xtd, Dir, ID[28]
	07F0h	5	Data Bytes 3-0	Reserved, Ctrl, MXtd, MDir
	07F4h	6	Data Bytes 7-4	-

11.2.4.11.3 ECC RAM

On devices with SECDED implementation for the message RAM, the ECC bits are stored in a dedicated ECC RAM area which is memory mapped as follows:

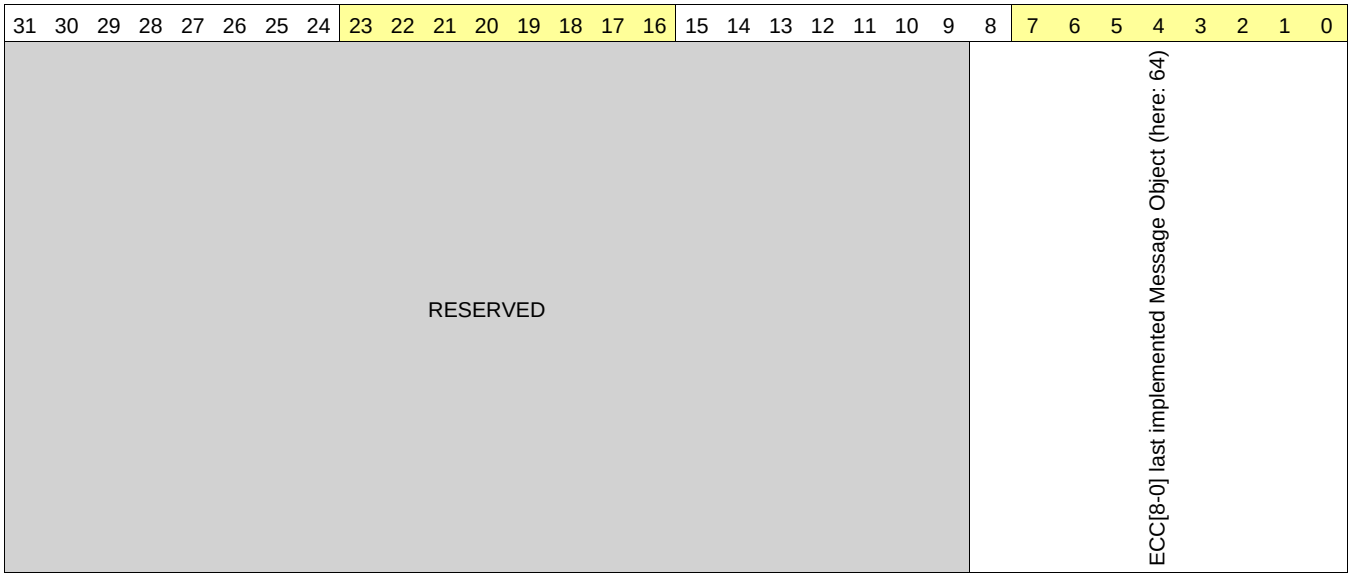
The location of the ECC bits for a particular message object in RAM is:

Message RAM base address + 1000h + (message object number) × 20h

NOTE: 0 is not a valid message object number. Therefore, at address 1000h, the ECC bits of the last implemented message object are located, that is, 64-th.

Table 11-214. ECC RAM Representation

Address	Msg RAM base + 0x1000



Address	Msg RAM base + 0x1020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								ECC[8-0] Message Object 1							

Address	Msg RAM base + 0x17E0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								ECC[8-0] Message Object 63							

As shown in Table 11-214 the ECC bits for the last implemented Message Object (here: 64) are located at offset 0x1000; the ECC bits for Message Object 1 are located at offset 0x1020, and the ECC bits for Message Object 63 are located at offset 0x17E0.

The ECC RAM is only memory mapped if SECDED diagnostic mode is enabled.

11.2.4.11.4 Message RAM Representation in Debug/Suspend Mode

In debug/suspend mode, the message RAM will be memory mapped. This allows the external debug unit to access the message RAM.

NOTE: During debug/suspend mode, the message RAM cannot be accessed via the IFx register sets.

Table 11-215. Message RAM Representation in Debug/Suspend Mode

Offset Within MO	0x00

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														Reserved for parity	

Offset Within MO	0x04

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MXtd	MDir	RESERVED	Msk[28-0]																												

Offset Within MO	0x08

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	Xtd	Dir	ID[28-0]																												

Offset Within MO	0x0C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																MsgLst	RESERVED	UMask	TxIE	RxTE	RmtEn	RESERVED	EOB	RESERVED				DLC[3-0]			

Offset Within MO	0x10

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_3								DATA_2								DATA_1								DATA_0							

Offset Within MO	0x14

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_7								DATA_6								DATA_5								DATA_4							

11.2.4.11.5 Message RAM Representation in Direct Access Mode

When the DCAN_TEST[9] RDA bit is set while the DCAN module is in test mode (DCAN_CTL[7] TEST = 1), the software has direct access to the message RAM. Due to the 32-bit bus structure, the RAM is split into word lines to support this feature. The software has access to one word line at a time only.

In RAM direct access mode, the RAM is represented by a continuous memory space within the address frame of the DCAN module, starting at the message RAM base address.

Note: During direct access mode, the message RAM cannot be accessed via the IFx register sets.

Any read or write to the RAM addresses for RAM Direct Access during normal operation mode (TEST bit or RDA bit not set) will be ignored.

Table 11-216. Message RAM Representation in RAM Direct Access Mode

Offset Within MO	0x00

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_4								DATA_5								DATA_6								DATA_7							

Offset Within MO	0x04

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_0								DATA_1								DATA_2								DATA_3							

Offset Within MO	0x08

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID[27-0]																												DLC[3-0]			

Offset Within MO	0x0C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Msk[28-0]																													Xtd	Dir	ID[28]

Offset Within MO	0x10

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																Reserved for parity				Unused				MsgLst	UMask	TxIE	RxTE	RmtEn	EOB	MXtd	MDir

NOTE: Writes to unused bits have no effect.

11.2.4.12 CAN Operation

After hardware reset, the DCAN_CTL[0] INIT is set and all CAN protocol functions are disabled. The CAN module must be initialized before operating it. Figure 11-88 illustrates the basic initialization flow for the CAN module.

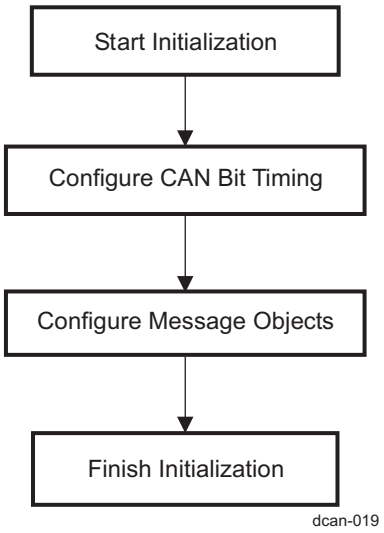
11.2.4.12.1 CAN Module Initialization

A general CAN module initialization would mean the following two critical steps:

1. Configuration of the CAN bit timing.
2. Configuration of message objects.

To initialize the CAN controller, the software has to set up the CAN bit timing and those message objects that have to be used for CAN communication. Message objects that are not needed, can be deactivated.

Figure 11-88. CAN Module General Initialization Flow



11.2.4.12.1.1 Configuration of CAN Bit Timing

The CAN module must be in initialization mode to configure the CAN bit timing.

For CAN bit timing software configuration flow, see Figure 11-89, *CAN Bit-Timing Configuration*.

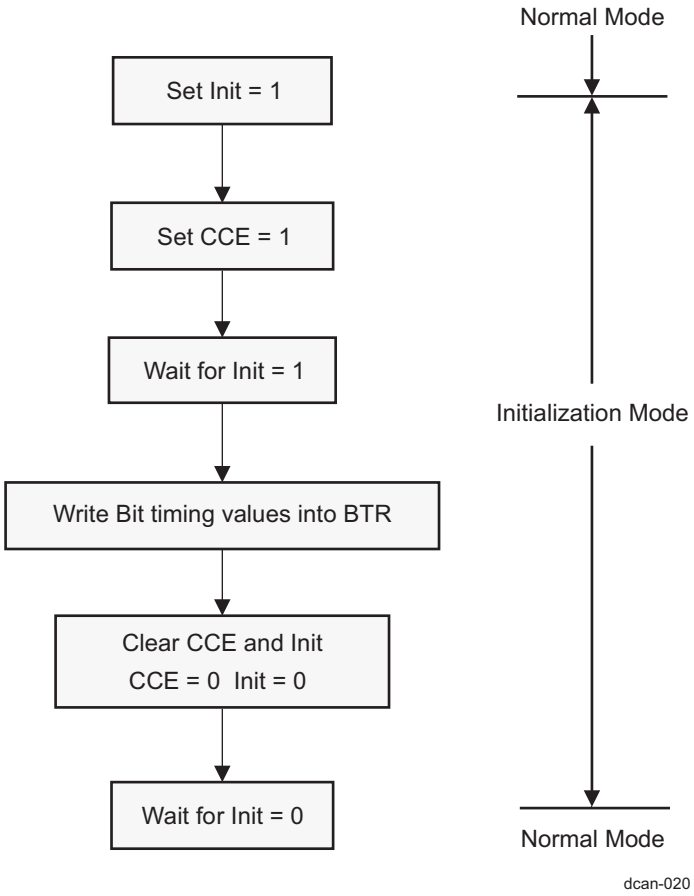
Step 1: Enter *initialization mode* (DCAN_CTL[0] INIT = 1).

While the INIT bit is set, the message transfer from and to the CAN bus is stopped, and the status of the CAN_TX output is recessive (high).

The CAN error counters are not updated. Setting the INIT bit does not change any other configuration register.

Also, note that the CAN module is in initialization mode on hardware reset and during Bus-Off.

Figure 11-89. CAN Bit-Timing Configuration



Step 2: Set the Configure Change Enable bit (DCAN_CTL[6] CCE = 1).

The access to the Bit Timing register (DCAN_BTR) for the configuration of the bit timing is enabled when both INIT = 1 and CCE = 1.

Step 3: Wait for the INIT bit to get set (=1). This would make sure that the module has entered Initialization mode.

Step 4: Write the bit timing values into DCAN_BTR. See Section 11.2.4.9.2, *DCAN Bit Timing Registers* for the values calculation for a given bit timing.

Step 5: Clear the CCE and INIT bits (=0).

Step 6: Wait for the INIT bit to clear (=0). This would ensure that the module has come out of Initialization mode.

Following these steps, the module comes to operation by synchronizing itself to the CAN bus, provided the DCAN_BTR is configured as per the CAN bus baud rate, although the message objects have to be configured before carrying out any communication.

NOTE: The module will not come out of the Initialization mode if any incorrect DCAN_BTR values are written in step 4.

NOTE: The required message objects should be configured as transmit or receive objects before the start of data transfer as explained in Section 11.2.4.12.1, *CAN Module Initialization*.

11.2.4.12.1.2 Configuration of Message Objects

The message objects can be configured only through the interface registers; the software does not have direct access to the message object (message RAM). For more details how to configuring the message objects, see Section 11.2.4.10, *Message Interface Register Sets*, for the interface register set (IFx) usage and see Section 11.2.4.11, *Message RAM* for the message object structure.

For more information regarding the procedure to configure the message objects, see Section 11.2.4.7, *Configuration of Message Objects Description*. All the message objects should be configured to particular identifiers or set to not valid before the message transfer is started. It is possible to change the configuration of message objects during normal operation (that is between data transfers).

NOTE: The message objects initialization is independent of the bit-timing configuration.

11.2.4.12.1.3 DCAN RAM Hardware Initialization

The memory hardware initialization for the DCAN module is enabled in the device BOOT_CFG module. Setting bit DCANx_RAMINIT_START to 1 (where x = 0 or 1), causes RAM initialization with zeros and sets ECC bits accordingly. Software must wait for the DCANx_RAMINIT_DONE bit to be set to ensure successful RAM initialization.

For more details on DCAN_RAMINIT register, see Section 5.1.4, *BOOT_CFG Registers*.

11.2.4.12.2 CAN Message Transfer (Normal Operation)

Once the DCAN is initialized and DCAN_CTL[0] INIT bit is reset (=0), the CAN core synchronizes itself to the CAN bus and is ready for message transfer as per the configured message objects.

The software may enable the interrupt lines (DCAN_CTL[1] IE0 and [17] IE1 = 1) at the same time when it clears [0] INIT and [6] CCE = 0. The status interrupts [3] EIE and [2] SIE may be enabled (=1) simultaneously.

The CAN communication can be carried out in any of the following two modes: interrupt and polling.

The DCAN_INT register points to those message objects with IntPnd = 1. It is updated even if the interrupt lines to the host processor are disabled (DCAN_CTL[1] IE0 and [17] IE1 = 0).

The software may poll all MessageObject's NewDat and TxRqst bits in parallel from the DCAN_NWDAT_X and the DCAN_TXRQ_X registers respectively. Polling can be made easier if all transmit objects are grouped at the low numbers and all receive objects are grouped at the high numbers.

Received messages are stored into their appropriate message objects if they pass acceptance filtering.

The whole message (including all arbitration bits, DLC and up to eight data bytes) is stored into the message object. As a consequence (for example, when the identifier mask is used), the arbitration bits which are masked to 'don't care' may change in the message object when a received message is stored.

The software may read or write each message at any time via the interface registers, as the message handler guarantees data consistency in case of concurrent accesses.

If a permanent message object (arbitration and control bits set up during configuration and leaving unchanged for multiple CAN transfers) exists for the message, it is possible to only update the data bytes.

If several transmit messages should be assigned to one message object, the whole message object has to be configured before the transmission of this message is requested.

The transmission of multiple message objects may be requested at the same time. They are subsequently transmitted, according to their internal priority.

Messages may be updated or set to not valid at any time, even if a requested transmission is still pending. However, the data bytes will be discarded if a message is updated before a pending transmission has started.

Depending on the configuration of the message object, a transmission may be automatically requested by the reception of a remote frame with a matching identifier.

11.2.4.12.2.1 Automatic Retransmission

According to the CAN Specification (ISO 11898), the DCAN provides a mechanism to automatically retransmit frames which have lost arbitration or have been disturbed by errors during transmission. The frame transmission service will not be confirmed to the user before the transmission is successfully completed.

By default, this automatic retransmission is enabled. It can be disabled by setting bit disable automatic retransmission (DCAN_CTL[5] DAR = 1). Further details to this mode are provided in Section 11.2.4.8.3, *Transmission of Messages in Event Driven CAN Communication*.

11.2.4.12.2.2 Auto-Bus-On

By default, after the DCAN has entered Bus-Off state, the software can start a Bus-Off-Recovery sequence by resetting the DCAN_CTL[0] INIT bit to 0. If this is not done, the module will stay in Bus-Off state.

The DCAN provides an automatic Auto-Bus-On feature which is enabled by bit DCAN_CTL[9] ABO. If set, the DCAN will automatically start the Bus-Off-Recovery sequence. The sequence can be delayed by a user-defined number of interface clock cycles which can be defined in the Auto-Bus-On Time register (DCAN_ABOTR).

NOTE: If the DCAN goes to Bus-Off state due to a massive occurrence of CAN bus errors, it stops all bus activities and automatically sets the INIT bit. Once the INIT bit has been reset by the software or due to the Auto-Bus-On feature, the device will wait for 129 occurrences of bus Idle (equal to 129 × 11 consecutive recessive bits) before resuming normal operation. At the end of the Bus-Off recovery sequence, the error counters will be reset.

11.2.4.12.3 Test Modes

The DCAN module provides several test modes which are mainly intended for production tests or self test.

For all test modes, the DCAN_CTL[7] TEST bit needs to be set to 1. This enables write access to the test register (DCAN_TEST).

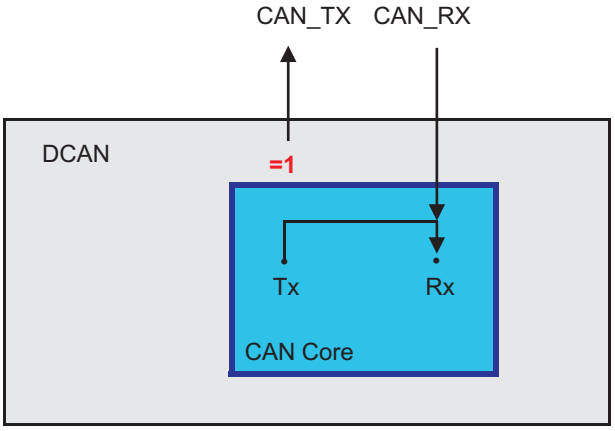
NOTE: It must be ensured by software that all message transfers are completed before entering test mode.

11.2.4.12.3.1 Silent Mode

The silent mode may be used to analyze the traffic on the CAN bus without affecting it by sending dominant bits (for example, acknowledge bit, overload flag, active error flag). The DCAN is still able to receive valid data frames and valid remote frames, but it will not send any dominant bits. However, these are internally routed to the CAN core.

Figure 11-90 shows the connection of signals CAN_TX and CAN_RX to the CAN core in silent mode. Silent mode can be activated by setting the DCAN_TEST[3] SILENT bit to 1. In ISO 11898-1, the silent mode is called the bus monitoring mode.

Figure 11-90. CAN Core in Silent Mode



11.2.4.12.3.2 Loopback Mode

The loopback mode is mainly intended for hardware self-test functions. In this mode, the CAN core uses internal feedback from Tx output to Rx input. Transmitted messages are treated as received messages, and can be stored into message objects if they pass acceptance filtering. The actual value of the CAN_RX input pin is disregarded by the CAN core. Transmitted messages can still be monitored at the CAN_TX pin.

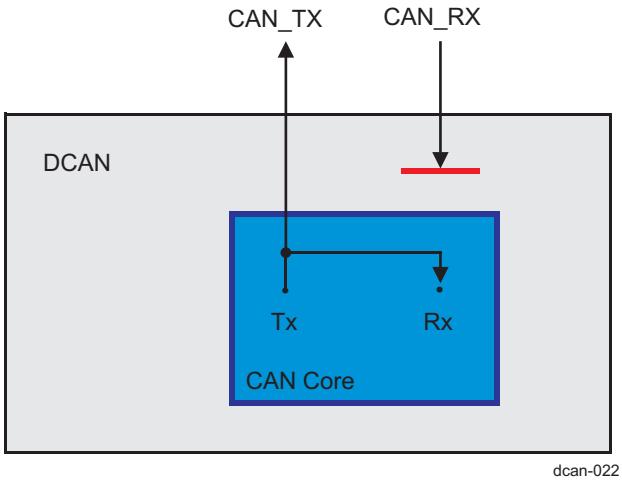
In order to be independent from external stimulation, the CAN core ignores acknowledge sampled in the acknowledge slot of a data/remote frame.

Figure 11-91 shows the connection of signals CAN_TX and CAN_RX to the CAN core in loopback mode.

Loopback mode can be activated by setting bit DCAN_TEST[4] LBACK to 1.

NOTE: In loopback mode, the signal path from CAN core to Tx pin, the Tx pin itself, and the signal path from Tx pin back to CAN core are disregarded. For including these into the testing, see Section 11.2.4.12.3.3, *External Loopback Mode*.

Figure 11-91. CAN Core in Loopback Mode



11.2.4.12.3.3 External Loopback Mode

The external loopback mode is similar to the loopback mode; however, it includes the signal path from CAN core to Tx pin, the Tx pin itself, and the signal path from Tx pin back to CAN core. When external loopback mode is selected, the input of the CAN core is connected to the input buffer of the Tx pin.

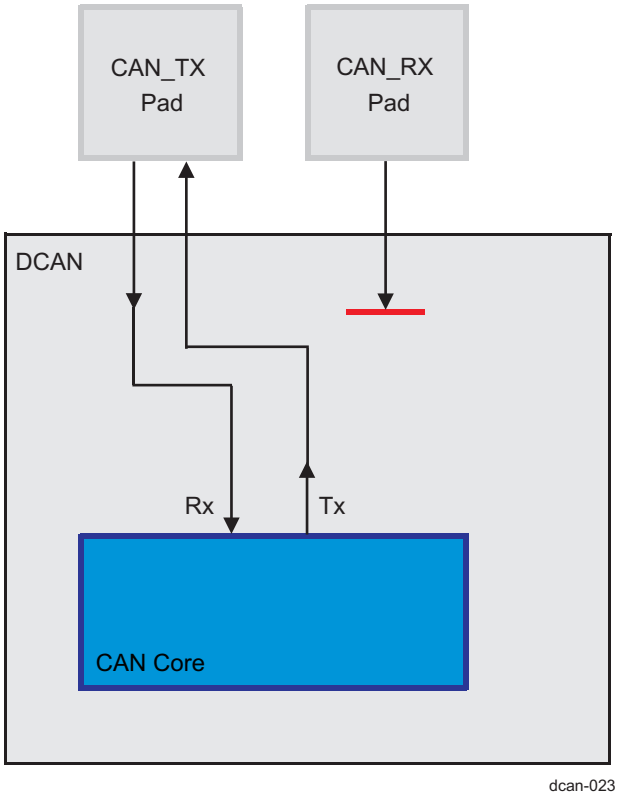
With this configuration, the Tx pin IO circuit can be tested.

External loopback mode can be activated by setting bit DCAN_TEST[8] EXL to 1.

Figure 11-92 shows the connection of signals CAN_TX and CAN_RX to the CAN core in external loopback mode.

NOTE: When loopback mode is active (LBACK bit set), the EXL bit will be ignored.

Figure 11-92. CAN Core in External Loopback Mode

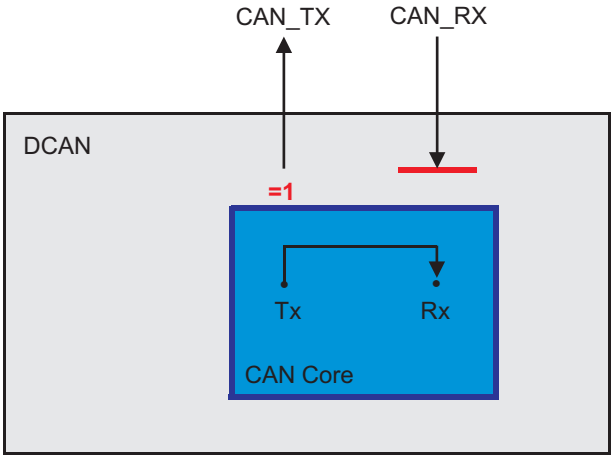


11.2.4.12.3.4 Loopback Mode Combined With Silent Mode

It is also possible to combine loopback mode and silent mode by setting bits DCAN_TEST[4] LBACK and [3] SILENT at the same time. This mode can be used for a "Hot Selftest", that is, the DCAN hardware can be tested without affecting the CAN network. In this mode, the CAN_RX pin is disconnected from the CAN core and no dominant bits will be sent on the CAN_TX pin.

Figure 11-93 shows the connection of the signals CAN_TX and CAN_RX to the CAN core in case of the combination of loopback mode with silent mode.

Figure 11-93. CAN Core in Loop Back Combined With Silent Mode



11.2.4.12.3.5 Software Control of CAN_TX Pin

Four output functions are available for the CAN transmit pin CAN_TX. In addition to its default function (serial data output), the CAN_TX pin can drive constant dominant or recessive values, or it can drive the CAN sample point signal to monitor the CAN core's bit timing.

Combined with the readable value of the CAN_RX pin, this function can be used to check the physical layer of the CAN bus.

The output mode of pin CAN_TX is selected by programming the DCAN_TEST[6-5] TX:

- 0x0: Normal operation, CAN_TX is controlled by the DCAN core
- 0x1: Sample point can be monitored at CAN_TX pin
- 0x2: CAN_TX pin drives a dominant value
- 0x3: CAN_TX pin drives a recessive value.

NOTE: The software control for the CAN_TX pin interferes with CAN protocol functions. For CAN message transfer or any of the test modes (loopback mode, external loopback mode or silent mode), the CAN_TX pin should operate in its default functionality.

11.2.5 DCAN Registers

Table 11-218 lists the memory-mapped registers for the DCAN. All register offset addresses not listed in Table 11-218 should be considered as reserved locations and the register contents should not be modified.

Table 11-217. DCAN Instances

Instance	Base Address
DCAN_0_DATA(1)	0260 4000h
DCAN_1_DATA(1)	0260 8000h
DCAN_0	0260 B200h
DCAN_1	0260 B400h

(1) Message RAM memory mapped space. See Section 11.2.4.11, *Message RAM* for more information.

NOTE: After hardware reset, the registers of the DCAN hold the values shown in the register descriptions.

Additionally, the Bus-Off state is reset and the CAN_TX pin is set to recessive (HIGH). The INIT bit in the DCAN_CTL is set to enable the software initialization. The DCAN will not influence the CAN bus until the software resets INIT to 0.

Table 11-218. DCAN Registers

Offset	Acronym	Register Name	DCAN_0_DATA Physical Address	DCAN_1_DATA Physical Address	DCAN_0 Physical Address	DCAN_1 Physical Address	Section
0h	DCAN_CTL	DCAN control register	0260 4000h	0260 8000h	0260 B200h	0260 B400h	Section 11.2.5.1
4h	DCAN_ES	Error and Status Register	0260 4004h	0260 8004h	0260 B204h	0260 B404h	Section 11.2.5.2
8h	DCAN_ERRC	Error Counter Register	0260 4008h	0260 8008h	0260 B208h	0260 B408h	Section 11.2.5.3
Ch	DCAN_BTR	Bit timing register	0260 400Ch	0260 800Ch	0260 B20Ch	0260 B40Ch	Section 11.2.5.4
10h	DCAN_INT	Interrupt register	0260 4010h	0260 8010h	0260 B210h	0260 B410h	Section 11.2.5.5
14h	DCAN_TEST	Test Register	0260 4014h	0260 8014h	0260 B214h	0260 B414h	Section 11.2.5.6
1Ch	DCAN_PERR	Parity Error Code Register	0260 401Ch	0260 801Ch	0260 B21Ch	0260 B41Ch	Section 11.2.5.7
20h	DCAN_REL	Core revision register	0260 4020h	0260 8020h	0260 B220h	0260 B420h	Section 11.2.5.8
24h	DCAN_ECCDIAG	ECC Diagnostic Register	0260 4024h	0260 8024h	0260 B224h	0260 B424h	Section 11.2.5.9
28h	DCAN_ECCDIAG_STAT	ECC Diagnostic Status Register	0260 4028h	0260 8028h	0260 B228h	0260 B428h	Section 11.2.5.10
2Ch	DCAN_ECC_CS	ECC Control and Status Register	0260 402Ch	0260 802Ch	0260 B22Ch	0260 B42Ch	Section 11.2.5.11
30h	DCAN_ECC_SERR	ECC Single Bit Error Code Register	0260 4030h	0260 8030h	0260 B230h	0260 B430h	Section 11.2.5.12
80h	DCAN_ABOTR	Auto-Bus-On Time Register	0260 4080h	0260 8080h	0260 B280h	0260 B480h	Section 11.2.5.13
84h	DCAN_TXRQ_X	Transmission Request X Register	0260 4084h	0260 8084h	0260 B284h	0260 B484h	Section 11.2.5.14
88h	DCAN_TXRQ12	Transmission Request Register	0260 4088h	0260 8088h	0260 B288h	0260 B488h	Section 11.2.5.15

Table 11-218. DCAN Registers (continued)

Offset	Acronym	Register Name	DCAN_0_DATA Physical Address	DCAN_1_DATA Physical Address	DCAN_0 Physical Address	DCAN_1 Physical Address	Section
8Ch	DCAN_TXRQ34	Transmission Request Register	0260 408Ch	0260 808Ch	0260 B28Ch	0260 B48Ch	Section 11.2.5.16
90h	DCAN_TXRQ56	Transmission Request Register	0260 4090h	0260 8090h	0260 B290h	0260 B490h	Section 11.2.5.17
94h	DCAN_TXRQ78	Transmission Request Register	0260 4094h	0260 8094h	0260 B294h	0260 B494h	Section 11.2.5.18
98h	DCAN_NWDAT_X	New Data X Register	0260 4098h	0260 8098h	0260 B298h	0260 B498h	Section 11.2.5.19
9Ch	DCAN_NWDAT12	New Data Register	0260 409Ch	0260 809Ch	0260 B29Ch	0260 B49Ch	Section 11.2.5.20
A0h	DCAN_NWDAT34	New Data Register	0260 40A0h	0260 80A0h	0260 B2A0h	0260 B4A0h	Section 11.2.5.21
A4h	DCAN_NWDAT56	New Data Register	0260 40A4h	0260 80A4h	0260 B2A4h	0260 B4A4h	Section 11.2.5.22
A8h	DCAN_NWDAT78	New Data Register	0260 40A8h	0260 80A8h	0260 B2A8h	0260 B4A8h	Section 11.2.5.23
ACh	DCAN_INTPND_X	Interrupt Pending X Register	0260 40ACh	0260 80ACh	0260 B2ACh	0260 B4ACh	Section 11.2.5.24
B0h	DCAN_INTPND12	Interrupt Pending Register	0260 40B0h	0260 80B0h	0260 B2B0h	0260 B4B0h	Section 11.2.5.25
B4h	DCAN_INTPND34	Interrupt Pending Register	0260 40B4h	0260 80B4h	0260 B2B4h	0260 B4B4h	Section 11.2.5.26
B8h	DCAN_INTPND56	Interrupt Pending Register	0260 40B8h	0260 80B8h	0260 B2B8h	0260 B4B8h	Section 11.2.5.27
BCh	DCAN_INTPND78	Interrupt Pending Register	0260 40BCh	0260 80BCh	0260 B2BCh	0260 B4BCh	Section 11.2.5.28
C0h	DCAN_MSGVAL_X	Message Valid X Register	0260 40C0h	0260 80C0h	0260 B2C0h	0260 B4C0h	Section 11.2.5.29
C4h	DCAN_MSGVAL12	Message Valid Register	0260 40C4h	0260 80C4h	0260 B2C4h	0260 B4C4h	Section 11.2.5.30
C8h	DCAN_MSGVAL34	Message Valid Register	0260 40C8h	0260 80C8h	0260 B2C8h	0260 B4C8h	Section 11.2.5.31
CCh	DCAN_MSGVAL56	Message Valid Register	0260 40CCh	0260 80CCh	0260 B2CCh	0260 B4CCh	Section 11.2.5.32
D0h	DCAN_MSGVAL78	Message Valid Register	0260 40D0h	0260 80D0h	0260 B2D0h	0260 B4D0h	Section 11.2.5.33
D8h	DCAN_INTMUX12	Interrupt Multiplexer Register	0260 40D8h	0260 80D8h	0260 B2D8h	0260 B4D8h	Section 11.2.5.34
DCh	DCAN_INTMUX34	Interrupt Multiplexer Register	0260 40DCh	0260 80DCh	0260 B2DCh	0260 B4DCh	Section 11.2.5.35
E0h	DCAN_INTMUX56	Interrupt Multiplexer Register	0260 40E0h	0260 80E0h	0260 B2E0h	0260 B4E0h	Section 11.2.5.36
E4h	DCAN_INTMUX78	Interrupt Multiplexer Register	0260 40E4h	0260 80E4h	0260 B2E4h	0260 B4E4h	Section 11.2.5.37
100h	DCAN_IF1CMD	IF1 Command Register	0260 4100h	0260 8100h	0260 B300h	0260 B500h	Section 11.2.5.38
104h	DCAN_IF1MSK	IF1 Mask Register	0260 4104h	0260 8104h	0260 B304h	0260 B504h	Section 11.2.5.39
108h	DCAN_IF1ARB	IF1 arbitration register	0260 4108h	0260 8108h	0260 B308h	0260 B508h	Section 11.2.5.40
10Ch	DCAN_IF1MCTL	IF1 Message Control Register	0260 410Ch	0260 810Ch	0260 B30Ch	0260 B50Ch	Section 11.2.5.41
110h	DCAN_IF1DATA	IF1 Data A Register	0260 4110h	0260 8110h	0260 B310h	0260 B510h	Section 11.2.5.42

Table 11-218. DCAN Registers (continued)

Offset	Acronym	Register Name	DCAN_0_DATA Physical Address	DCAN_1_DATA Physical Address	DCAN_0 Physical Address	DCAN_1 Physical Address	Section
114h	DCAN_IF1DATB	IF1 Data B Register	0260 4114h	0260 8114h	0260 B314h	0260 B514h	Section 11.2.5.43
120h	DCAN_IF2CMD	IF2 Command Register	0260 4120h	0260 8120h	0260 B320h	0260 B520h	Section 11.2.5.44
124h	DCAN_IF2MSK	IF2 Mask Register	0260 4124h	0260 8124h	0260 B324h	0260 B524h	Section 11.2.5.45
128h	DCAN_IF2ARB	IF2 arbitration register	0260 4128h	0260 8128h	0260 B328h	0260 B528h	Section 11.2.5.46
12Ch	DCAN_IF2MCTL	IF2 Message Control Register	0260 412Ch	0260 812Ch	0260 B32Ch	0260 B52Ch	Section 11.2.5.47
130h	DCAN_IF2DATA	IF2 Data A Register	0260 4130h	0260 8130h	0260 B330h	0260 B530h	Section 11.2.5.48
134h	DCAN_IF2DATB	IF2 Data B Register	0260 4134h	0260 8134h	0260 B334h	0260 B534h	Section 11.2.5.49
140h	DCAN_IF3OBS	IF3 Observation Register	0260 4140h	0260 8140h	0260 B340h	0260 B540h	Section 11.2.5.50
144h	DCAN_IF3MSK	IF3 Mask Register	0260 4144h	0260 8144h	0260 B344h	0260 B544h	Section 11.2.5.51
148h	DCAN_IF3ARB	IF3 Arbitration Register	0260 4148h	0260 8148h	0260 B348h	0260 B548h	Section 11.2.5.52
14Ch	DCAN_IF3MCTL	IF3 Message Control Register	0260 414Ch	0260 814Ch	0260 B34Ch	0260 B54Ch	Section 11.2.5.53
150h	DCAN_IF3DATA	IF3 Data A	0260 4150h	0260 8150h	0260 B350h	0260 B550h	Section 11.2.5.54
154h	DCAN_IF3DATB	IF3 Data B	0260 4154h	0260 8154h	0260 B354h	0260 B554h	Section 11.2.5.55
160h	DCAN_IF3UPD12	Update Enable Register	0260 4160h	0260 8160h	0260 B360h	0260 B560h	Section 11.2.5.56
164h	DCAN_IF3UPD34	Update Enable Register	0260 4164h	0260 8164h	0260 B364h	0260 B564h	Section 11.2.5.57
168h	DCAN_IF3UPD56	Update Enable Register	0260 4168h	0260 8168h	0260 B368h	0260 B568h	Section 11.2.5.58
16Ch	DCAN_IF3UPD78	Update Enable Register	0260 416Ch	0260 816Ch	0260 B36Ch	0260 B56Ch	Section 11.2.5.59
1E0h	RESERVED		0260 41E0h	0260 81E0h	0260 B3E0h	0260 B5E0h	
1E4h	RESERVED		0260 41E4h	0260 81E4h	0260 B3E4h	0260 B5E4h	

11.2.5.1 DCAN_CTL Register (Offset = 0h) [reset = 1401h]

DCAN_CTL is shown in Figure 11-94 and described in Table 11-220.

DCAN control register

NOTE: The Bus-Off recovery sequence (refer to CAN specification) cannot be shortened by setting or resetting INIT bit. If the module goes Bus-Off, it will automatically set the INIT bit and stop all bus activities. When the INIT bit is cleared by the application again, the module will then wait for 129 occurrences of Bus Idle (129 - 11 consecutive recessive bits) before resuming normal operation. At the end of the bus-off recovery sequence, the error counters will be reset. After the INIT bit is reset, each time when a sequence of 11 recessive bits is monitored, a Bit0 error code is written to DCAN_ES, enabling the software to check whether the CAN bus is stuck at dominant or continuously disturbed, and to monitor the proceeding of the bus-off recovery sequence.

Table 11-219. DCAN_CTL Instances

Instance	Physical Address
DCAN_0_DATA	0260 4000h
DCAN_1_DATA	0260 8000h
DCAN_0	0260 B200h
DCAN_1	0260 B400h

Figure 11-94. DCAN_CTL Register

31	30	29	28	27	26	25	24
RESERVED						RESERVED	PDR
R-0h						R-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED			DE3	DE2	DE1	IE1	INITDBG
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
SWR	RESERVED	PMD				ABO	IDS
R/W-0h	R-0h	R/W-5h				R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TEST	CCE	DAR	RESERVED	EIE	SIE	IE0	INIT
R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-220. DCAN_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
25	RESERVED	R	0h	
24	PDR	R/W	0h	Request for local low power-down mode 0h (R/W) = No application request for local low power-down mode. If the application has cleared this bit while DCAN in local power-down mode, also the INIT bit has to be cleared. 1h (R/W) = Local power-down mode has been requested by application. The DCAN will acknowledge the local power-down mode by setting bit PDA in the DCAN_ES register. The local clocks will be turned off by DCAN internal logic (Additional information can be found in Local Power-Down Mode).
23-21	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
20	DE3	R/W	0h	Enable DMA request line for IF3. Note: A pending DMA request for IF3 remains active until first access to one of the IF3 registers. 0h (R/W) = Disabled 1h (R/W) = Enabled

Table 11-220. DCAN_CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	DE2	R/W	0h	Enable DMA request line for IF2. Note: A pending DMA request for IF2 remains active until first access to one of the IF2 registers. 0h (R/W) = Disabled 1h (R/W) = Enabled
18	DE1	R/W	0h	Enable DMA request line for IF1. Note: A pending DMA request for IF1 remains active until first access to one of the IF1 registers. 0h (R/W) = Disabled 1h (R/W) = Enabled
17	IE1	R/W	0h	Interrupt line 1 enable 0h (R/W) = Disabled - Module interrupt INT1 is always low. 1h (R/W) = Enabled - interrupts will assert line INT1 to one; line remains active until pending interrupts are processed.
16	INITDBG	R/W	0h	Internal init state while debug access 0h (R/W) = Not in debug mode, or debug mode requested but not entered. 1h (R/W) = Debug mode requested and internally entered; the DCAN is ready for debug accesses.
15	SWR	R/W	0h	Software reset enable. Note: To execute software reset, the following procedure is necessary: 0h (R/W) = Normal Operation 1h (R/W) = Module is forced to reset state. This bit will automatically get cleared after execution of software reset after one DCAN_i_VBUS_CLK (i = 0 or 1) clock cycle.
14	RESERVED	R	0h	This bit is always read as 0. Writes have no effect.
13-10	PMD	R/W	5h	ECC on/offOthers: function enabled. 5h (R/W) = function disabled
9	ABO	R/W	0h	Auto-Bus-On enable 0h (R/W) = The Auto-Bus-On feature is disabled 1h (R/W) = The Auto-Bus-On feature is enabled
8	IDS	R/W	0h	Interruption debug support enable 0h (R/W) = When Debug/Suspend mode is requested, DCAN will wait for a started transmission or reception to be completed before entering Debug/Suspend mode 1h (R/W) = When Debug/Suspend mode is requested, DCAN will interrupt any transmission or reception, and enter Debug/Suspend mode immediately.
7	TEST	R/W	0h	Test mode enable 0h (R/W) = Normal Operation 1h (R/W) = Test Mode
6	CCE	R/W	0h	Configuration change enable 0h (R/W) = The software has no write access to the configuration registers. 1h (R/W) = The software has write access to the configuration registers (when INIT bit is set).
5	DAR	R/W	0h	Disable automatic retransmission 0h (R/W) = Automatic retransmission of not successful messages enabled. 1h (R/W) = Automatic retransmission disabled.
4	RESERVED	R	0h	This bit is always read as 0. Writes have no effect.
3	EIE	R/W	0h	Error interrupt enable 0h (R/W) = Disabled - PER, BOFF and EWARN bits can not generate an interrupt. 1h (R/W) = Enabled - PER, BOFF and EWARN bits can generate an interrupt at INT0 line and affect the interrupt register.

Table 11-220. DCAN_CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	SIE	R/W	0h	Status change interrupt enable 0h (R/W) = Disabled - WAKEUPPND, RXOK, TXOK and LEC bits can not generate an interrupt. 1h (R/W) = Enabled - WAKEUPPND, RXOK, TXOK and LEC can generate an interrupt at INT0 line and affect the interrupt register.
1	IE0	R/W	0h	Interrupt line 0 enable 0h (R/W) = Disabled - Module interrupt INT0 is always low. 1h (R/W) = Enabled - interrupts will assert line INT0 to one; line remains active until pending interrupts are processed.
0	INIT	R/W	1h	Initialization 0h (R/W) = Normal operation 1h (R/W) = Initialization mode is entered

Table 11-221. Register Call Summary for DCAN_CTL

DCAN Functional Description

- SECDED Mechanism: [0]
- Auto-Bus-On: [0][1]
- Status Change Interrupts: [0][1]
- Test Modes: [0]
- SECDED Testing: [0]
- Transmission of Messages in Event Driven CAN Communication: [0]
- IF3 Register Set: [0]
- CAN Message Transfer (Normal Operation): [0][1][2]
- Automatic Retransmission: [0]
- Message RAM Representation in Direct Access Mode: [0]
- Error Interrupts: [0][1]
- DMA Functionality: [0]
- Debug/Suspend Mode: [0][1]
- Interrupt Functionality: [0]
- Structure of Message Objects: [0]
- Entering Local Power-Down Mode: [0][1]
- Configuration of CAN Bit Timing: [0][1]
- CAN Operation: [0]

DCAN Registers

- DCAN_IF3ARB Register (Offset = 148h) [reset = 0h]: [0]
- DCAN_IF2ARB Register (Offset = 128h) [reset = 0h]: [0]
- DCAN Registers: [0][1]
- DCAN_INTMUX12 Register (Offset = D8h) [reset = 0h]: [0]
- DCAN_CTL Register (Offset = 0h) [reset = 1401h]: [0]
- DCAN_IF1ARB Register (Offset = 108h) [reset = 0h]: [0]
- DCAN_ABOTR Register (Offset = 80h) [reset = 0h]: [0][1]
- DCAN_TEST Register (Offset = 14h) [reset = 0h]: [0]
- DCAN_ES Register (Offset = 4h) [reset = 7h]: [0][1]
- DCAN_BTR Register (Offset = Ch) [reset = 2301h]: [0]

11.2.5.2 DCAN_ES Register (Offset = 4h) [reset = 7h]

DCAN_ES is shown in Figure 11-95 and described in Table 11-223.

Error and Status Register
Interrupts are generated by bits PER, BOFF and EWARN (if EIE bit in DCAN_CTL is 1) and by bits WAKEUPPND, RXOK, TXOK, and LEC (if SIE bit in DCAN_CTL is 1). A change of bit EPASS will not generate an interrupt.
Reading the DCAN_ES clears the WAKEUPPND, PER, RXOK and TXOK bits and set the LEC to value '7.' Additionally, the status interrupt value (0x8000) in the DCAN_INT will be replaced by the next lower priority interrupt value.
For debug support, the auto clear functionality of DCAN_ES (clear of status flags by read) is disabled when in debug/suspend mode.

Table 11-222. DCAN_ES Instances

Instance	Physical Address
DCAN_0_DATA	0260 4004h
DCAN_1_DATA	0260 8004h
DCAN_0	0260 B204h
DCAN_1	0260 B404h

Figure 11-95. DCAN_ES Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					PDA	WAKEUPPND	PER
R-0h					R-0h	R-0h	R/W-0h
7	6	5	4	3	2	1	0
BOFF	EWARN	EPASS	RXOK	TXOK	LEC		
R-0h	R-0h	R-0h	R-0h	R-0h	R-7h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-223. DCAN_ES Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
10	PDA	R	0h	Local power-down mode acknowledge 0h (R) = DCAN is not in local power-down mode. 1h (R) = Application request for setting DCAN to local power-down mode was successful. DCAN is in local power-down mode.
9	WAKEUPPND	R	0h	Wake up pending. This bit can be used by the software to identify the DCAN as the source to wake up the system. This bit will be reset if DCAN_ES is read. 0h (R) = No Wake Up is requested by DCAN. 1h (R) = DCAN has initiated a wake up of the system due to dominant CAN bus while module power down.

Table 11-223. DCAN_ES Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	PER	R/W	0h	Single/Double bit error detected. This bit will be set on double bit errors and additionally on single bit errors if single bit error correction is disabled with the ECCMODE bitfield in DCAN_ECC_CS. 0h (R/W) = No effect 1h (R/W) = End of interrupt (EOI) for single/double error on DCAN_ERR_INT interrupt line
7	BOFF	R	0h	Bus-Off state 0h (R) = The CAN module is not bus-off state. 1h (R) = The CAN module is in bus-off state.
6	EWARN	R	0h	Warning state 0h (R) = Both error counters are below the error warning limit of 96. 1h (R) = At least one of the error counters has reached the error warning limit of 96.
5	EPASS	R	0h	Error passive state 0h (R) = On CAN Bus error, the DCAN could send active error frames. 1h (R) = The CAN core is in the error passive state as defined in the CAN Specification.
4	RXOK	R	0h	Received a message successfully. This bit will be reset if DCAN_ES register is read. 0h (R) = No message has been successfully received since the last time when this bit was read by the software. This bit is never reset by DCAN internal events. 1h (R) = A message has been successfully received since the last time when this bit was reset by a read access of the software (independent of the result of acceptance filtering).
3	TXOK	R	0h	Transmitted a message successfully. This bit will be reset if DCAN_ES register is read. 0h (R) = No message has been successfully transmitted since the last time when this bit was read by the software. This bit is never reset by DCAN internal events. 1h (R) = A message has been successfully transmitted (error free and acknowledged by at least one other node) since the last time when this bit was reset by a read access of the software.

Table 11-223. DCAN_ES Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	LEC	R	7h	Last error code. The LEC field indicates the type of the last error on the CAN bus. This field will be cleared to '0' when a message has been transferred (reception or transmission) without error. 0h (R) = No error 1h (R) = Stuff error: More than five equal bits in a row have been detected in a part of a received message where this is not allowed. 2h (R) = Form error: A fixed format part of a received frame has the wrong format. 3h (R) = Ack error: The message this CAN core transmitted was not acknowledged by another node. 4h (R) = Bit1 error: During the transmission of a message (with the exception of the arbitration field), the device wanted to send a recessive level (bit of logical value '1'), but the monitored bus value was dominant. 5h (R) = Bit0 error: During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device wanted to send a dominant level (logical value '0'), but the monitored bus level was recessive. During Bus-Off recovery, this status is set each time a sequence of 11 recessive bits has been monitored. This enables the software to monitor the proceeding of the Bus-Off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed). 6h (R) = CRC error: In a received message, the CRC check sum was incorrect. (CRC received for an incoming message does not match the calculated CRC for the received data). 7h (R) = No CAN bus event was detected since the last time the software read DCAN_ES. Any read access to DCAN_ES re-initializes the LEC to value '7.'

Table 11-224. Register Call Summary for DCAN_ES

DCAN Functional Description • Error Interrupts: [0] • Debug/Suspend Mode: [0] • Interrupt Functionality: [0][1] • Status Change Interrupts: [0][1] • Behavior on Double Bit Error: [0] • Behavior on Single Bit Error: [0] • Entering Local Power-Down Mode: [0]
DCAN Registers • DCAN_CTL Register (Offset = 0h) [reset = 1401h]: [0][1] • DCAN_INT Register (Offset = 10h) [reset = 0h]: [0] • DCAN_ES Register (Offset = 4h) [reset = 7h]: [0][1][2][3][4][5][6][7] • DCAN Registers: [0] • DCAN_PERR Register (Offset = 1Ch) [reset = 0h]: [0][1]

11.2.5.3 DCAN_ERRC Register (Offset = 8h) [reset = 0h]

DCAN_ERRC is shown in Figure 11-96 and described in Table 11-226.

Error Counter Register

Table 11-225. DCAN_ERRC Instances

Instance	Physical Address
DCAN_0_DATA	0260 4008h
DCAN_1_DATA	0260 8008h
DCAN_0	0260 B208h
DCAN_1	0260 B408h

Figure 11-96. DCAN_ERRC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RP	REC							TEC							
R-0h	R-0h							R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-226. DCAN_ERRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
15	RP	R	0h	Receive error passive 0h (R) = The receive error counter is below the error passive level. 1h (R) = The receive error counter has reached the error passive level as defined in the CAN specification.
14-8	REC	R	0h	Receive error counter. Actual state of the receive error counter
7-0	TEC	R	0h	Transmit error counter. Actual state of the transmit error counter

Table 11-227. Register Call Summary for DCAN_ERRC

DCAN Registers
- DCAN_ERRC Register (Offset = 8h) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.4 DCAN_BTR Register (Offset = Ch) [reset = 2301h]

DCAN_BTR is shown in Figure 11-97 and described in Table 11-229.

Bit timing register
This register is only writable if CCE and INIT bits in the DCAN_CTL are set.
The CAN bit time may be programmed in the range of 8 to 25 time quanta
The CAN time quantum may be programmed in the range of 1 to 1024 CAN_CLK periods.

Table 11-228. DCAN_BTR Instances

Instance	Physical Address
DCAN_0_DATA	0260 400Ch
DCAN_1_DATA	0260 800Ch
DCAN_0	0260 B20Ch
DCAN_1	0260 B40Ch

Figure 11-97. DCAN_BTR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				BRPE			
R-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED	TSEG2			TSEG1			
R-0h	R/W-2h			R/W-3h			
7	6	5	4	3	2	1	0
SJW		BRP					
R/W-0h		R/W-1h					

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-229. DCAN_BTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
19-16	BRPE	R/W	0h	Baud rate prescaler extension.Valid programmed values are 0 to 15. By programming BRPE the baud rate prescaler can be extended to values up to 1024.
15	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
14-12	TSEG2	R/W	2h	Time segment after the sample pointValid programmed values are 0 to 7. The actual TSeg2 value which is interpreted for the bit timing will be the programmed TSeg2 value + 1.
11-8	TSEG1	R/W	3h	Time segment before the sample pointValid programmed values are 1 to15. The actual TSeg1 value interpreted for the bit timing will be the programmed TSeg1 value + 1.
7-6	SJW	R/W	0h	Synchronization Jump WidthValid programmed values are 0 to 3. The actual SJW value interpreted for the synchronization will be the programmed SJW value + 1.
5-0	BRP	R/W	1h	Baud rate prescalerValue by which the CAN_CLK frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid programmed values are 0 to 63. The actual BRP value interpreted for the bit timing will be the programmed BRP value + 1.

Table 11-230. Register Call Summary for DCAN_BTR

DCAN Functional Description • Calculation of the Bit Timing Parameters: [0] • Example for Bit Timing Calculation: [0] • DCAN Bit Timing Registers: [0][1][2][3] • Configuration of CAN Bit Timing: [0][1][2][3]
DCAN Registers • DCAN Registers: [0] • DCAN_BTR Register (Offset = Ch) [reset = 2301h]: [0]

11.2.5.5 DCAN_INT Register (Offset = 10h) [reset = 0h]

DCAN_INT is shown in Figure 11-98 and described in Table 11-232.

Interrupt register

Table 11-231. DCAN_INT Instances

Instance	Physical Address
DCAN_0_DATA	0260 4010h
DCAN_1_DATA	0260 8010h
DCAN_0	0260 B210h
DCAN_1	0260 B410h

Figure 11-98. DCAN_INT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								INT1ID								INT0ID															
R-0h								R-0h								R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-232. DCAN_INT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
23-16	INT1ID	R	0h	Interrupt 1 Identifier (indicates the message object with the highest pending interrupt)0x01-0x80: Number of message object which caused the interrupt. 0x81-0xFF: Unused. If several interrupts are pending,DCAN_INT will point to the pending interrupt with the highest priority. The INT1 interrupt line remains active until INT1ID reaches value 0 (the cause of the interrupt is reset) or until IE1 is cleared. A message interrupt is cleared by clearing the message object's IntPnd bit. Among the message interrupts, the message object's interrupt priority decreases with increasing message number. 0h (R) = No interrupt is pending
15-0	INT0ID	R	0h	Interrupt Identifier (the number here indicates the source of the interrupt)0x0001-0x0080: Number of message object which caused the interrupt. 0x0081-0x7FFF: Unused. 0x8001-0xFFFF: Unused. If several interrupts are pending,DCAN_INTwill point to the pending interrupt with the highest priority. The INT0 interrupt line remains active until INT0ID reaches value 0 (the cause of the interrupt is reset) or until IE0 is cleared. The Status interrupt has the highest priority. Among the message interrupts, the message object's interrupt priority decreases with increasing message number. 0h (R) = No interrupt is pending 8000h (R) = DCAN_ES value is not 0x07.

Table 11-233. Register Call Summary for DCAN_INT

DCAN Functional Description
• Interrupt Functionality: [0][1][2] • CAN Message Transfer (Normal Operation): [0] • Reception of Data Frames: [0] • Structure of Message Objects: [0]

Table 11-233. Register Call Summary for DCAN_INT (continued)

DCAN Registers
• DCAN_IF2MCTL Register (Offset = 12Ch) [reset = 0h]: [0] • DCAN_INTMUX78 Register (Offset = E4h) [reset = 0h]: [0] • DCAN_IF3MCTL Register (Offset = 14Ch) [reset = 0h]: [0] • DCAN Registers: [0] • DCAN_INTMUX34 Register (Offset = DCh) [reset = 0h]: [0] • DCAN_INT Register (Offset = 10h) [reset = 0h]: [0][1] • DCAN_INTMUX56 Register (Offset = E0h) [reset = 0h]: [0] • DCAN_IF1MCTL Register (Offset = 10Ch) [reset = 0h]: [0] • DCAN_ES Register (Offset = 4h) [reset = 7h]: [0] • DCAN_INTMUX12 Register (Offset = D8h) [reset = 0h]: [0]

11.2.5.6 DCAN_TEST Register (Offset = 14h) [reset = 0h]

DCAN_TEST is shown in Figure 11-99 and described in Table 11-235.

Test Register
For all test modes, the TEST bit in DCAN_CTL control register needs to be set to 1. If TEST bit is set, the RDA, EXL, TX1, TX0, LBACK and SILENT bits are writable. Bit RX monitors the state of pin CAN_RX and therefore is only readable. All test register functions are disabled when TEST bit is cleared.

Table 11-234. DCAN_TEST Instances

Instance	Physical Address
DCAN_0_DATA	0260 4014h
DCAN_1_DATA	0260 8014h
DCAN_0	0260 B214h
DCAN_1	0260 B414h

Figure 11-99. DCAN_TEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						RDA	EXL
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RX	TX		LBACK	SILENT	RESERVED		
R--h	R/W-0h		R/W-0h	R/W-0h	R-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-235. DCAN_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
9	RDA	R/W	0h	RAM direct access enable 0h (R/W) = Normal operation 1h (R/W) = Direct access to the RAM is enabled while in test mode
8	EXL	R/W	0h	External loopback mode. When the internal loop-back mode is active (bit LBACK is set), bit EXL will be ignored. 0h (R/W) = Disabled 1h (R/W) = Enabled
7	RX	R	-h	Receive pin. Monitors the actual value of the CAN_RX pin 0h (R) = The CAN bus is dominant 1h (R) = The CAN bus is recessive
6-5	TX	R/W	0h	Control of CAN_TX pin. Setting Tx[1:0] other than '00' will disturb message transfer. 0h (R/W) = Normal operation, CAN_TX is controlled by the CAN core. 1h (R/W) = Sample point can be monitored at CAN_TX pin. 2h (R/W) = CAN_TX pin drives a dominant value. 3h (R/W) = CAN_TX pin drives a recessive value.

Table 11-235. DCAN_TEST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	LBACK	R/W	0h	Loopback mode. When the internal loop-back mode is active (bit LBACK is set), bit EXL will be ignored. 0h (R/W) = Disabled 1h (R/W) = Enabled
3	SILENT	R/W	0h	Silent mode 0h (R/W) = Disabled 1h (R/W) = Enabled
2-0	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.

Table 11-236. Register Call Summary for DCAN_TEST

DCAN Functional Description • Message RAM Representation in Direct Access Mode: [0] • Test Modes: [0] • Loopback Mode: [0] • Silent Mode: [0] • Loopback Mode Combined With Silent Mode: [0] • External Loopback Mode: [0] • Software Control of CAN_TX Pin: [0]
DCAN Registers • DCAN_TEST Register (Offset = 14h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.7 DCAN_PERR Register (Offset = 1Ch) [reset = 0h]

DCAN_PERR is shown in Figure 11-100 and described in Table 11-238.

Parity Error Code Register
If a parity error is detected, the PER flag will be set in the DCAN_ES. This bit is not reset by the parity check mechanism; it must be reset by reading DCAN_ES. In addition to the PER flag, the parity error code register will indicate the memory area where the parity error has been detected (message number and word number). If more than one word with a parity error was detected, the highest word number with a parity error will be displayed. After a parity error has been detected, the register will hold the last error code until power is removed.

Table 11-237. DCAN_PERR Instances

Instance	Physical Address
DCAN_0_DATA	0260 401Ch
DCAN_1_DATA	0260 801Ch
DCAN_0	0260 B21Ch
DCAN_1	0260 B41Ch

Figure 11-100. DCAN_PERR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					WORD_NUMBER		
R-0h					R--h		
7	6	5	4	3	2	1	0
MESSAGE_NUMBER							
R--h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-238. DCAN_PERR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
10-8	WORD_NUMBER	R	-h	Word number where parity error has been detectedRDA word number (1 to 5) of the message object (according to the message RAM representation in RDA mode).
7-0	MESSAGE_NUMBER	R	-h	Message object number where parity error has been detected (0x01-0x80)

Table 11-239. Register Call Summary for DCAN_PERR

DCAN Functional Description • Behavior on Double Bit Error: [0][1]
DCAN Registers • DCAN_PERR Register (Offset = 1Ch) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.8 DCAN_REL Register (Offset = 20h) [reset = A3170504h]

DCAN_REL is shown in Figure 11-101 and described in Table 11-241.

Core revision register

Table 11-240. DCAN_REL Instances

Instance	Physical Address
DCAN_0_DATA	0260 4020h
DCAN_1_DATA	0260 8020h
DCAN_0	0260 B220h
DCAN_1	0260 B420h

Figure 11-101. DCAN_REL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REVISION																															
R-A317 0504h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-241. DCAN_REL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REVISION	R	A317 0504h	TI internal data. Identifies revision of peripheral.

Table 11-242. Register Call Summary for DCAN_REL

DCAN Registers
- DCAN_REL Register (Offset = 20h) [reset = A3170504hh]: [0]
- DCAN Registers: [0]

11.2.5.9 DCAN_ECCDIAG Register (Offset = 24h) [reset = Ah]

DCAN_ECCDIAG is shown in Figure 11-102 and described in Table 11-244.

ECC Diagnostic Register. This register is writable only in privileged mode.

Table 11-243. DCAN_ECCDIAG Instances

Instance	Physical Address
DCAN_0_DATA	0260 4024h
DCAN_1_DATA	0260 8024h
DCAN_0	0260 B224h
DCAN_1	0260 B424h

Figure 11-102. DCAN_ECCDIAG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												ECCDIAG			
R-0h												R/W-Ah			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-244. DCAN_ECCDIAG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-0	ECCDIAG	R/W	Ah	SECDED diagnostic mode enable/disable 0x5: Diagnostic mode is enabled. Single and double bit errors are shown in the DCAN_ECCDIAG_STAT and the DCAN_ECC_CS. A double bit error (or single bit error with single bit error correction disabled) also triggers the parity interrupt flag (PER). Memory mapping of ECC RAM is enabled 0xA: Diagnostic mode is disabled, single and double bit errors are shown only in the DCAN_ECC_CS.

Table 11-245. Register Call Summary for DCAN_ECCDIAG

DCAN Functional Description • SECDED Testing: [0][1]
DCAN Registers • DCAN_ECCDIAG Register (Offset = 24h) [reset = Ah]: [0] • DCAN Registers: [0]

11.2.5.10 DCAN_ECCDIAG_STAT Register (Offset = 28h) [reset = 0h]

DCAN_ECCDIAG_STAT is shown in Figure 11-103 and described in Table 11-247.

ECC Diagnostic Status Register

Table 11-246. DCAN_ECCDIAG_STAT Instances

Instance	Physical Address
DCAN_0_DATA	0260 4028h
DCAN_1_DATA	0260 8028h
DCAN_0	0260 B228h
DCAN_1	0260 B428h

Figure 11-103. DCAN_ECCDIAG_STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							DEFLG_DIAG
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED							SEFLG_DIAG
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-247. DCAN_ECCDIAG_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	DEFLG_DIAG	R/W	0h	Double bit error flag diagnostic Read 0: No double bit error detected. Write 0: The bit is unchanged. Read 1: Double bit error detected in diagnostic mode. Write 1: The bit is cleared to 0.
7-1	RESERVED	R	0h	Reserved
0	SEFLG_DIAG	R/W	0h	Single bit error flag diagnostic Read 0: No single bit error detected. Write 0: The bit is unchanged. Read 1: Single bit error detected in diagnostic mode. Write 1: The bit is cleared to 0.

Table 11-248. Register Call Summary for DCAN_ECCDIAG_STAT

DCAN Functional Description • SECDED Testing: [0]
DCAN Registers • DCAN_ECCDIAG Register (Offset = 24h) [reset = Ah]: [0] • DCAN Registers: [0] • DCAN_ECCDIAG_STAT Register (Offset = 28h) [reset = 0h]: [0]

11.2.5.11 DCAN_ECC_CS Register (Offset = 2Ch) [reset = 0h]

DCAN_ECC_CS is shown in Figure 11-104 and described in Table 11-250.

ECC Control and Status Register

Table 11-249. DCAN_ECC_CS Instances

Instance	Physical Address
DCAN_0_DATA	0260 402Ch
DCAN_1_DATA	0260 802Ch
DCAN_0	0260 B22Ch
DCAN_1	0260 B42Ch

Figure 11-104. DCAN_ECC_CS Register

31	30	29	28	27	26	25	24
RESERVED				SBE_EVT_EN			
R-0h				R/W-0h			
23	22	21	20	19	18	17	16
RESERVED				ECCMODE			
R-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED							DEFLG
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED							SEFLG
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-250. DCAN_ECC_CS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	SBE_EVT_EN	R/W	0h	Enable/disable SECDED single bit error event (CAN_SERR signal). Write in privileged mode only. 0x5: SECDED single bit error event is disabled, single bit errors are not signaled with a high pulse on CAN_SERR signal. Others: SECDED single bit error event is enabled, single bit errors are signaled with a high pulse on CAN_SERR signal.
23-20	RESERVED	R	0h	Reserved
19-16	ECCMODE	R/W	0h	Enable/disable SECDED single bit error correction. Write in privileged mode only. 0x5: SECDED single bit error correction disabled Others: SECDED single bit error correction enabled
15-9	RESERVED	R	0h	Reserved
8	DEFLG	R/W	0h	Double bit error flag Read 0: No double bit error detected. Write 0: The bit is unchanged. Read 1: Double bit error detected. Write 1: The bit is cleared to 0.
7-1	RESERVED	R	0h	Reserved

Table 11-250. DCAN_ECC_CS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	SEFLG	R/W	0h	Single bit error flag Read 0: No single bit error detected. Write 0: The bit is unchanged. Read 1: Single bit error detected. Write 1: The bit is cleared to 0.

Table 11-251. Register Call Summary for DCAN_ECC_CS

DCAN Functional Description • SECDED Mechanism: [0] • Behavior on Double Bit Error: [0] • SECDED Testing: [0]
DCAN Registers • DCAN_ECC_SERR Register (Offset = 30h) [reset = 0h]: [0] • DCAN_ECC_CS Register (Offset = 2Ch) [reset = 0h]: [0] • DCAN_ES Register (Offset = 4h) [reset = 7h]: [0] • DCAN Registers: [0] • DCAN_ECCDIAG Register (Offset = 24h) [reset = Ah]: [0][1]

11.2.5.12 DCAN_ECC_SERR Register (Offset = 30h) [reset = 0h]

DCAN_ECC_SERR is shown in Figure 11-105 and described in Table 11-253.

ECC Single Bit Error Code Register.
If an ECC single bit error is detected, the SEFLG flag will be set in the DCAN_ECC_CS. In addition, MESSAGE_NUMBER will indicate the memory area where the single bit error has been detected (message object number only).
If more than one word with an ECC single bit error was detected, the highest word number with an ECC single bit error error will be displayed.
After an ECC single bit error has been detected, the register will hold the last error code until power is removed.

Table 11-252. DCAN_ECC_SERR Instances

Instance	Physical Address
DCAN_0_DATA	0260 4030h
DCAN_1_DATA	0260 8030h
DCAN_0	0260 B230h
DCAN_1	0260 B430h

Figure 11-105. DCAN_ECC_SERR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								MESSAGE_NUMBER							
R-0h								R/W--h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-253. DCAN_ECC_SERR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	MESSAGE_NUMBER	R/W	-h	Message object number where ECC single bit error has been detected. 0x0: Reserved

Table 11-254. Register Call Summary for DCAN_ECC_SERR

DCAN Functional Description • Behavior on Single Bit Error: [0][1]
DCAN Registers • DCAN_ECC_SERR Register (Offset = 30h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.13 DCAN_ABOTR Register (Offset = 80h) [reset = 0h]

DCAN_ABOTR is shown in Figure 11-106 and described in Table 11-256.

Auto-Bus-On Time Register
On write access to the DCAN_CTL while Auto-Bus-On timer is running, the Auto-Bus-On procedure will be aborted. During Debug/Suspend mode, running Auto-Bus-On timer will be paused.

Table 11-255. DCAN_ABOTR Instances

Instance	Physical Address
DCAN_0_DATA	0260 4080h
DCAN_1_DATA	0260 8080h
DCAN_0	0260 B280h
DCAN_1	0260 B480h

Figure 11-106. DCAN_ABOTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABO_TIME																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-256. DCAN_ABOTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ABO_TIME	R/W	0h	Number of DCAN_i_VBUS_CLK (i = 0 or 1) clock cycles before a Bus-Off recovery sequence is started by clearing the INIT bit. This function has to be enabled by setting bit ABO in DCAN_CTL. The Auto-Bus-On timer is realized by a 32-bit counter which starts to count down to zero when the module goes Bus-Off. The counter will be reloaded with the preload value of the DCAN_ABOTR after this phase.

Table 11-257. Register Call Summary for DCAN_ABOTR

DCAN Functional Description • Auto-Bus-On: [0]
DCAN Registers • DCAN_ABOTR Register (Offset = 80h) [reset = 0h]: [0][1] • DCAN Registers: [0]

11.2.5.14 DCAN_TXRQ_X Register (Offset = 84h) [reset = 0h]

DCAN_TXRQ_X is shown in Figure 11-107 and described in Table 11-259.

Transmission Request X Register
The software can detect if one or more bits in the different transmission request registers are set. Each register bit represents a group of eight message objects. If at least one of the TxRqst bits of these message objects are set, the corresponding bit in the transmission request X register will be set.

Table 11-258. DCAN_TXRQ_X Instances

Instance	Physical Address
DCAN_0_DATA	0260 4084h
DCAN_1_DATA	0260 8084h
DCAN_0	0260 B284h
DCAN_1	0260 B484h

Figure 11-107. DCAN_TXRQ_X Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
TXRQSTREG8		TXRQSTREG7		TXRQSTREG6		TXRQSTREG5	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
TXRQSTREG4		TXRQSTREG3		TXRQSTREG2		TXRQSTREG1	
R-0h		R-0h		R-0h		R-0h	

LEGEND: R = Read Only; *-n* = value after reset

Table 11-259. DCAN_TXRQ_X Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	RESERVED
15-14	TXRQSTREG8	R	0h	Transmission request bits (aggregate for 113-128 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
13-12	TXRQSTREG7	R	0h	Transmission request bits (aggregate for 97-112 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
11-10	TXRQSTREG6	R	0h	Transmission request bits (aggregate for 81-96 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
9-8	TXRQSTREG5	R	0h	Transmission request bits (aggregate for 65-80 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
7-6	TXRQSTREG4	R	0h	Transmission request bits (aggregate for 49-64 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
5-4	TXRQSTREG3	R	0h	Transmission request bits (aggregate for 33-48 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.

Table 11-259. DCAN_TXRQ_X Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	TXRQSTREG2	R	0h	Transmission request bits (aggregate for 17-32 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
1-0	TXRQSTREG1	R	0h	Transmission request bits (aggregate for 1-16 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.

Table 11-260. Register Call Summary for DCAN_TXRQ_X

DCAN Functional Description • CAN Message Transfer (Normal Operation): [0]
DCAN Registers • DCAN_TXRQ_X Register (Offset = 84h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.15 DCAN_TXRQ12 Register (Offset = 88h) [reset = 0h]

DCAN_TXRQ12 is shown in Figure 11-108 and described in Table 11-262.

Transmission Request Register
This register holds the TxRqst bits of the implemented message objects. By reading out these bits, the software can check for pending transmission requests. The TxRqst bit in a specific message object can be set/reset by the software via the IF1/IF2 message interface registers, or by the message handler after reception of a remote frame or after a successful transmission.

Table 11-261. DCAN_TXRQ12 Instances

Instance	Physical Address
DCAN_0_DATA	0260 4088h
DCAN_1_DATA	0260 8088h
DCAN_0	0260 B288h
DCAN_1	0260 B488h

Figure 11-108. DCAN_TXRQ12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXRQS																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-262. DCAN_TXRQ12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXRQS	R	0h	Transmission request bits (for 1-32 message objects) 0: No transmission has been requested for this message object. 1: The transmission of this message object is requested and is not yet done.

Table 11-263. Register Call Summary for DCAN_TXRQ12

DCAN Functional Description • Transmission of Messages in Event Driven CAN Communication: [0]
DCAN Registers • DCAN_TXRQ12 Register (Offset = 88h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.16 DCAN_TXRQ34 Register (Offset = 8Ch) [reset = 0h]

DCAN_TXRQ34 is shown in Figure 11-109 and described in Table 11-265.

Transmission Request Register
This register holds the TxRqst bits of the implemented message objects. By reading out these bits, the software can check for pending transmission requests. The TxRqst bit in a specific message object can be set/reset by the software via the IF1/IF2 message interface registers, or by the message handler after reception of a remote frame or after a successful transmission.

Table 11-264. DCAN_TXRQ34 Instances

Instance	Physical Address
DCAN_0_DATA	0260 408Ch
DCAN_1_DATA	0260 808Ch
DCAN_0	0260 B28Ch
DCAN_1	0260 B48Ch

Figure 11-109. DCAN_TXRQ34 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXRQS																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-265. DCAN_TXRQ34 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXRQS	R	0h	Transmission request bits (for 33-64 message objects) 0: No transmission has been requested for this message object. 1: The transmission of this message object is requested and is not yet done.

Table 11-266. Register Call Summary for DCAN_TXRQ34

DCAN Registers
- DCAN_TXRQ34 Register (Offset = 8Ch) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.17 DCAN_TXRQ56 Register (Offset = 90h) [reset = 0h]

DCAN_TXRQ56 is shown in Figure 11-110 and described in Table 11-268.

Transmission Request Register
This register holds the TxRqst bits of the implemented message objects. By reading out these bits, the software can check for pending transmission requests. The TxRqst bit in a specific message object can be set/reset by the software via the IF1/IF2 message interface registers, or by the message handler after reception of a remote frame or after a successful transmission.

Table 11-267. DCAN_TXRQ56 Instances

Instance	Physical Address
DCAN_0_DATA	0260 4090h
DCAN_1_DATA	0260 8090h
DCAN_0	0260 B290h
DCAN_1	0260 B490h

Figure 11-110. DCAN_TXRQ56 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXRQS																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-268. DCAN_TXRQ56 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXRQS	R	0h	Transmission request bits (for 65-96 message objects) 0: No transmission has been requested for this message object. 1: The transmission of this message object is requested and is not yet done.

Table 11-269. Register Call Summary for DCAN_TXRQ56

DCAN Registers
- DCAN_TXRQ56 Register (Offset = 90h) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.18 DCAN_TXRQ78 Register (Offset = 94h) [reset = 0h]

DCAN_TXRQ78 is shown in Figure 11-111 and described in Table 11-271.

Transmission Request Register
This register holds the TxRqst bits of the implemented message objects. By reading out these bits, the software can check for pending transmission requests. The TxRqst bit in a specific message object can be set/reset by the software via the IF1/IF2 message interface registers, or by the message handler after reception of a remote frame or after a successful transmission.

Table 11-270. DCAN_TXRQ78 Instances

Instance	Physical Address
DCAN_0_DATA	0260 4094h
DCAN_1_DATA	0260 8094h
DCAN_0	0260 B294h
DCAN_1	0260 B494h

Figure 11-111. DCAN_TXRQ78 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXRQS																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-271. DCAN_TXRQ78 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXRQS	R	0h	Transmission request bits (for 97-128 message objects) 0: No transmission has been requested for this message object. 1: The transmission of this message object is requested and is not yet done.

Table 11-272. Register Call Summary for DCAN_TXRQ78

DCAN Functional Description • Transmission of Messages in Event Driven CAN Communication: [0]
DCAN Registers • DCAN_TXRQ78 Register (Offset = 94h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.19 DCAN_NWDAT_X Register (Offset = 98h) [reset = 0h]

DCAN_NWDAT_X is shown in Figure 11-112 and described in Table 11-274.

New Data X Register
With the new data X register, the software can detect if one or more bits in the different new data registers are set. Each register bit represents a group of eight message objects. If at least on of the NewDat bits of these message objects are set, the corresponding bit in the new data X register will be set

Table 11-273. DCAN_NWDAT_X Instances

Instance	Physical Address
DCAN_0_DATA	0260 4098h
DCAN_1_DATA	0260 8098h
DCAN_0	0260 B298h
DCAN_1	0260 B498h

Figure 11-112. DCAN_NWDAT_X Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
NEWDATREG8		NEWDATREG7		NEWDATREG6		NEWDATREG5	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
NEWDATREG4		NEWDATREG3		NEWDATREG2		NEWDATREG1	
R-0h		R-0h		R-0h		R-0h	

LEGEND: R = Read Only; *-n* = value after reset

Table 11-274. DCAN_NWDAT_X Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-14	NEWDATREG8	R	0h	New data bits (aggregate for 113-128 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
13-12	NEWDATREG7	R	0h	New data bits (aggregate for 97-112 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
11-10	NEWDATREG6	R	0h	New data bits (aggregate for 81-96 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
9-8	NEWDATREG5	R	0h	New data bits (aggregate for 65-80 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
7-6	NEWDATREG4	R	0h	New data bits (aggregate for 49-64 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
5-4	NEWDATREG3	R	0h	New data bits (aggregate for 33-48 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.

Table 11-274. DCAN_NWDAT_X Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	NEWDATREG2	R	0h	New data bits (aggregate for 17-32 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
1-0	NEWDATREG1	R	0h	New data bits (aggregate for 1-16 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.

Table 11-275. Register Call Summary for DCAN_NWDAT_X

DCAN Functional Description • CAN Message Transfer (Normal Operation): [0]
DCAN Registers • DCAN_NWDAT_X Register (Offset = 98h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.20 DCAN_NWDAT12 Register (Offset = 9Ch) [reset = 0h]

DCAN_NWDAT12 is shown in Figure 11-113 and described in Table 11-277.

New Data Register
This register hold the NewDat bits of the implemented message objects. By reading out these bits, the software can check for new data in the message objects. The NewDat bit of a specific message object can be set/reset by the software via the IF1/IF2 interface register sets, or by the message handler after reception of a data frame or after a successful transmission.

Table 11-276. DCAN_NWDAT12 Instances

Instance	Physical Address
DCAN_0_DATA	0260 409Ch
DCAN_1_DATA	0260 809Ch
DCAN_0	0260 B29Ch
DCAN_1	0260 B49Ch

Figure 11-113. DCAN_NWDAT12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NEWDAT																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-277. DCAN_NWDAT12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NEWDAT	R	0h	New Data Bits (for 1-32 message objects) 0: No new data has been written into the data portion of this message object by the Message Handler since the last time when this flag was cleared by the software. 1: The Message Handler or the software has written new data into the data portion of this message object.

Table 11-278. Register Call Summary for DCAN_NWDAT12

DCAN Registers
- DCAN_NWDAT12 Register (Offset = 9Ch) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.21 DCAN_NWDAT34 Register (Offset = A0h) [reset = 0h]

DCAN_NWDAT34 is shown in Figure 11-114 and described in Table 11-280.

New Data Register
This register hold the NewDat bits of the implemented message objects. By reading out these bits, the software can check for new data in the message objects. The NewDat bit of a specific message object can be set/reset by the software via the IF1/IF2 interface register sets, or by the message handler after reception of a data frame or after a successful transmission.

Table 11-279. DCAN_NWDAT34 Instances

Instance	Physical Address
DCAN_0_DATA	0260 40A0h
DCAN_1_DATA	0260 80A0h
DCAN_0	0260 B2A0h
DCAN_1	0260 B4A0h

Figure 11-114. DCAN_NWDAT34 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NEWDAT																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-280. DCAN_NWDAT34 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NEWDAT	R	0h	New Data Bits (for 33-64 message objects) 0: No new data has been written into the data portion of this message object by the Message Handler since the last time when this flag was cleared by the software. 1: The Message Handler or the software has written new data into the data portion of this message object.

Table 11-281. Register Call Summary for DCAN_NWDAT34

DCAN Registers
- DCAN_NWDAT34 Register (Offset = A0h) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.22 DCAN_NWDAT56 Register (Offset = A4h) [reset = 0h]

DCAN_NWDAT56 is shown in Figure 11-115 and described in Table 11-283.

New Data Register
This register hold the NewDat bits of the implemented message objects. By reading out these bits, the software can check for new data in the message objects. The NewDat bit of a specific message object can be set/reset by the software via the IF1/IF2 interface register sets, or by the message handler after reception of a data frame or after a successful transmission.

Table 11-282. DCAN_NWDAT56 Instances

Instance	Physical Address
DCAN_0_DATA	0260 40A4h
DCAN_1_DATA	0260 80A4h
DCAN_0	0260 B2A4h
DCAN_1	0260 B4A4h

Figure 11-115. DCAN_NWDAT56 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NEWDAT																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-283. DCAN_NWDAT56 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NEWDAT	R	0h	New Data Bits (for 65-96 message objects) 0: No new data has been written into the data portion of this message object by the Message Handler since the last time when this flag was cleared by the software. 1: The Message Handler or the software has written new data into the data portion of this message object.

Table 11-284. Register Call Summary for DCAN_NWDAT56

DCAN Registers
- DCAN_NWDAT56 Register (Offset = A4h) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.23 DCAN_NWDAT78 Register (Offset = A8h) [reset = 0h]

DCAN_NWDAT78 is shown in Figure 11-116 and described in Table 11-286.

New Data Register
This register hold the NewDat bits of the implemented message objects. By reading out these bits, the software can check for new data in the message objects. The NewDat bit of a specific message object can be set/reset by the software via the IF1/IF2 interface register sets, or by the message handler after reception of a data frame or after a successful transmission.

Table 11-285. DCAN_NWDAT78 Instances

Instance	Physical Address
DCAN_0_DATA	0260 40A8h
DCAN_1_DATA	0260 80A8h
DCAN_0	0260 B2A8h
DCAN_1	0260 B4A8h

Figure 11-116. DCAN_NWDAT78 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NEWDAT																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-286. DCAN_NWDAT78 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NEWDAT	R	0h	New Data Bits (for 97-128 message objects) 0: No new data has been written into the data portion of this message object by the Message Handler since the last time when this flag was cleared by the software. 1: The Message Handler or the software has written new data into the data portion of this message object.

Table 11-287. Register Call Summary for DCAN_NWDAT78

DCAN Registers
- DCAN_NWDAT78 Register (Offset = A8h) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.24 DCAN_INTPND_X Register (Offset = ACh) [reset = 0h]

DCAN_INTPND_X is shown in Figure 11-117 and described in Table 11-289.

Interrupt Pending X Register
With the interrupt pending X register, the software can detect if one or more bits in the different interrupt pending registers are set. Each bit of this register represents a group of eight message objects. If at least one of the IntPnd bits of these message objects are set, the corresponding bit in the interrupt pending X register will be set.

Table 11-288. DCAN_INTPND_X Instances

Instance	Physical Address
DCAN_0_DATA	0260 40ACh
DCAN_1_DATA	0260 80ACh
DCAN_0	0260 B2ACh
DCAN_1	0260 B4ACh

Figure 11-117. DCAN_INTPND_X Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
INTPNDREG8		INTPNDREG7		INTPNDREG6		INTPNDREG5	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
INTPNDREG4		INTPNDREG3		INTPNDREG2		INTPNDREG1	
R-0h		R-0h		R-0h		R-0h	

LEGEND: R = Read Only; *-n* = value after reset

Table 11-289. DCAN_INTPND_X Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-14	INTPNDREG8	R	0h	Interrupt Pending bits (aggregate for 113-128 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
13-12	INTPNDREG7	R	0h	Interrupt Pending bits (aggregate for 97-112 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
11-10	INTPNDREG6	R	0h	Interrupt Pendingbits (aggregate for 81-96 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
9-8	INTPNDREG5	R	0h	Interrupt Pending bits (aggregate for 65-80 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
7-6	INTPNDREG4	R	0h	Interrupt Pending bits (aggregate for 49-64 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
5-4	INTPNDREG3	R	0h	Interrupt Pending bits (aggregate for 33-48 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.

Table 11-289. DCAN_INTPND_X Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	INTPNDREG2	R	0h	Interrupt Pending bits (aggregate for 17-32 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
1-0	INTPNDREG1	R	0h	Interrupt Pending bits (aggregate for 1-16 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.

Table 11-290. Register Call Summary for DCAN_INTPND_X

DCAN Registers
- DCAN_INTPND_X Register (Offset = ACh) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.25 DCAN_INTPND12 Register (Offset = B0h) [reset = 0h]

DCAN_INTPND12 is shown in Figure 11-118 and described in Table 11-292.

Interrupt Pending Register
This register holds the IntPnd bits of the implemented message objects. By reading out these bits, the software can check for pending interrupts in the message objects. The IntPnd bit of a specific message object can be set/reset by the software via the IF1/IF2 interface register sets, or by the message handler after a reception or a successful transmission.

Table 11-291. DCAN_INTPND12 Instances

Instance	Physical Address
DCAN_0_DATA	0260 40B0h
DCAN_1_DATA	0260 80B0h
DCAN_0	0260 B2B0h
DCAN_1	0260 B4B0h

Figure 11-118. DCAN_INTPND12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTPND																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-292. DCAN_INTPND12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INTPND	R	0h	Interrupt Pending Bits (for 1-32 message objects) 0h (R) = This message object is not the source of an interrupt. 1h (R) = This message object is the source of an interrupt.

Table 11-293. Register Call Summary for DCAN_INTPND12

DCAN Registers
- DCAN_INTPND12 Register (Offset = B0h) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.26 DCAN_INTPND34 Register (Offset = B4h) [reset = 0h]

DCAN_INTPND34 is shown in Figure 11-119 and described in Table 11-295.

Interrupt Pending Register
This register holds the IntPnd bits of the implemented message objects. By reading out these bits, the software can check for pending interrupts in the message objects. The IntPnd bit of a specific message object can be set/reset by the software via the IF1/IF2 interface register sets, or by the message handler after a reception or a successful transmission.

Table 11-294. DCAN_INTPND34 Instances

Instance	Physical Address
DCAN_0_DATA	0260 40B4h
DCAN_1_DATA	0260 80B4h
DCAN_0	0260 B2B4h
DCAN_1	0260 B4B4h

Figure 11-119. DCAN_INTPND34 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTPND																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-295. DCAN_INTPND34 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INTPND	R	0h	Interrupt Pending Bits (for 33-64 message objects) 0h (R) = This message object is not the source of an interrupt. 1h (R) = This message object is the source of an interrupt.

Table 11-296. Register Call Summary for DCAN_INTPND34

DCAN Registers
- DCAN_INTPND34 Register (Offset = B4h) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.27 DCAN_INTPND56 Register (Offset = B8h) [reset = 0h]

DCAN_INTPND56 is shown in Figure 11-120 and described in Table 11-298.

Interrupt Pending Register
This register holds the IntPnd bits of the implemented message objects. By reading out these bits, the software can check for pending interrupts in the message objects. The IntPnd bit of a specific message object can be set/reset by the software via the IF1/IF2 interface register sets, or by the message handler after a reception or a successful transmission.

Table 11-297. DCAN_INTPND56 Instances

Instance	Physical Address
DCAN_0_DATA	0260 40B8h
DCAN_1_DATA	0260 80B8h
DCAN_0	0260 B2B8h
DCAN_1	0260 B4B8h

Figure 11-120. DCAN_INTPND56 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTPND																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-298. DCAN_INTPND56 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INTPND	R	0h	Interrupt Pending Bits (for 65-96 message objects) 0h (R) = This message object is not the source of an interrupt. 1h (R) = This message object is the source of an interrupt.

Table 11-299. Register Call Summary for DCAN_INTPND56

DCAN Registers
- DCAN_INTPND56 Register (Offset = B8h) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.28 DCAN_INTPND78 Register (Offset = BCh) [reset = 0h]

DCAN_INTPND78 is shown in Figure 11-121 and described in Table 11-301.

Interrupt Pending Register
This register holds the IntPnd bits of the implemented message objects. By reading out these bits, the software can check for pending interrupts in the message objects. The IntPnd bit of a specific message object can be set/reset by the software via the IF1/IF2 interface register sets, or by the message handler after a reception or a successful transmission.

Table 11-300. DCAN_INTPND78 Instances

Instance	Physical Address
DCAN_0_DATA	0260 40BCh
DCAN_1_DATA	0260 80BCh
DCAN_0	0260 B2BCh
DCAN_1	0260 B4BCh

Figure 11-121. DCAN_INTPND78 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTPND																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-301. DCAN_INTPND78 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INTPND	R	0h	Interrupt Pending Bits (for 97-128 message objects) 0h (R) = This message object is not the source of an interrupt. 1h (R) = This message object is the source of an interrupt.

Table 11-302. Register Call Summary for DCAN_INTPND78

DCAN Registers
- DCAN_INTPND78 Register (Offset = BCh) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.29 DCAN_MSGVAL_X Register (Offset = C0h) [reset = 0h]

DCAN_MSGVAL_X is shown in Figure 11-122 and described in Table 11-304.

Message Valid X Register
With the message valid X register, the software can detect if one or more bits in the different message valid registers are set. Each bit of this register represents a group of eight message objects. If at least one of the MsgVal bits of these message objects are set, the corresponding bit in the message valid X register will be set.

Table 11-303. DCAN_MSGVAL_X Instances

Instance	Physical Address
DCAN_0_DATA	0260 40C0h
DCAN_1_DATA	0260 80C0h
DCAN_0	0260 B2C0h
DCAN_1	0260 B4C0h

Figure 11-122. DCAN_MSGVAL_X Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
MSGVALREG8		MSGVALREG7		MSGVALREG6		MSGVALREG5	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
MSGVALREG4		MSGVALREG3		MSGVALREG2		MSGVALREG1	
R-0h		R-0h		R-0h		R-0h	

LEGEND: R = Read Only; *-n* = value after reset

Table 11-304. DCAN_MSGVAL_X Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-14	MSGVALREG8	R	0h	Message valid bits (aggregate for 113-128 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
13-12	MSGVALREG7	R	0h	Message valid bits (aggregate for 97-112 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
11-10	MSGVALREG6	R	0h	Message valid bits (aggregate for 81-96 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
9-8	MSGVALREG5	R	0h	Message valid bits (aggregate for 65-80 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
7-6	MSGVALREG4	R	0h	Message valid bits (aggregate for 49-64 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
5-4	MSGVALREG3	R	0h	Message valid bits (aggregate for 33-48 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.

Table 11-304. DCAN_MSGVAL_X Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MSGVALREG2	R	0h	Message valid bits (aggregate for 17-32 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.
1-0	MSGVALREG1	R	0h	Message valid bits (aggregate for 1-16 message objects). Lower bit represents first 8 message objects. Higher bit represents second 8 message objects.

Table 11-305. Register Call Summary for DCAN_MSGVAL_X

DCAN Registers
- DCAN_MSGVAL_X Register (Offset = C0h) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.30 DCAN_MSGVAL12 Register (Offset = C4h) [reset = 0h]

DCAN_MSGVAL12 is shown in Figure 11-123 and described in Table 11-307.

Message Valid Register
These registers hold the MsgVal bits of the implemented message objects. By reading out these bits, the software can check which message objects are valid. The MsgVal bit of a specific message object can be set/reset by the software via the IF1/IF2 interface register sets, or by the message handler after a reception or a successful transmission

Table 11-306. DCAN_MSGVAL12 Instances

Instance	Physical Address
DCAN_0_DATA	0260 40C4h
DCAN_1_DATA	0260 80C4h
DCAN_0	0260 B2C4h
DCAN_1	0260 B4C4h

Figure 11-123. DCAN_MSGVAL12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSGVAL																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-307. DCAN_MSGVAL12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MSGVAL	R	0h	Message valid Bits (for 1-32 message objects) 0h (R) = This message object is ignored by the message handler. 1h (R) = This message object is configured and will be considered by the message handler.

Table 11-308. Register Call Summary for DCAN_MSGVAL12

DCAN Functional Description • Transmission of Messages in Event Driven CAN Communication: [0]
DCAN Registers • DCAN_MSGVAL12 Register (Offset = C4h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.31 DCAN_MSGVAL34 Register (Offset = C8h) [reset = 0h]

DCAN_MSGVAL34 is shown in Figure 11-124 and described in Table 11-310.

Message Valid Register
These registers hold the MsgVal bits of the implemented message objects. By reading out these bits, the software can check which message objects are valid. The MsgVal bit of a specific message object can be set/reset by the software via the IF1/IF2 interface register sets, or by the message handler after a reception or a successful transmission

Table 11-309. DCAN_MSGVAL34 Instances

Instance	Physical Address
DCAN_0_DATA	0260 40C8h
DCAN_1_DATA	0260 80C8h
DCAN_0	0260 B2C8h
DCAN_1	0260 B4C8h

Figure 11-124. DCAN_MSGVAL34 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSGVAL																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-310. DCAN_MSGVAL34 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MSGVAL	R	0h	Message valid Bits (for 33-64 message objects) 0h (R) = This message object is ignored by the message handler. 1h (R) = This message object is configured and will be considered by the message handler.

Table 11-311. Register Call Summary for DCAN_MSGVAL34

DCAN Registers
- DCAN_MSGVAL34 Register (Offset = C8h) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.32 DCAN_MSGVAL56 Register (Offset = CCh) [reset = 0h]

DCAN_MSGVAL56 is shown in Figure 11-125 and described in Table 11-313.

Message Valid Register
These registers hold the MsgVal bits of the implemented message objects. By reading out these bits, the software can check which message objects are valid. The MsgVal bit of a specific message object can be set/reset by the software via the IF1/IF2 interface register sets, or by the message handler after a reception or a successful transmission

Table 11-312. DCAN_MSGVAL56 Instances

Instance	Physical Address
DCAN_0_DATA	0260 40CCh
DCAN_1_DATA	0260 80CCh
DCAN_0	0260 B2CCh
DCAN_1	0260 B4CCh

Figure 11-125. DCAN_MSGVAL56 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSGVAL																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-313. DCAN_MSGVAL56 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MSGVAL	R	0h	Message valid Bits (for 65-96 message objects) 0h (R) = This message object is ignored by the message handler. 1h (R) = This message object is configured and will be considered by the message handler.

Table 11-314. Register Call Summary for DCAN_MSGVAL56

DCAN Registers
- DCAN_MSGVAL56 Register (Offset = CCh) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.33 DCAN_MSGVAL78 Register (Offset = D0h) [reset = 0h]

DCAN_MSGVAL78 is shown in Figure 11-126 and described in Table 11-316.

Message Valid Register
These registers hold the MsgVal bits of the implemented message objects. By reading out these bits, the software can check which message objects are valid. The MsgVal bit of a specific message object can be set/reset by the software via the IF1/IF2 interface register sets, or by the message handler after a reception or a successful transmission

Table 11-315. DCAN_MSGVAL78 Instances

Instance	Physical Address
DCAN_0_DATA	0260 40D0h
DCAN_1_DATA	0260 80D0h
DCAN_0	0260 B2D0h
DCAN_1	0260 B4D0h

Figure 11-126. DCAN_MSGVAL78 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSGVAL																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-316. DCAN_MSGVAL78 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MSGVAL	R	0h	Message valid Bits (for 97-128 message objects) 0h (R) = This message object is ignored by the message handler. 1h (R) = This message object is configured and will be considered by the message handler.

Table 11-317. Register Call Summary for DCAN_MSGVAL78

DCAN Functional Description • Transmission of Messages in Event Driven CAN Communication: [0]
DCAN Registers • DCAN_MSGVAL78 Register (Offset = D0h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.34 DCAN_INTMUX12 Register (Offset = D8h) [reset = 0h]

DCAN_INTMUX12 is shown in Figure 11-127 and described in Table 11-319.

Interrupt Multiplexer Register
The IntMux flag determine for each message object, which of the two interrupt lines (INT0 or INT1) will be asserted when the IntPnd of this message object is set. Both interrupt lines can be globally enabled or disabled by setting or clearing IE0 and IE1 bits in DCAN_CTL. The IntPnd bit of a specific message object can be set or reset by the software via the IF1/IF2 interface register sets, or by message handler after reception or successful transmission of a frame. This will also affect the INT0ID resp. INT1ID flags in the DCAN_INT register.

Table 11-318. DCAN_INTMUX12 Instances

Instance	Physical Address
DCAN_0_DATA	0260 40C8h
DCAN_1_DATA	0260 80C8h
DCAN_0	0260 B2D8h
DCAN_1	0260 B4D8h

Figure 11-127. DCAN_INTMUX12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTMUX																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-319. DCAN_INTMUX12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INTMUX	R/W	0h	Multiplexes IntPnd value to either INT0 or INT1 interrupt lines (bit 0 -> last implemented message object) (bits 1:31 -> 1-31 message objects) 0h (R/W) = INT0 line is active if corresponding IntPnd flag is one. 1h (R/W) = INT1 line is active if corresponding IntPnd flag is one.

Table 11-320. Register Call Summary for DCAN_INTMUX12

DCAN Functional Description • Message Object Interrupts: [0]
DCAN Registers • DCAN_INTMUX12 Register (Offset = D8h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.35 DCAN_INTMUX34 Register (Offset = DCh) [reset = 0h]

DCAN_INTMUX34 is shown in Figure 11-128 and described in Table 11-322.

Interrupt Multiplexer Register
The IntMux flag determine for each message object, which of the two interrupt lines (INT0 or INT1) will be asserted when the IntPnd of this message object is set. Both interrupt lines can be globally enabled or disabled by setting or clearing IE0 and IE1 bits in CAN control register. The IntPnd bit of a specific message object can be set or reset by the software via the IF1/IF2 interface register sets, or by message handler after reception or successful transmission of a frame. This will also affect the INT0ID resp INT1ID flags in the DCAN_INT register.

Table 11-321. DCAN_INTMUX34 Instances

Instance	Physical Address
DCAN_0_DATA	0260 40DCh
DCAN_1_DATA	0260 80DCh
DCAN_0	0260 B2DCh
DCAN_1	0260 B4DCh

Figure 11-128. DCAN_INTMUX34 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTMUX																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-322. DCAN_INTMUX34 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INTMUX	R/W	0h	Multiplexes IntPnd value to either INT0 or INT1 interrupt lines (bits 0:31 -> 32-63 message objects) 0h (R/W) = INT0 line is active if corresponding IntPnd flag is one. 1h (R/W) = INT1 line is active if corresponding IntPnd flag is one.

Table 11-323. Register Call Summary for DCAN_INTMUX34

DCAN Registers
- DCAN_INTMUX34 Register (Offset = DCh) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.36 DCAN_INTMUX56 Register (Offset = E0h) [reset = 0h]

DCAN_INTMUX56 is shown in Figure 11-129 and described in Table 11-325.

Interrupt Multiplexer Register
The IntMux flag determine for each message object, which of the two interrupt lines (INT0 or INT1) will be asserted when the IntPnd of this message object is set. Both interrupt lines can be globally enabled or disabled by setting or clearing IE0 and IE1 bits in CAN control register. The IntPnd bit of a specific message object can be set or reset by the software via the IF1/IF2 interface register sets, or by message handler after reception or successful transmission of a frame. This will also affect the INT0ID resp INT1ID flags in the DCAN_INT register.

Table 11-324. DCAN_INTMUX56 Instances

Instance	Physical Address
DCAN_0_DATA	0260 40E0h
DCAN_1_DATA	0260 80E0h
DCAN_0	0260 B2E0h
DCAN_1	0260 B4E0h

Figure 11-129. DCAN_INTMUX56 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTMUX																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-325. DCAN_INTMUX56 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INTMUX	R/W	0h	Multiplexes IntPnd value to either INT0 or INT1 interrupt lines (bits 0:31 -> 64-95 message objects) 0h (R/W) = INT0 line is active if corresponding IntPnd flag is one. 1h (R/W) = INT1 line is active if corresponding IntPnd flag is one.

Table 11-326. Register Call Summary for DCAN_INTMUX56

DCAN Registers
- DCAN_INTMUX56 Register (Offset = E0h) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.37 DCAN_INTMUX78 Register (Offset = E4h) [reset = 0h]

DCAN_INTMUX78 is shown in Figure 11-130 and described in Table 11-328.

Interrupt Multiplexer Register
The IntMux flag determine for each message object, which of the two interrupt lines (INT0 or INT1) will be asserted when the IntPnd of this message object is set. Both interrupt lines can be globally enabled or disabled by setting or clearing IE0 and IE1 bits in CAN control register. The IntPnd bit of a specific message object can be set or reset by the software via the IF1/IF2 interface register sets, or by message handler after reception or successful transmission of a frame. This will also affect the INT0ID resp INT1ID flags in the DCAN_INT register.

Table 11-327. DCAN_INTMUX78 Instances

Instance	Physical Address
DCAN_0_DATA	0260 40E4h
DCAN_1_DATA	0260 80E4h
DCAN_0	0260 B2E4h
DCAN_1	0260 B4E4h

Figure 11-130. DCAN_INTMUX78 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTMUX																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-328. DCAN_INTMUX78 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INTMUX	R/W	0h	Multiplexes IntPnd value to either INT0 or INT1 interrupt lines (bits 0:31 -> 96-127 message objects) 0h (R/W) = INT0 line is active if corresponding IntPnd flag is one. 1h (R/W) = INT1 line is active if corresponding IntPnd flag is one.

Table 11-329. Register Call Summary for DCAN_INTMUX78

DCAN Functional Description • Message Object Interrupts: [0]
DCAN Registers • DCAN_INTMUX78 Register (Offset = E4h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.38 DCAN_IF1CMD Register (Offset = 100h) [reset = 1h]

DCAN_IF1CMD is shown in Figure 11-131 and described in Table 11-331.

IF1 Command Register
The IF1 Command Register (DCAN_IF1CMD) configure and initiate the transfer between the IF1 register set and the message RAM. It is configurable which portions of the message object should be transferred. A transfer is started when the software writes the message number to bits [7-0] MESSAGE_NUMBER. With this write operation, the BUSY bit is automatically set to 1 to indicate that a transfer is in progress. After 4 to 14 DCAN_i_VBUS_CLK (i = 0 or 1) clock cycles, the transfer between the interface register and the message RAM will be completed and the BUSY bit is cleared. The maximum number of cycles is needed when the message transfer concurs with a CAN message transmission, acceptance filtering, or message storage. If the software writes to both DCAN_IF1CMD/DCAN_IF2CMD consecutively (request of a second transfer while first transfer is still in progress), the second transfer will start after the first one has been completed.
While BUSY bit is one, IF1/IF2 register sets are write protected.
For debug support, the auto clear functionality of the IF1/IF2 command registers (clear of DMAACTIVE flag by r/w) is disabled during Debug/Suspend mode.
If an invalid Message Number is written to bits [7-0] MESSAGE_NUMBER, the message handler may access an implemented (valid) message object instead.

Table 11-330. DCAN_IF1CMD Instances

Instance	Physical Address
DCAN_0_DATA	0260 4100h
DCAN_1_DATA	0260 8100h
DCAN_0	0260 B300h
DCAN_1	0260 B500h

Figure 11-131. DCAN_IF1CMD Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
WR_RD	MASK	ARB	CONTROL	CLRINTPND	TXRQST_NEWDAT	DATA_A	DATA_B
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
BUSY	DMAACTIVE	RESERVED					
R/W-0h	R/W-0h	R-0h					
7	6	5	4	3	2	1	0
MESSAGE_NUMBER							
R/W-1h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-331. DCAN_IF1CMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
23	WR_RD	R/W	0h	Write/Read 0h (R/W) = Direction = Read: Transfer direction is from the message object addressed by MESSAGE_NUMBER to the IF1 register set. 1h (R/W) = Direction = Write: Transfer direction is from the IF1 register set to the message object addressed by MESSAGE_NUMBER.

Table 11-331. DCAN_IF1CMD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	MASK	R/W	0h	Access mask bits 0h (R/W) = Mask bits will not be changed 1h (R/W) = Direction = Write: The mask bits (identifier mask + MDir + MXtd) will be transferred from the IF1 register set to the message object addressed by MESSAGE_NUMBER.
21	ARB	R/W	0h	Access arbitration bits 0h (R/W) = Arbitration bits will not be changed 1h (R/W) = Direction = Write: The Arbitration bits (Identifier + Dir + Xtd + MsgVal) will be transferred from the IF1 register set to the message object addressed by MESSAGE_NUMBER.
20	CONTROL	R/W	0h	Access control bitsIf the TXRQST_NEWDAT bit in this register(Bit [18]) is set, the TXRQST/ NEWDAT bits in the DCAN_IF1MCTL will be ignored. 0h (R/W) = Control bits will not be changed 1h (R/W) = Direction = Write: The message control bits will be transferred from the IF1 registerset to the message object addressed by MESSAGE_NUMBER.
19	CLRINTPND	R/W	0h	Clear interrupt pending bit 0h (R/W) = IntPnd bit will not be changed 1h (R/W) = Direction = Write: This bit is ignored. Copying of IntPnd flag from IF1 Registers to message RAM can only be controlled by the CONTROL flag (Bit [20]).
18	TXRQST_NEWDAT	R/W	0h	Access transmission request bitNote: If a CAN transmission is requested by setting TXRQST_NEWDAT in this register, the TxRqst/NewDat bits in the message object will be set to one independent of the values in DCAN_IF1MCTL. Note: A read access to a message object can be combined with the reset of the control bits IntPnd and NewDat. The values of these bits transferred to the DCAN_IF1MCTL always reflect the status before resetting them. 0h (R/W) = Direction = Read: NewDat bit will not be changed. Direction = Write: TxRqst/NewDat bit will be handled according to the CONTROL bit. 1h (R/W) = Direction = Read: Clears NewDat bit in the message object. 1: Direction = Write: Sets TxRqst/NewDat in message object.
17	DATA_A	R/W	0h	Access Data Bytes 0-3Note: The duration of the message transfer is independent of the number of bytes to be transferred. 0h (R/W) = Data Bytes 0-3 will not be changed. 1h (R/W) = Direction = Write: The data bytes 0-3 will be transferred from the IF1 registerset to the message object addressed by the MESSAGE_NUMBER.
16	DATA_B	R/W	0h	Access Data Bytes 4-7Note: The duration of the message transfer is independent of the number of bytes to be transferred. 0h (R/W) = Data Bytes 4-7 will not be changed. 1h (R/W) = Direction = Write: The data bytes 4-7 will be transferred from the IF1 registerset to the message object addressed by MESSAGE_NUMBER.
15	BUSY	R/W	0h	Busy flagThis bit is set to one after the message number has been written to bits [7-0] MESSAGE_NUMBER. IF1 register set will be write protected. The bit is cleared after read/write action has been finished. 0h (R/W) = No transfer between IF1 register set and message RAM is in progress. 1h (R/W) = Transfer between IF1 register set and message RAM is in progress.

Table 11-331. DCAN_IF1CMD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	DMAACTIVE	R/W	0h	Activation of DMA feature for subsequent internal IF1 updateThe DMA request remains active until the first read or write to one of the IF1 registers; an exception is a write to MESSAGE_NUMBER when DMAACTIVE is one. Note: Due to the auto reset feature of the DMAACTIVE bit, this bit has to be set for each subsequent DMA cycle separately. 0h (R/W) = DMA request line is independent of IF1 activities. 1h (R/W) = DMA is requested after completed transfer between IF1 register set and message RAM.
13-8	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
7-0	MESSAGE_NUMBER	R/W	1h	Number of message object in message RAM which is used for data transfer0x01-0x80: Valid message numbers. 0x81-0xFF: Invalid message numbers. 0h (R/W) = Invalid message number

Table 11-332. Register Call Summary for DCAN_IF1CMD

DCAN Functional Description • Reading Received Messages: [0] • Requesting New Data for a Receive Object: [0] • DMA Functionality: [0] • Debug/Suspend Mode: [0] • Interrupt Functionality: [0] • Changing a Transmit Object: [0][1] • Updating a Transmit Object: [0] • Transmission of Messages in Event Driven CAN Communication: [0] • Message Interface Register Sets 1 and 2: [0][1] • Message Interface Register Sets: [0][1]
DCAN Registers • DCAN_IF1DATA Register (Offset = 110h) [reset = 0h]: [0] • DCAN_IF1DATB Register (Offset = 114h) [reset = 0h]: [0] • DCAN_IF2CMD Register (Offset = 120h) [reset = 1h]: [0] • DCAN_IF2MCTL Register (Offset = 12Ch) [reset = 0h]: [0] • DCAN_IF1ARB Register (Offset = 108h) [reset = 0h]: [0] • DCAN_IF1MCTL Register (Offset = 10Ch) [reset = 0h]: [0] • DCAN_IF2MSK Register (Offset = 124h) [reset = FFFFFFFFh]: [0] • DCAN_IF1MSK Register (Offset = 104h) [reset = FFFFFFFFh]: [0] • DCAN_IF1CMD Register (Offset = 100h) [reset = 1h]: [0][1][2] • DCAN_IF2ARB Register (Offset = 128h) [reset = 0h]: [0] • DCAN Registers: [0] • DCAN_IF2DATB Register (Offset = 134h) [reset = 0h]: [0] • DCAN_IF2DATA Register (Offset = 130h) [reset = 0h]: [0]

11.2.5.39 DCAN_IF1MSK Register (Offset = 104h) [reset = FFFFFFFFh]

DCAN_IF1MSK is shown in Figure 11-132 and described in Table 11-334.

IF1 Mask Register
The bits of the IF1/IF2 mask registers mirror the mask bits of a message object. The function of the relevant message objects bits is described in *Structure of Message Objects*.
While BUSY bit of DCAN_IF1CMD/DCAN_IF2CMD register is one, IF1/IF2 register set is write protected.

Table 11-333. DCAN_IF1MSK Instances

Instance	Physical Address
DCAN_0_DATA	0260 4104h
DCAN_1_DATA	0260 8104h
DCAN_0	0260 B304h
DCAN_1	0260 B504h

Figure 11-132. DCAN_IF1MSK Register

31	30	29	28	27	26	25	24
MXTD	MDIR	RESERVED	MSK				
R/W-1h	R/W-1h	R-1h	R/W-1FFFFFFFh				
23	22	21	20	19	18	17	16
MSK							
R/W-1FFFFFFFh							
15	14	13	12	11	10	9	8
MSK							
R/W-1FFFFFFFh							
7	6	5	4	3	2	1	0
MSK							
R/W-1FFFFFFFh							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-334. DCAN_IF1MSK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MXTD	R/W	1h	Mask Extended IdentifierWhen 11-bit (standard) identifiers are used for a message object, the identifiers of received data frames are written into bits ID[28-18]. For acceptance filtering, only these bits together with mask bits Msk[28-18] are considered. 0h (R/W) = The extended identifier bit (IDE) has no effect on the acceptance filtering. 1h (R/W) = The extended identifier bit (IDE) is used for acceptance filtering.
30	MDIR	R/W	1h	Mask Message Direction 0h (R/W) = The message direction bit (Dir) has no effect on the acceptance filtering. 1h (R/W) = The message direction bit (Dir) is used for acceptance filtering.
29	RESERVED	R	1h	This bit is always read as 1. Writes have no effect.
28-0	MSK	R/W	1FFFFFFFh	Identifier Mask 0h (R/W) = The corresponding bit in the identifier of the message object is not used for acceptance filtering (don't care). 1h (R/W) = The corresponding bit in the identifier of the message object is used for acceptance filtering.

Table 11-335. Register Call Summary for DCAN_IF1MSK

DCAN Registers
• DCAN Registers: [0] • DCAN_IF1MSK Register (Offset = 104h) [reset = FFFFFFFFh]: [0]

11.2.5.40 DCAN_IF1ARB Register (Offset = 108h) [reset = 0h]

DCAN_IF1ARB is shown in Figure 11-133 and described in Table 11-337.

IF1 arbitration register
The Arbitration bits ID[28-0], XTD, and DIR are used to define the identifier and type of outgoing messages and (together with the mask bits MSK[28-0], MXTD, and MDIR) for acceptance filtering of incoming messages. A received message is stored into the valid message object with matching identifier and Direction = receive (data frame) or Direction = transmit (remote frame). Extended frames can be stored only in message objects with XTD = 1, standard frames in message objects with XTD = 0. If a received message (data frame or remote frame) matches more than one valid message objects, it is stored into the one with the lowest message number.
The bits of the IF1/IF2 arbitration registers mirror the arbitration bits of a message object. The function of the relevant message objects bits is described in *Structure of Message Objects*
While BUSY bit of DCAN_IF1CMD/DCAN_IF2CMD register is one, IF1/IF2 register set is write protected.

Table 11-336. DCAN_IF1ARB Instances

Instance	Physical Address
DCAN_0_DATA	0260 4108h
DCAN_1_DATA	0260 8108h
DCAN_0	0260 B308h
DCAN_1	0260 B508h

Figure 11-133. DCAN_IF1ARB Register

31	30	29	28	27	26	25	24
MSGVAL	XTD	DIR	ID				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				
23	22	21	20	19	18	17	16
ID							
R/W-0h							
15	14	13	12	11	10	9	8
ID							
R/W-0h							
7	6	5	4	3	2	1	0
ID							
R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-337. DCAN_IF1ARB Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MSGVAL	R/W	0h	Message validThe software should reset the MsgVal bit of all unused Messages Objects during the initialization before it resets bit INIT in the DCAN_CTL. This bit must also be reset if the messages object is no longer required. 0h (R/W) = The message object is ignored by the message handler. 1h (R/W) = The message object is to be used by the message handler.
30	XTD	R/W	0h	Extended identifier 0h (R/W) = The 11-bit (standard) Identifier is used for this message object. 1h (R/W) = The 29-bit (extended) Identifier is used for this message object.

Table 11-337. DCAN_IF1ARB Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
29	DIR	R/W	0h	Message direction 0h (R/W) = Direction = receive: On TxRqst, a remote frame with the identifier of this message object is transmitted. On reception of a data frame with matching identifier, this message is stored in this message object. 1h (R/W) = Direction = transmit: On TxRqst, the respective message object is transmitted as a data frame. On reception of a remote frame with matching identifier, the TxRqst bit of this message object is set (if RmtEn = 1).
28-0	ID	R/W	0h	Message identifierID[28-0]: 29-bit identifier (extended frame). ID[28-18]: 11-bit identifier (standard frame).

Table 11-338. Register Call Summary for DCAN_IF1ARB

DCAN Functional Description • Configuration of a Single Receive Object for Data Frames: [0]
DCAN Registers • DCAN_IF1ARB Register (Offset = 108h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.41 DCAN_IF1MCTL Register (Offset = 10Ch) [reset = 0h]

DCAN_IF1MCTL is shown in Figure 11-134 and described in Table 11-340.

IF1 Message Control Register
The bits of the IF1/IF2 message control registers mirror the message control bits of a message object. The function of the relevant message objects bits is described in *Structure of Message Objects*
While BUSY bit of DCAN_IF1CMD/DCAN_IF2CMD register is one, IF1/IF2 register set is write protected.

Table 11-339. DCAN_IF1MCTL Instances

Instance	Physical Address
DCAN_0_DATA	0260 410Ch
DCAN_1_DATA	0260 810Ch
DCAN_0	0260 B30Ch
DCAN_1	0260 B50Ch

Figure 11-134. DCAN_IF1MCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
NEWDAT	MSGLST	INTPND	UMASK	TXIE	RXIE	RMTEN	TXRQST
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EOB	RESERVED			DLC			
R/W-0h	R-0h			R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-340. DCAN_IF1MCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
15	NEWDAT	R/W	0h	New data 0h (R/W) = No new data has been written into the data portion of this message object by the message handler since the last time when this flag was cleared by the software. 1h (R/W) = The message handler or the software has written new data into the data portion of this message object.
14	MSGLST	R/W	0h	Message lost (only valid for message objects with direction = receive) 0h (R/W) = No message lost since the last time when this bit was reset by the software. 1h (R/W) = The message handler stored a new message into this object when NewDat was still set, so the previous message has been overwritten.
13	INTPND	R/W	0h	Interrupt pending 0h (R/W) = This message object is not the source of an interrupt. 1h (R/W) = This message object is the source of an interrupt. The Interrupt Identifier in DCAN_INT will point to this message object if there is no other interrupt source with higher priority.

Table 11-340. DCAN_IF1MCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	UMASK	R/W	0h	Use acceptance maskIf the UMASK bit is set to one, the message object's mask bits have to be programmed during initialization of the message object before MsgVal is set to one. 0h (R/W) = Mask ignored 1h (R/W) = Use mask (Msk[28-0], MXtd, and MDir) for acceptance filtering
11	TXIE	R/W	0h	Transmit interrupt enable 0h (R/W) = IntPnd will not be triggered after the successful transmission of a frame. 1h (R/W) = IntPnd will be triggered after the successful transmission of a frame.
10	RXIE	R/W	0h	Receive interrupt enable 0h (R/W) = IntPnd will not be triggered after the successful reception of a frame. 1h (R/W) = IntPnd will be triggered after the successful reception of a frame.
9	RMTEN	R/W	0h	Remote enable 0h (R/W) = At the reception of a remote frame, TxRqst is not changed. 1h (R/W) = At the reception of a remote frame, TxRqst is set.
8	TXRQST	R/W	0h	Transmit request 0h (R/W) = This message object is not waiting for a transmission. 1h (R/W) = The transmission of this message object is requested and is not yet done.
7	EOB	R/W	0h	End of BlockNote: This bit is used to concatenate multiple message objects to build a FIFO Buffer. For single message objects (not belonging to a FIFO Buffer), this bit must always be set to 1. 0h (R/W) = The message object is part of a FIFO Buffer block and is not the last message object of the FIFO Buffer block. 1h (R/W) = The message object is a single message object or the last message object in a FIFO Buffer Block.
6-4	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
3-0	DLC	R/W	0h	Data length code0-8: Data frame has 0-8 data bytes. 9-15 Data frame has 8 data bytes. Note: The data length code of a message object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the message handler stores a data frame, it will write the DLC to the value given by the received message.

Table 11-341. Register Call Summary for DCAN_IF1MCTL

DCAN Functional Description • Reading Received Messages: [0]
DCAN Registers • DCAN_IF1CMD Register (Offset = 100h) [reset = 1h]: [0][1][2] • DCAN_IF1MCTL Register (Offset = 10Ch) [reset = 0h]: [0] • DCAN Registers: [0] • DCAN_IF2CMD Register (Offset = 120h) [reset = 1h]: [0][1][2]

11.2.5.42 DCAN_IF1DATA Register (Offset = 110h) [reset = 0h]

DCAN_IF1DATA is shown in Figure 11-135 and described in Table 11-343.

IF1 Data A Register
The data bytes of CAN messages are stored in the IF1/IF2 registers in the following order: In a CAN data frame, Data 0 is the first, and Data 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.
While BUSY bit of DCAN_IF1CMD/DCAN_IF2CMD register is one, IF1/IF2 register set is write protected.

Table 11-342. DCAN_IF1DATA Instances

Instance	Physical Address
DCAN_0_DATA	0260 4110h
DCAN_1_DATA	0260 8110h
DCAN_0	0260 B310h
DCAN_1	0260 B510h

Figure 11-135. DCAN_IF1DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_3								DATA_2								DATA_1								DATA_0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-343. DCAN_IF1DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DATA_3	R/W	0h	Data byte 3
23-16	DATA_2	R/W	0h	Data byte 2
15-8	DATA_1	R/W	0h	Data byte 1
7-0	DATA_0	R/W	0h	Data byte 0

Table 11-344. Register Call Summary for DCAN_IF1DATA

DCAN Functional Description • Updating a Transmit Object: [0]
DCAN Registers • DCAN_IF1DATA Register (Offset = 110h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.43 DCAN_IF1DATB Register (Offset = 114h) [reset = 0h]

DCAN_IF1DATB is shown in Figure 11-136 and described in Table 11-346.

IF1 Data B Register
The data bytes of CAN messages are stored in the IF1/IF2 registers in the following order: In a CAN data frame, Data 0 is the first, and Data 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.
While BUSY bit of DCAN_IF1CMD/DCAN_IF2CMD register is one, IF1/IF2 register set is write protected.

Table 11-345. DCAN_IF1DATB Instances

Instance	Physical Address
DCAN_0_DATA	0260 4114h
DCAN_1_DATA	0260 8114h
DCAN_0	0260 B314h
DCAN_1	0260 B514h

Figure 11-136. DCAN_IF1DATB Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_7								DATA_6								DATA_5								DATA_4							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-346. DCAN_IF1DATB Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DATA_7	R/W	0h	Data byte 7
23-16	DATA_6	R/W	0h	Data byte 6
15-8	DATA_5	R/W	0h	Data byte 5
7-0	DATA_4	R/W	0h	Data byte 4

Table 11-347. Register Call Summary for DCAN_IF1DATB

DCAN Functional Description • Updating a Transmit Object: [0]
DCAN Registers • DCAN_IF1DATB Register (Offset = 114h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.44 DCAN_IF2CMD Register (Offset = 120h) [reset = 1h]

DCAN_IF2CMD is shown in Figure 11-137 and described in Table 11-349.

IF2 Command Register
The IF2 Command Register (DCAN_IF2CMD) configure and initiate the transfer between the IF2 register set and the message RAM. It is configurable which portions of the message object should be transferred. A transfer is started when the software writes the message number to bits [7-0] MESSAGE_NUMBER. With this write operation, the BUSY bit is automatically set to 1 to indicate that a transfer is in progress. After 4 to 14 DCAN_i_VBUS_CLK (i = 0 or 1) clock cycles, the transfer between the interface register and the message RAM will be completed and the BUSY bit is cleared. The maximum number of cycles is needed when the message transfer concurs with a CAN message transmission, acceptance filtering, or message storage. If the software writes to both DCAN_IF1CMD/DCAN_IF2CMD consecutively (request of a second transfer while first transfer is still in progress), the second transfer will start after the first one has been completed.
While BUSY bit is one, IF1/IF2 register sets are write protected.
For debug support, the auto clear functionality of the IF1/IF2 command registers (clear of DMAACTIVE flag by r/w) is disabled during Debug/Suspend mode.
If an invalid Message Number is written to bits [7-0] MESSAGE_NUMBER, the message handler may access an implemented (valid) message object instead.

Table 11-348. DCAN_IF2CMD Instances

Instance	Physical Address
DCAN_0_DATA	0260 4120h
DCAN_1_DATA	0260 8120h
DCAN_0	0260 B320h
DCAN_1	0260 B520h

Figure 11-137. DCAN_IF2CMD Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
WR_RD	MASK	ARB	CONTROL	CLRINTPND	TXRQST_NEWDAT	DATA_A	DATA_B
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
BUSY	DMAACTIVE	RESERVED					
R/W-0h	R/W-0h	R-0h					
7	6	5	4	3	2	1	0
MESSAGE_NUMBER							
R/W-1h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-349. DCAN_IF2CMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
23	WR_RD	R/W	0h	Write/Read 0h (R/W) = Direction = Read: Transfer direction is from the message object addressed by MESSAGE_NUMBER to the IF2 register set. 1h (R/W) = Direction = Write: Transfer direction is from the IF2 register set to the message object addressed by MESSAGE_NUMBER.

Table 11-349. DCAN_IF2CMD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	MASK	R/W	0h	Access mask bits 0h (R/W) = Mask bits will not be changed 1h (R/W) = Direction = Write: The mask bits (identifier mask + MDir + MXtd) will be transferred from the IF2 register set to the message object addressed by MESSAGE_NUMBER.
21	ARB	R/W	0h	Access arbitration bits 0h (R/W) = Arbitration bits will not be changed 1h (R/W) = Direction = Write: The Arbitration bits (Identifier + Dir + Xtd + MsgVal) will be transferred from the IF2 register set to the message object addressed by MESSAGE_NUMBER.
20	CONTROL	R/W	0h	Access control bitsIf the TXRQST_NEWDAT bit in this register(Bit [18]) is set, the TXRQST/ NEWDAT bits in the DCAN_IF1MCTL/DCAN_IF2MCTL will be ignored. 0h (R/W) = Control bits will not be changed 1h (R/W) = Direction = Write: The message control bits will be transferred from the IF2 registerset to the message object addressed by MESSAGE_NUMBER.
19	CLRINTPND	R/W	0h	Clear interrupt pending bit 0h (R/W) = IntPnd bit will not be changed 1h (R/W) = Direction = Write: This bit is ignored. Copying of IntPnd flag from IF2 Registers to message RAM can only be controlled by the CONTROL flag (Bit [20]).
18	TXRQST_NEWDAT	R/W	0h	Access transmission request bitNote: If a CAN transmission is requested by setting TXRQST_NEWDAT in this register, the TxRqst/NewDat bits in the message object will be set to one independent of the values in DCAN_IF1MCTL/DCAN_IF2MCTL. Note: A read access to a message object can be combined with the reset of the control bits IntPnd and NewDat. The values of these bits transferred to the DCAN_IF1MCTL/DCAN_IF2MCTL always reflect the status before resetting them. 0h (R/W) = Direction = Read: NewDat bit will not be changed. Direction = Write: TxRqst/NewDat bit will be handled according to the CONTROL bit. 1h (R/W) = Direction = Read: Clears NewDat bit in the message object. 1: Direction = Write: Sets TxRqst/NewDat in message object.
17	DATA_A	R/W	0h	Access Data Bytes 0-3Note: The duration of the message transfer is independent of the number of bytes to be transferred. 0h (R/W) = Data Bytes 0-3 will not be changed. 1h (R/W) = Direction = Write: The data bytes 0-3 will be transferred from the IF2 registerset to the message object addressed by the MESSAGE_NUMBER.
16	DATA_B	R/W	0h	Access Data Bytes 4-7Note: The duration of the message transfer is independent of the number of bytes to be transferred. 0h (R/W) = Data Bytes 4-7 will not be changed. 1h (R/W) = Direction = Write: The data bytes 4-7 will be transferred from the IF2 registerset to the message object addressed by MESSAGE_NUMBER.
15	BUSY	R/W	0h	Busy flagThis bit is set to one after the message number has been written to bits [7-0] MESSAGE_NUMBER. IF2 register set will be write protected. The bit is cleared after read/write action has been finished. 0h (R/W) = No transfer between IF2 register set and message RAM is in progress. 1h (R/W) = Transfer between IF2 register set and message RAM is in progress.

Table 11-349. DCAN_IF2CMD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	DMAACTIVE	R/W	0h	Activation of DMA feature for subsequent internal IF2 update The DMA request remains active until the first read or write to one of the IF2 registers; an exception is a write to MESSAGE_NUMBER when DMAACTIVE is one. Note: Due to the auto reset feature of the DMAACTIVE bit, this bit has to be set for each subsequent DMA cycle separately. 0h (R/W) = DMA request line is independent of IF2 activities. 1h (R/W) = DMA is requested after completed transfer between IF2 register set and message RAM.
13-8	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
7-0	MESSAGE_NUMBER	R/W	1h	Number of message object in message RAM which is used for data transfer0x01-0x80: Valid message numbers. 0x81-0xFF: Invalid message numbers. 0h (R/W) = Invalid message number

Table 11-350. Register Call Summary for DCAN_IF2CMD

DCAN Functional Description • Reading Received Messages: [0] • Requesting New Data for a Receive Object: [0] • DMA Functionality: [0] • Debug/Suspend Mode: [0] • Interrupt Functionality: [0] • Changing a Transmit Object: [0][1] • Updating a Transmit Object: [0] • Transmission of Messages in Event Driven CAN Communication: [0] • Message Interface Register Sets 1 and 2: [0][1] • Message Interface Register Sets: [0][1]
DCAN Registers • DCAN_IF1DATA Register (Offset = 110h) [reset = 0h]: [0] • DCAN_IF1DATB Register (Offset = 114h) [reset = 0h]: [0] • DCAN_IF2CMD Register (Offset = 120h) [reset = 1h]: [0][1][2] • DCAN_IF2MCTL Register (Offset = 12Ch) [reset = 0h]: [0] • DCAN_IF1ARB Register (Offset = 108h) [reset = 0h]: [0] • DCAN_IF1MCTL Register (Offset = 10Ch) [reset = 0h]: [0] • DCAN_IF2MSK Register (Offset = 124h) [reset = FFFFFFFFh]: [0] • DCAN_IF1MSK Register (Offset = 104h) [reset = FFFFFFFFh]: [0] • DCAN_IF1CMD Register (Offset = 100h) [reset = 1h]: [0] • DCAN_IF2ARB Register (Offset = 128h) [reset = 0h]: [0] • DCAN Registers: [0] • DCAN_IF2DATB Register (Offset = 134h) [reset = 0h]: [0] • DCAN_IF2DATA Register (Offset = 130h) [reset = 0h]: [0]

11.2.5.45 DCAN_IF2MSK Register (Offset = 124h) [reset = FFFFFFFFh]

DCAN_IF2MSK is shown in Figure 11-138 and described in Table 11-352.

IF2 Mask Register
The bits of the IF1/IF2 mask registers mirror the mask bits of a message object. The function of the relevant message objects bits is described in *Structure of Message Objects*.
While BUSY bit of DCAN_IF1CMD/DCAN_IF2CMD register is one, IF1/IF2 register set is write protected.

Table 11-351. DCAN_IF2MSK Instances

Instance	Physical Address
DCAN_0_DATA	0260 4124h
DCAN_1_DATA	0260 8124h
DCAN_0	0260 B324h
DCAN_1	0260 B524h

Figure 11-138. DCAN_IF2MSK Register

31	30	29	28	27	26	25	24
MXTD	MDIR	RESERVED	MSK				
R/W-1h	R/W-1h	R-1h	R/W-1FFFFFFFh				
23	22	21	20	19	18	17	16
MSK							
R/W-1FFFFFFFh							
15	14	13	12	11	10	9	8
MSK							
R/W-1FFFFFFFh							
7	6	5	4	3	2	1	0
MSK							
R/W-1FFFFFFFh							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-352. DCAN_IF2MSK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MXTD	R/W	1h	Mask Extended IdentifierWhen 11-bit (standard) identifiers are used for a message object, the identifiers of received data frames are written into bits ID[28-18]. For acceptance filtering, only these bits together with mask bits Msk[28-18] are considered. 0h (R/W) = The extended identifier bit (IDE) has no effect on the acceptance filtering. 1h (R/W) = The extended identifier bit (IDE) is used for acceptance filtering.
30	MDIR	R/W	1h	Mask Message Direction 0h (R/W) = The message direction bit (Dir) has no effect on the acceptance filtering. 1h (R/W) = The message direction bit (Dir) is used for acceptance filtering.
29	RESERVED	R	1h	This bit is always read as 1. Writes have no effect.
28-0	MSK	R/W	1FFFFFFFh	Identifier Mask 0h (R/W) = The corresponding bit in the identifier of the message object is not used for acceptance filtering (don't care). 1h (R/W) = The corresponding bit in the identifier of the message object is used for acceptance filtering.

Table 11-353. Register Call Summary for DCAN_IF2MSK

DCAN Registers
• DCAN_IF2MSK Register (Offset = 124h) [reset = FFFFFFFFh]: [0] • DCAN Registers: [0]

11.2.5.46 DCAN_IF2ARB Register (Offset = 128h) [reset = 0h]

DCAN_IF2ARB is shown in Figure 11-139 and described in Table 11-355.

IF2 arbitration register
The Arbitration bits ID[28-0], XTD, and DIR are used to define the identifier and type of outgoing messages and (together with the mask bits MSK[28-0], MXTD, and MDIR) for acceptance filtering of incoming messages. A received message is stored into the valid message object with matching identifier and Direction = receive (data frame) or Direction = transmit (remote frame). Extended frames can be stored only in message objects with Xtd = 1, standard frames in message objects with Xtd = 0. If a received message (data frame or remote frame) matches more than one valid message objects, it is stored into the one with the lowest message number.
The bits of the IF1/IF2 arbitration registers mirror the arbitration bits of a message object. The function of the relevant message objects bits is described in *Structure of Message Objects*
While BUSY bit of DCAN_IF1CMD/DCAN_IF2CMD register is one, IF1/IF2 register set is write protected.

Table 11-354. DCAN_IF2ARB Instances

Instance	Physical Address
DCAN_0_DATA	0260 4128h
DCAN_1_DATA	0260 8128h
DCAN_0	0260 B328h
DCAN_1	0260 B528h

Figure 11-139. DCAN_IF2ARB Register

31	30	29	28	27	26	25	24
MSGVAL	XTD	DIR	ID				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				
23	22	21	20	19	18	17	16
ID							
R/W-0h							
15	14	13	12	11	10	9	8
ID							
R/W-0h							
7	6	5	4	3	2	1	0
ID							
R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-355. DCAN_IF2ARB Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MSGVAL	R/W	0h	Message validThe software should reset the MsgVal bit of all unused Messages Objects during the initialization before it resets bit INIT in the DCAN_CTL. This bit must also be reset if the messages object is no longer required. 0h (R/W) = The message object is ignored by the message handler. 1h (R/W) = The message object is to be used by the message handler.
30	XTD	R/W	0h	Extended identifier 0h (R/W) = The 11-bit (standard) Identifier is used for this message object. 1h (R/W) = The 29-bit (extended) Identifier is used for this message object.

Table 11-355. DCAN_IF2ARB Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
29	DIR	R/W	0h	Message direction 0h (R/W) = Direction = receive: On TxRqst, a remote frame with the identifier of this message object is transmitted. On reception of a data frame with matching identifier, this message is stored in this message object. 1h (R/W) = Direction = transmit: On TxRqst, the respective message object is transmitted as a data frame. On reception of a remote frame with matching identifier, the TxRqst bit of this message object is set (if RmtEn = 1).
28-0	ID	R/W	0h	Message identifierID[28-0]: 29-bit identifier (extended frame). ID[28-18]: 11-bit identifier (standard frame).

Table 11-356. Register Call Summary for DCAN_IF2ARB

DCAN Functional Description • Configuration of a Single Receive Object for Data Frames: [0]
DCAN Registers • DCAN_IF2ARB Register (Offset = 128h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.47 DCAN_IF2MCTL Register (Offset = 12Ch) [reset = 0h]

DCAN_IF2MCTL is shown in Figure 11-140 and described in Table 11-358.

IF2 Message Control Register
The bits of the IF1/IF2 message control registers mirror the message control bits of a message object. The function of the relevant message objects bits is described in *Structure of Message Objects*
While BUSY bit of DCAN_IF1CMD/DCAN_IF2CMD register is one, IF1/IF2 register set is write protected.

Table 11-357. DCAN_IF2MCTL Instances

Instance	Physical Address
DCAN_0_DATA	0260 412Ch
DCAN_1_DATA	0260 812Ch
DCAN_0	0260 B32Ch
DCAN_1	0260 B52Ch

Figure 11-140. DCAN_IF2MCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
NEWDAT	MSGLST	INTPND	UMASK	TXIE	RXIE	RMTEN	TXRQST
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EOB	RESERVED			DLC			
R/W-0h	R-0h			R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-358. DCAN_IF2MCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
15	NEWDAT	R/W	0h	New data 0h (R/W) = No new data has been written into the data portion of this message object by the message handler since the last time when this flag was cleared by the software. 1h (R/W) = The message handler or the software has written new data into the data portion of this message object.
14	MSGLST	R/W	0h	Message lost (only valid for message objects with direction = receive) 0h (R/W) = No message lost since the last time when this bit was reset by the software. 1h (R/W) = The message handler stored a new message into this object when NewDat was still set, so the previous message has been overwritten.
13	INTPND	R/W	0h	Interrupt pending 0h (R/W) = This message object is not the source of an interrupt. 1h (R/W) = This message object is the source of an interrupt. The Interrupt Identifier in DCAN_INT will point to this message object if there is no other interrupt source with higher priority.

Table 11-358. DCAN_IF2MCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	UMASK	R/W	0h	Use acceptance maskIf the UMask bit is set to one, the message object's mask bits have to be programmed during initialization of the message object before MsgVal is set to one. 0h (R/W) = Mask ignored 1h (R/W) = Use mask (Msk[28-0], MXtd, and MDir) for acceptance filtering
11	TXIE	R/W	0h	Transmit interrupt enable 0h (R/W) = IntPnd will not be triggered after the successful transmission of a frame. 1h (R/W) = IntPnd will be triggered after the successful transmission of a frame.
10	RXIE	R/W	0h	Receive interrupt enable 0h (R/W) = IntPnd will not be triggered after the successful reception of a frame. 1h (R/W) = IntPnd will be triggered after the successful reception of a frame.
9	RMTEN	R/W	0h	Remote enable 0h (R/W) = At the reception of a remote frame, TxRqst is not changed. 1h (R/W) = At the reception of a remote frame, TxRqst is set.
8	TXRQST	R/W	0h	Transmit request 0h (R/W) = This message object is not waiting for a transmission. 1h (R/W) = The transmission of this message object is requested and is not yet done.
7	EOB	R/W	0h	End of BlockNote: This bit is used to concatenate multiple message objects to build a FIFO Buffer. For single message objects (not belonging to a FIFO Buffer), this bit must always be set to one. 0h (R/W) = The message object is part of a FIFO Buffer block and is not the last message object of the FIFO Buffer block. 1h (R/W) = The message object is a single message object or the last message object in a FIFO Buffer Block.
6-4	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
3-0	DLC	R/W	0h	Data length code0-8: Data frame has 0-8 data bytes. 9-15 Data frame has 8 data bytes. Note: The data length code of a message object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the message handler stores a data frame, it will write the DLC to the value given by the received message.

Table 11-359. Register Call Summary for DCAN_IF2MCTL

DCAN Functional Description • Reading Received Messages: [0]
DCAN Registers • DCAN_IF2MCTL Register (Offset = 12Ch) [reset = 0h]: [0] • DCAN_IF2CMD Register (Offset = 120h) [reset = 1h]: [0][1][2] • DCAN Registers: [0]

11.2.5.48 DCAN_IF2DATA Register (Offset = 130h) [reset = 0h]

DCAN_IF2DATA is shown in Figure 11-141 and described in Table 11-361.

IF2 Data A Register
The data bytes of CAN messages are stored in the IF1/IF2 registers in the following order: In a CAN data frame, Data 0 is the first, and Data 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.
While BUSY bit of DCAN_IF1CMD/DCAN_IF2CMD register is one, IF1/IF2 register set is write protected.

Table 11-360. DCAN_IF2DATA Instances

Instance	Physical Address
DCAN_0_DATA	0260 4130h
DCAN_1_DATA	0260 8130h
DCAN_0	0260 B330h
DCAN_1	0260 B530h

Figure 11-141. DCAN_IF2DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_3								DATA_2								DATA_1								DATA_0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-361. DCAN_IF2DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DATA_3	R/W	0h	Data byte 3
23-16	DATA_2	R/W	0h	Data byte 2
15-8	DATA_1	R/W	0h	Data byte 1
7-0	DATA_0	R/W	0h	Data byte 0

Table 11-362. Register Call Summary for DCAN_IF2DATA

DCAN Functional Description • Updating a Transmit Object: [0]
DCAN Registers • DCAN_IF2DATA Register (Offset = 130h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.49 DCAN_IF2DATB Register (Offset = 134h) [reset = 0h]

DCAN_IF2DATB is shown in Figure 11-142 and described in Table 11-364.

IF2 Data B Register
The data bytes of CAN messages are stored in the IF1/IF2 registers in the following order: In a CAN data frame, Data 0 is the first, and Data 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.
While BUSY bit of DCAN_IF1CMD/DCAN_IF2CMD register is one, IF1/IF2 register set is write protected.

Table 11-363. DCAN_IF2DATB Instances

Instance	Physical Address
DCAN_0_DATA	0260 4134h
DCAN_1_DATA	0260 8134h
DCAN_0	0260 B334h
DCAN_1	0260 B534h

Figure 11-142. DCAN_IF2DATB Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_7								DATA_6								DATA_5								DATA_4							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-364. DCAN_IF2DATB Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DATA_7	R/W	0h	Data byte 7
23-16	DATA_6	R/W	0h	Data byte 6
15-8	DATA_5	R/W	0h	Data byte 5
7-0	DATA_4	R/W	0h	Data byte 4

Table 11-365. Register Call Summary for DCAN_IF2DATB

DCAN Functional Description • Updating a Transmit Object: [0]
DCAN Registers • DCAN_IF2DATB Register (Offset = 134h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.50 DCAN_IF3OBS Register (Offset = 140h) [reset = 0h]

DCAN_IF3OBS is shown in Figure 11-143 and described in Table 11-367.

IF3 Observation Register

The IF3 register set can automatically be updated with received message objects without the need to initiate the transfer from message RAM by software (Additional information can be found in *Structure of Message Objects*). The observation flags (Bits [4-0]) are used to determine, which data sections of the IF3 interface register set have to be read in order to complete a DMA read cycle. After all marked data sections are read, the DCAN is enabled to update the IF3 interface register set with new data. Any access order of single bytes or half-words is supported. When using byte or half-word accesses, a data section is marked as completed, if all bytes are read.
NOTE: If IF3 Update Enable is used and no Observation flag is set, the corresponding message objects will be copied to IF3 without activating the DMA request line and without waiting for DMA read accesses. A write access to this register aborts a pending DMA cycle by resetting the DMA line and enables updating of IF3 interface register set with new data. To avoid data inconsistency, the DMA controller should be disabled before reconfiguring IF3 observation register. The status of the current read-cycle can be observed via status flags (Bits [12-8]).

Table 11-366. DCAN_IF3OBS Instances

Instance	Physical Address
DCAN_0_DATA	0260 4140h
DCAN_1_DATA	0260 8140h
DCAN_0	0260 B340h
DCAN_1	0260 B540h

Figure 11-143. DCAN_IF3OBS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
IF3_UPD	RESERVED		IF3_SDB	IF3_SDA	IF3_SC	IF3_SA	IF3_SM
R-0h	R-0h		R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED			DATAB	DATAA	CTRL	ARB	MASK
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-367. DCAN_IF3OBS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
15	IF3_UPD	R	0h	IF3 Update Data 0h (R) = No new data has been loaded since last IF3 read. 1h (R) = New data has been loaded since last IF3 read.
14-13	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
12	IF3_SDB	R	0h	IF3 Status of Data B read access 0h (R) = All Data B bytes are already read out, or are not marked to be read. 1h (R) = Data B section has still data to be read out.

Table 11-367. DCAN_IF3OBS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	IF3_SDA	R	0h	IF3 Status of Data A read access 0h (R) = All Data A bytes are already read out, or are not marked to be read. 1h (R) = Data A section has still data to be read out.
10	IF3_SC	R	0h	IF3 Status of control bits read access 0h (R) = All control section bytes are already read out, or are not marked to be read. 1h (R) = Control section has still data to be read out.
9	IF3_SA	R	0h	IF3 Status of Arbitration data read access 0h (R) = All Arbitration data bytes are already read out, or are not marked to be read. 1h (R) = Arbitration section has still data to be read out.
8	IF3_SM	R	0h	IF3 Status of Mask data read access 0h (R) = All mask data bytes are already read out, or are not marked to be read. 1h (R) = Mask section has still data to be read out.
7-5	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
4	DATAB	R/W	0h	Data B read observation 0h (R/W) = Data B section has not to be read. 1h (R/W) = Data B section has to be read to enable next IF3 update.
3	DATAA	R/W	0h	Data A read observation 0h (R/W) = Data A section has not to be read. 1h (R/W) = Data A section has to be read to enable next IF3 update.
2	CTRL	R/W	0h	Ctrl read observation 0h (R/W) = Ctrl section has not to be read. 1h (R/W) = Ctrl section has to be read to enable next IF3 update.
1	ARB	R/W	0h	Arbitration data read observation 0h (R/W) = Arbitration data has not to be read. 1h (R/W) = Arbitration data has to be read to enable next IF3 update.
0	MASK	R/W	0h	Mask data read observation 0h (R/W) = Mask data has not to be read. 1h (R/W) = Mask data has to be read to enable next IF3 update.

Table 11-368. Register Call Summary for DCAN_IF3OBS

DCAN Functional Description • IF3 Register Set: [0]
DCAN Registers • DCAN_IF3OBS Register (Offset = 140h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.51 DCAN_IF3MSK Register (Offset = 144h) [reset = FFFFFFFFh]

DCAN_IF3MSK is shown in Figure 11-144 and described in Table 11-370.

IF3 Mask Register

Table 11-369. DCAN_IF3MSK Instances

Instance	Physical Address
DCAN_0_DATA	0260 4144h
DCAN_1_DATA	0260 8144h
DCAN_0	0260 B344h
DCAN_1	0260 B544h

Figure 11-144. DCAN_IF3MSK Register

31	30	29	28	27	26	25	24
MXTD	MDIR	RESERVED	MSK				
R-1h	R-1h	R-1h	R/W-1FFFFFFFh				
23	22	21	20	19	18	17	16
MSK							
R/W-1FFFFFFFh							
15	14	13	12	11	10	9	8
MSK							
R/W-1FFFFFFFh							
7	6	5	4	3	2	1	0
MSK							
R/W-1FFFFFFFh							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-370. DCAN_IF3MSK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MXTD	R	1h	Mask Extended IdentifierWhen 11-bit (standard) identifiers are used for a message object, the identifiers of received data frames are written into bits ID[28-18]. For acceptance filtering, only these bits together with mask bits Msk[28-18] are considered. 0h (R) = The extended identifier bit (IDE) has no effect on the acceptance filtering. 1h (R) = The extended identifier bit (IDE) is used for acceptance filtering.
30	MDIR	R	1h	Mask Message Direction 0h (R) = The message direction bit (Dir) has no effect on the acceptance filtering. 1h (R) = The message direction bit (Dir) is used for acceptance filtering.
29	RESERVED	R	1h	These bits are always read as 1. Writes have no effect.
28-0	MSK	R/W	1FFFFFFFh	Identifier Mask 0h (R/W) = The corresponding bit in the identifier of the message object is not used for acceptance filtering (don't care). 1h (R/W) = The corresponding bit in the identifier of the message object is used for acceptance filtering.

Table 11-371. Register Call Summary for DCAN_IF3MSK

DCAN Registers
- DCAN_IF3MSK Register (Offset = 144h) [reset = FFFFFFFFh]: [0]
- DCAN Registers: [0]

11.2.5.52 DCAN_IF3ARB Register (Offset = 148h) [reset = 0h]

DCAN_IF3ARB is shown in Figure 11-145 and described in Table 11-373.

IF3 Arbitration Register

Table 11-372. DCAN_IF3ARB Instances

Instance	Physical Address
DCAN_0_DATA	0260 4148h
DCAN_1_DATA	0260 8148h
DCAN_0	0260 B348h
DCAN_1	0260 B548h

Figure 11-145. DCAN_IF3ARB Register

31	30	29	28	27	26	25	24
MSGVAL	XTD	DIR	ID				
R-0h	R-0h	R-0h	R-0h				
23	22	21	20	19	18	17	16
ID							
R-0h							
15	14	13	12	11	10	9	8
ID							
R-0h							
7	6	5	4	3	2	1	0
ID							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-373. DCAN_IF3ARB Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MSGVAL	R	0h	Message ValidThe software should reset the MsgVal bit of all unused Messages Objects during the initialization before it resets bit INIT in the DCAN_CTL. This bit must also be reset before the identifier ID[28-0], the control bits Xtd, Dir or DLC[3-0] are modified, or if the messages object is no longer required. 0h (R) = The message object is ignored by the message handler. 1h (R) = The message object is to be used by the message handler.
30	XTD	R	0h	Extended Identifier 0h (R) = The 11-bit (standard) Identifier is used for this message object. 1h (R) = The 29-bit (extended) Identifier is used for this message object.
29	DIR	R	0h	Message Direction 0h (R) = Direction = receive: On TxRqst, a remote frame with the identifier of this message object is transmitted. On reception of a data frame with matching identifier, this message is stored in this message object. 1h (R) = Direction = transmit: On TxRqst, the respective message object is transmitted as a data frame. On reception of a remote frame with matching identifier, the TxRqst bit of this message object is set (if RmtEn = 1).
28-0	ID	R	0h	Message IdentifierID[28-0]: 29-bit Identifier (extended frame). ID[28-18]: 11-bit Identifier (standard frame).

Table 11-374. Register Call Summary for DCAN_IF3ARB

DCAN Functional Description • Configuration of a Single Receive Object for Data Frames: [0]
DCAN Registers • DCAN Registers: [0] • DCAN_IF3ARB Register (Offset = 148h) [reset = 0h]: [0]

11.2.5.53 DCAN_IF3MCTL Register (Offset = 14Ch) [reset = 0h]

DCAN_IF3MCTL is shown in Figure 11-146 and described in Table 11-376.

IF3 Message Control Register

Table 11-375. DCAN_IF3MCTL Instances

Instance	Physical Address
DCAN_0_DATA	0260 414Ch
DCAN_1_DATA	0260 814Ch
DCAN_0	0260 B34Ch
DCAN_1	0260 B54Ch

Figure 11-146. DCAN_IF3MCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
NEWDAT	MSGLST	INTPND	UMASK	TXIE	RXIE	RMTEN	TXRQST
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
EOB	RESERVED			DLC			
R-0h	R-0h			R-0h			

LEGEND: R = Read Only; *-n* = value after reset

Table 11-376. DCAN_IF3MCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
15	NEWDAT	R	0h	New Data 0h (R) = No new data has been written into the data portion of this message object by the message handler since the last time when this flag was cleared by the software. 1h (R) = The message handler or the software has written new data into the data portion of this message object.
14	MSGLST	R	0h	Message Lost (only valid for message objects with direction = receive) 0h (R) = No message lost since the last time when this bit was reset by the software. 1h (R) = The message handler stored a new message into this object when NewDat was still set, so the previous message has been overwritten.
13	INTPND	R	0h	Interrupt Pending 0h (R) = This message object is not the source of an interrupt. 1h (R) = This message object is the source of an interrupt. The Interrupt Identifier in DCAN_INT will point to this message object if there is no other interrupt source with higher priority.
12	UMASK	R	0h	Use Acceptance MaskIf the UMASK bit is set to one, the message object's mask bits have to be programmed during initialization of the message object before MsgVal is set to one. 0h (R) = Mask ignored 1h (R) = Use mask (Msk[28-0], MXtd, and MDir) for acceptance filtering

Table 11-376. DCAN_IF3MCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	TXIE	R	0h	Transmit Interrupt enable 0h (R) = IntPnd will not be triggered after the successful transmission of a frame. 1h (R) = IntPnd will be triggered after the successful transmission of a frame.
10	RXIE	R	0h	Receive Interrupt enable 0h (R) = IntPnd will not be triggered after the successful reception of a frame. 1h (R) = IntPnd will be triggered after the successful reception of a frame.
9	RMTEN	R	0h	Remote enable 0h (R) = At the reception of a remote frame, TxRqst is not changed. 1h (R) = At the reception of a remote frame, TxRqst is set.
8	TXRQST	R	0h	Transmit Request 0h (R) = This message object is not waiting for a transmission. 1h (R) = The transmission of this message object is requested and is not yet done.
7	EOB	R	0h	End of BlockNote: This bit is used to concatenate multiple message objects to build a FIFO Buffer. For single message objects (not belonging to a FIFO Buffer), this bit must always be set to one. 0h (R) = The message object is part of a FIFO Buffer block and is not the last message object of the FIFO Buffer block. 1h (R) = The message object is a single message object or the last message object in a FIFO Buffer Block.
6-4	RESERVED	R	0h	These bits are always read as 0. Writes have no effect.
3-0	DLC	R	0h	Data Length Code0-8: Data frame has 0-8 data bits. 9-15: Data frame has 8 data bytes. Note: The data length code of a message object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the message handler stores a data frame, it will write the DLC to the value given by the received message.

Table 11-377. Register Call Summary for DCAN_IF3MCTL

DCAN Functional Description • Reading Received Messages: [0]
DCAN Registers • DCAN_IF3MCTL Register (Offset = 14Ch) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.54 DCAN_IF3DATA Register (Offset = 150h) [reset = 0h]

DCAN_IF3DATA is shown in Figure 11-147 and described in Table 11-379.

IF3 Data A
The data bytes of CAN messages are stored in the IF3 registers in the following order: In a CAN data frame, Data 0 is the first, and Data 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.

Table 11-378. DCAN_IF3DATA Instances

Instance	Physical Address
DCAN_0_DATA	0260 4150h
DCAN_1_DATA	0260 8150h
DCAN_0	0260 B350h
DCAN_1	0260 B550h

Figure 11-147. DCAN_IF3DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_3								DATA_2								DATA_1								DATA_0							
R-0h								R-0h								R-0h								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-379. DCAN_IF3DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DATA_3	R	0h	Data byte 3
23-16	DATA_2	R	0h	Data byte 2
15-8	DATA_1	R	0h	Data byte 1
7-0	DATA_0	R	0h	Data byte 0

Table 11-380. Register Call Summary for DCAN_IF3DATA

DCAN Registers
- DCAN_IF3DATA Register (Offset = 150h) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.55 DCAN_IF3DATB Register (Offset = 154h) [reset = 0h]

DCAN_IF3DATB is shown in Figure 11-148 and described in Table 11-382.

IF3 Data B
The data bytes of CAN messages are stored in the IF3 registers in the following order: In a CAN data frame, Data 0 is the first, and Data 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.

Table 11-381. DCAN_IF3DATB Instances

Instance	Physical Address
DCAN_0_DATA	0260 4154h
DCAN_1_DATA	0260 8154h
DCAN_0	0260 B354h
DCAN_1	0260 B554h

Figure 11-148. DCAN_IF3DATB Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_7								DATA_6								DATA_5								DATA_4							
R-0h								R-0h								R-0h								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-382. DCAN_IF3DATB Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DATA_7	R	0h	Data byte 7
23-16	DATA_6	R	0h	Data byte 6
15-8	DATA_5	R	0h	Data byte 5
7-0	DATA_4	R	0h	Data byte 4

Table 11-383. Register Call Summary for DCAN_IF3DATB

DCAN Registers
- DCAN_IF3DATB Register (Offset = 154h) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.56 DCAN_IF3UPD12 Register (Offset = 160h) [reset = 0h]

DCAN_IF3UPD12 is shown in Figure 11-149 and described in Table 11-385.

Update Enable Register
The automatic update functionality of the IF3 register set can be configured for each message object. A message object is enabled for automatic IF3 update, if the dedicated IF3UPDEN flag is set. This means that an active NewDat flag of this message object (for example due to reception of a CAN frame) will trigger an automatic copy of the whole message object to IF3 register set
NOTE: IF3 Update enable should not be set for transmit objects.

Table 11-384. DCAN_IF3UPD12 Instances

Instance	Physical Address
DCAN_0_DATA	0260 4160h
DCAN_1_DATA	0260 8160h
DCAN_0	0260 B360h
DCAN_1	0260 B560h

Figure 11-149. DCAN_IF3UPD12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IF3UPDEN																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-385. DCAN_IF3UPD12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IF3UPDEN	R/W	0h	IF3 Update Enabled (for 1-32 message objects) 0h (R/W) = Automatic IF3 update is disabled for this message object. 1h (R/W) = Automatic IF3 update is enabled for this message object. A message object is scheduled to be copied to IF3 register set, if NewDat flag of the message object is active.

Table 11-386. Register Call Summary for DCAN_IF3UPD12

DCAN Functional Description • IF3 Register Set: [0]
DCAN Registers • DCAN_IF3UPD12 Register (Offset = 160h) [reset = 0h]: [0] • DCAN Registers: [0]

11.2.5.57 DCAN_IF3UPD34 Register (Offset = 164h) [reset = 0h]

DCAN_IF3UPD34 is shown in Figure 11-150 and described in Table 11-388.

Update Enable Register
The automatic update functionality of the IF3 register set can be configured for each message object. A message object is enabled for automatic IF3 update, if the dedicated IF3UPDEN flag is set. This means that an active NewDat flag of this message object (for example due to reception of a CAN frame) will trigger an automatic copy of the whole message object to IF3 register set
NOTE: IF3 Update enable should not be set for transmit objects.

Table 11-387. DCAN_IF3UPD34 Instances

Instance	Physical Address
DCAN_0_DATA	0260 4164h
DCAN_1_DATA	0260 8164h
DCAN_0	0260 B364h
DCAN_1	0260 B564h

Figure 11-150. DCAN_IF3UPD34 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IF3UPDEN																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-388. DCAN_IF3UPD34 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IF3UPDEN	R/W	0h	IF3 Update Enabled (for 33-64 message objects) 0h (R/W) = Automatic IF3 update is disabled for this message object. 1h (R/W) = Automatic IF3 update is enabled for this message object. A message object is scheduled to be copied to IF3 register set, if NewDat flag of the message object is active.

Table 11-389. Register Call Summary for DCAN_IF3UPD34

DCAN Registers
- DCAN_IF3UPD34 Register (Offset = 164h) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.58 DCAN_IF3UPD56 Register (Offset = 168h) [reset = 0h]

DCAN_IF3UPD56 is shown in Figure 11-151 and described in Table 11-391.

Update Enable Register
The automatic update functionality of the IF3 register set can be configured for each message object. A message object is enabled for automatic IF3 update, if the dedicated IF3UPDEN flag is set. This means that an active NewDat flag of this message object (for example due to reception of a CAN frame) will trigger an automatic copy of the whole message object to IF3 register set
NOTE: IF3 Update enable should not be set for transmit objects.

Table 11-390. DCAN_IF3UPD56 Instances

Instance	Physical Address
DCAN_0_DATA	0260 4168h
DCAN_1_DATA	0260 8168h
DCAN_0	0260 B368h
DCAN_1	0260 B568h

Figure 11-151. DCAN_IF3UPD56 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IF3UPDEN																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-391. DCAN_IF3UPD56 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IF3UPDEN	R/W	0h	IF3 Update Enabled (for 65-96 message objects) 0h (R/W) = Automatic IF3 update is disabled for this message object. 1h (R/W) = Automatic IF3 update is enabled for this message object. A message object is scheduled to be copied to IF3 register set, if NewDat flag of the message object is active.

Table 11-392. Register Call Summary for DCAN_IF3UPD56

DCAN Registers
- DCAN_IF3UPD56 Register (Offset = 168h) [reset = 0h]: [0]
- DCAN Registers: [0]

11.2.5.59 DCAN_IF3UPD78 Register (Offset = 16Ch) [reset = 0h]

DCAN_IF3UPD78 is shown in Figure 11-152 and described in Table 11-394.

Update Enable Register
The automatic update functionality of the IF3 register set can be configured for each message object. A message object is enabled for automatic IF3 update, if the dedicated IF3UPDEN flag is set. This means that an active NewDat flag of this message object (for example due to reception of a CAN frame) will trigger an automatic copy of the whole message object to IF3 register set
NOTE: IF3 Update enable should not be set for transmit objects.

Table 11-393. DCAN_IF3UPD78 Instances

Instance	Physical Address
DCAN_0_DATA	0260 416Ch
DCAN_1_DATA	0260 816Ch
DCAN_0	0260 B36Ch
DCAN_1	0260 B56Ch

Figure 11-152. DCAN_IF3UPD78 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IF3UPDEN																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-394. DCAN_IF3UPD78 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IF3UPDEN	R/W	0h	IF3 Update Enabled (for 97-128 message objects) 0h (R/W) = Automatic IF3 update is disabled for this message object. 1h (R/W) = Automatic IF3 update is enabled for this message object. A message object is scheduled to be copied to IF3 register set, if NewDat flag of the message object is active.

Table 11-395. Register Call Summary for DCAN_IF3UPD78

DCAN Functional Description • IF3 Register Set: [0]
DCAN Registers • DCAN_IF3UPD78 Register (Offset = 16Ch) [reset = 0h]: [0] • DCAN Registers: [0]

11.3 Display Subsystem (DSS)

This section describes the Display Subsytem (DSS) in the device.

NOTE: The supported set of features and peripherals is device part number dependent. For more information, see the device Data Manual.

11.3.1 DSS Overview

The Display Subsystem (DSS) provides the logic to interface display peripherals. DSS includes a DMA engine as part of the integrated Display Controller (DISPC) module, which allows direct access to the frame buffer (system memory). Various pixel processing capabilities are supported, such as: color space conversion, filtering, scaling, among others.

The supported display interfaces (connections to external panel) are:

- One parallel interface, which can be used for MIPI® DPI 2.0, or BT-656 or BT-1120.
- One RFBI interface, supporting MIPI DBI 2.0.

The modules integrated in DSS are:

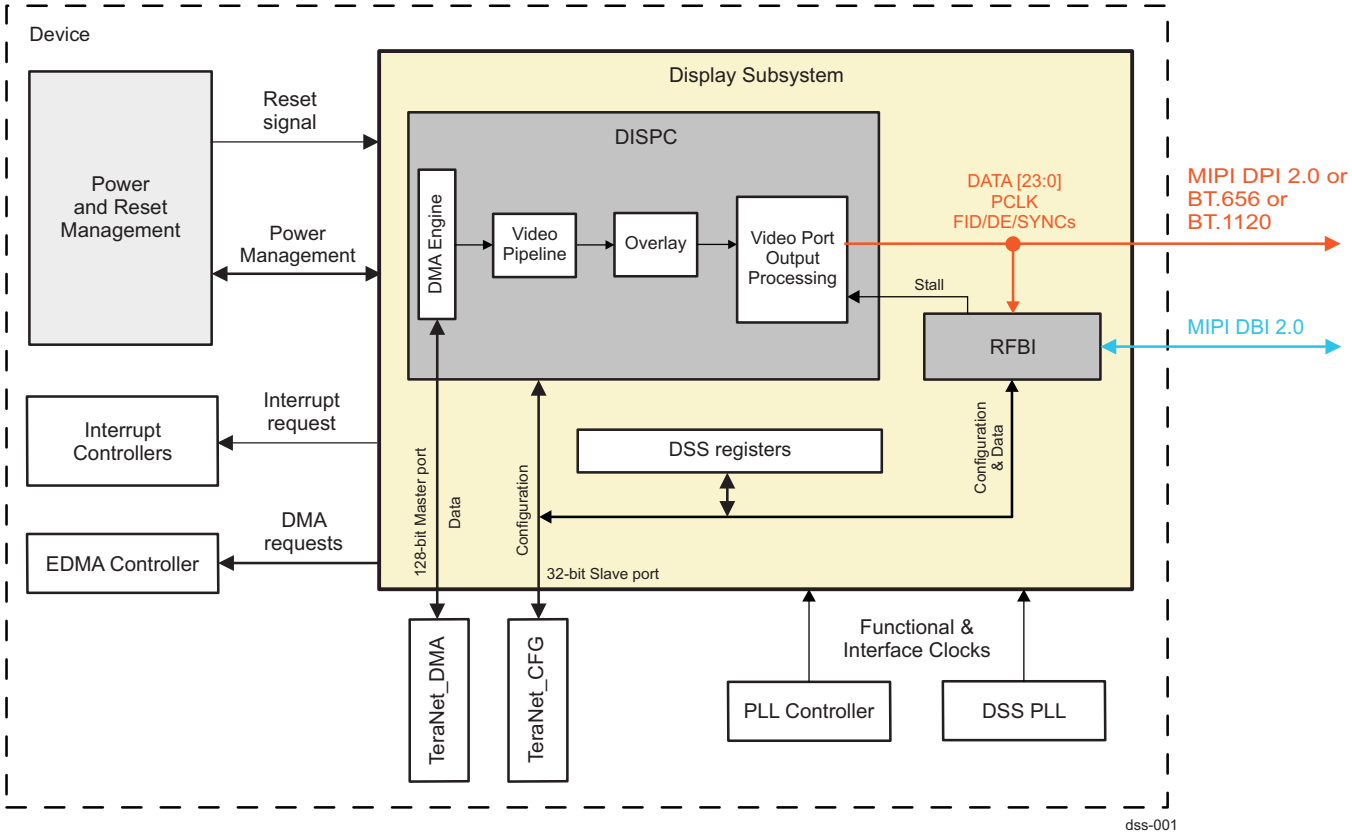
- Display Controller (DISPC), with the following main features
  - One Direct Memory Access (DMA) engine
  - One Video Pipeline (VID1), including VC1 Range Mapping Unit, Scaler, and Color Space Converter (CSC)
  - One Overlay Manager (OVR1)
  - Active Matrix color support for 12/16/18/24-bit (truncated or dithered encoded pixel values)
  - One Video Port (VP1) with programmable timing generator to support up to 148.5 MHz pixel clock video formats defined in CEA-861-E and VESA DMT standards
  - Supported maximum FrameBuffer width of 4096 for all pixel formats
  - Configurable output mode: progressive or interlaced
  - Selection between RGB and YUV422 output pixel formats (YUV422 only available when BT-656 or BT-1120 output mode is enabled on the parallel interface)
  - Stall Mode support for RFBI
- Remote Frame Buffer Interface (RFBI) module, with the following main features:
  - Access to RFB direct “ARMSS interface”
    - Sending commands and data to the RFB panel, received from DISPC or from ARMSS (through the 32-bit interconnect slave port)
    - Reading data/status from the RFB through the 32-bit interconnect slave port
  - RFB interface:
    - 8/9/12/16-bit data bus (for up to QVGA @30fps)
    - Two programmable configurations for two peripheral devices connected to the RFBI module
    - Tearing Effect control logic: Horizontal Synchronization (HSync) and Vertical Synchronization (VSync) embedded in a single signal (TE) or using two signals (HS+VS)
    - Programmable pixel memory and output formats

DSS provides two interfaces to SoC interconnect:

- One 128-bit master port (with MFLAG support). The DMA engine in DISPC uses this single master port to read data from SoC system memory.
- One 32-bit slave port. Used for configuration of the memory mapped registers inside DSS. It is further connected internally to DISPC and RFBI modules.

Figure 11-153 is a high-level overview of the display subsystem.

Figure 11-153. DSS Overview

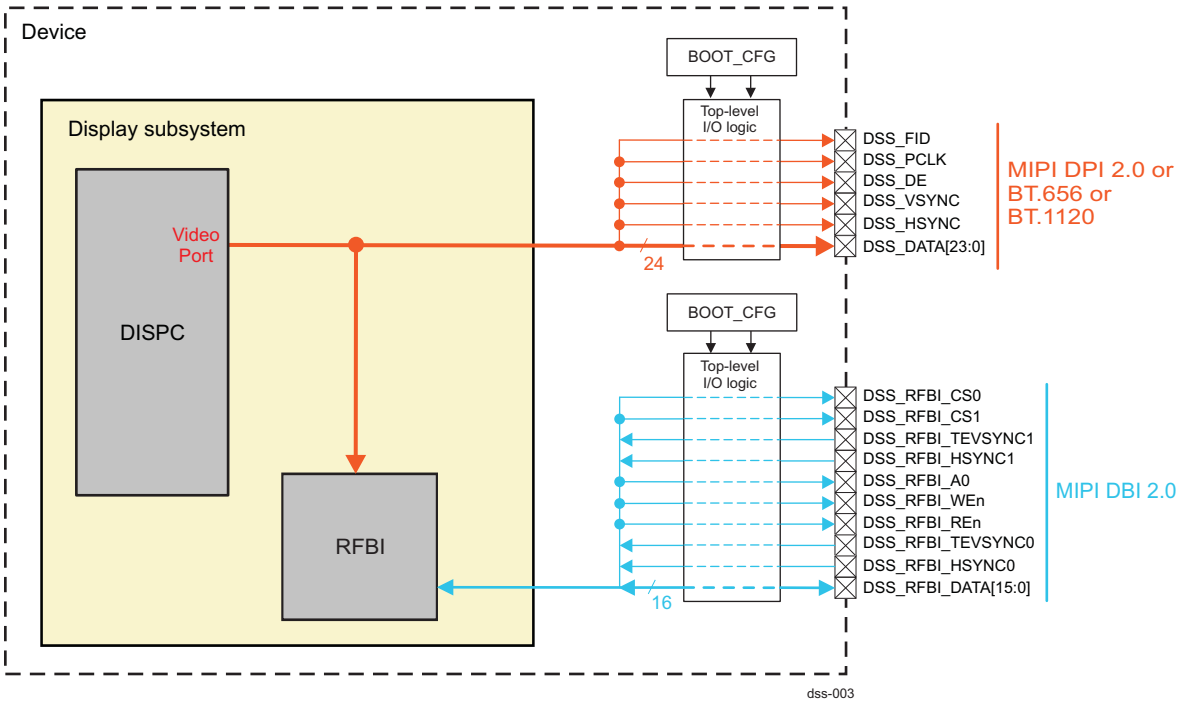


11.3.2 DSS Environment

This section describes the various interfaces handled by the display subsystem.

Figure 11-154 is a diagram of the DSS external connections.

Figure 11-154. DSS Environment



NOTE: The path from a module pin to SoC pad(s) is defined at SoC I/O logic level. The I/O logic maps the module signals to the different pads of the SoC and is programmable in the BOOT_CFG registers and dedicated module registers. For more information, see Section 5.1.3.1.1 *Pad Configuration Registers*.

CAUTION

The parallel interface and RFBI interface cannot be used simultaneously:

- When the DSS_DPI_CTRL[0] DPI_ENABLE is set to 0x1, the parallel output is enabled for usage.
- When the DSS_DPI_CTRL[0] DPI_ENABLE is set to 0x0, the parallel output is disabled. The output through RFBI module can be used, by enabling it via DSS_RFBI_CTRL[0] RFBI_ENABLE register bit.

11.3.2.1 DISPC Environment

The DISPC video port output (VP1) provides the required data and control signals to:

- Connect directly to device pads for MIPI DPI 2.0, or BT.656, or BT.1120 parallel interface support
- Connect to RFBI module for MIPI DBI 2.0 interface support (for more details, see Section 11.3.2.2, *RFBI Environment*)

In synchronous parallel interface configuration, the DISPC outputs data and control signals directly on device I/O pads: DATA[23:0] (data bus), HSYNC (horizontal synchronization signal), VSYNC (vertical synchronization signal), DE (data enable), FID (field identification, odd/even), and PCLK (pixel clock). The data and control signals can be provided directly to an external MIPI DPI 2.0 compatible LCD panel.

Table 11-396 describes the DISPC VP1 interface signals for direct connection to device pads for MIPI DPI 2.0, or BT.656 or BT.1120 parallel interface support.

Table 11-396. DISPC Video Port Signals for MIPI DPI 2.0, or BT.656 or BT.1120

Signal Name	Type[1]	Description
DSS_DATA[23:0]	O	Pixel data
DSS_PCLK	O	Pixel clock
DSS_VSYNC	O	Vertical synchronization. The frame synchronization pulse (vsync) toggles after all the lines in a frame are transmitted and a programmable number of line clock cycles has elapsed at the beginning and the end of each frame.
DSS_HSYNC	O	Horizontal synchronization. The line synchronization pulse (hsync) toggles after all pixels in a line are transmitted and a programmable number of pixel clock wait-states has elapsed at the beginning and the end of each line.
DSS_DE	O	Pixel data output-enable signal to indicate when data must be latched using the pixel clock.
DSS_FID	O	The FID signal indicates the output field identifier: • 0 means even • 1 means odd The FID signal is 0 when the progressive mode is selected for the VP output.

(1) I = Input, O = Output, I/O = Input/Output

11.3.2.1.1 DISPC VP1 Output and Data Formats

This section describes the pixel data bus and shows timing diagrams of transactions and synchronizations.

In the Active matrix display type, one pixel per pixel clock is displayed. The diagrams represent the configuration of assertion of the data on the rising edge of the pixel clock. It is possible to program the interface timing to output the data on the falling edge of the pixel clock.

Active matrix displays bypass the STN dithering logic block and the output FIFO. Each line represents one pixel.Figure 11-155 through Figure 11-158 show 12-, 16-, 18-, and 24-active matrix displays, respectively. The width of the data bus can be configured through DISPC_VP1_CONTROL[10-8] DATALINES register bitfield.

Figure 11-155. DISPC VP1 Pixel Data Color-12 Active Matrix

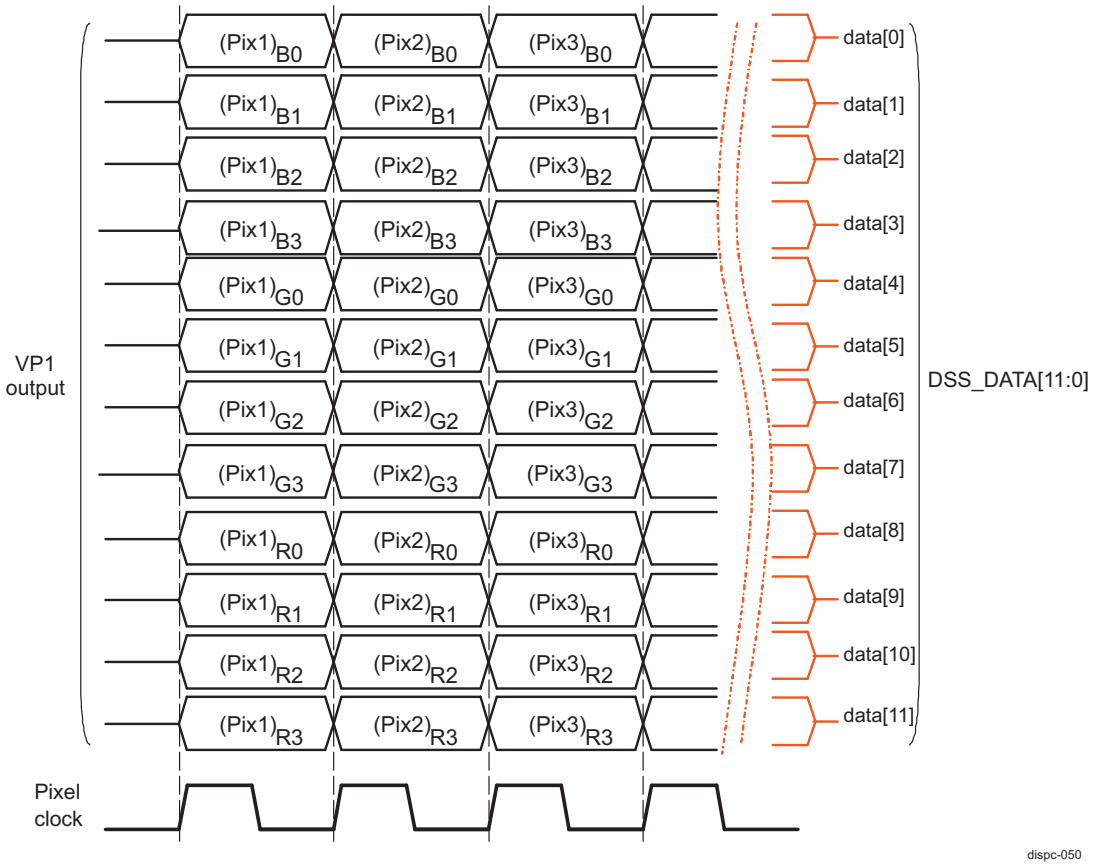


Figure 11-156. DISPC VP1 Pixel Data Color-16 Active Matrix

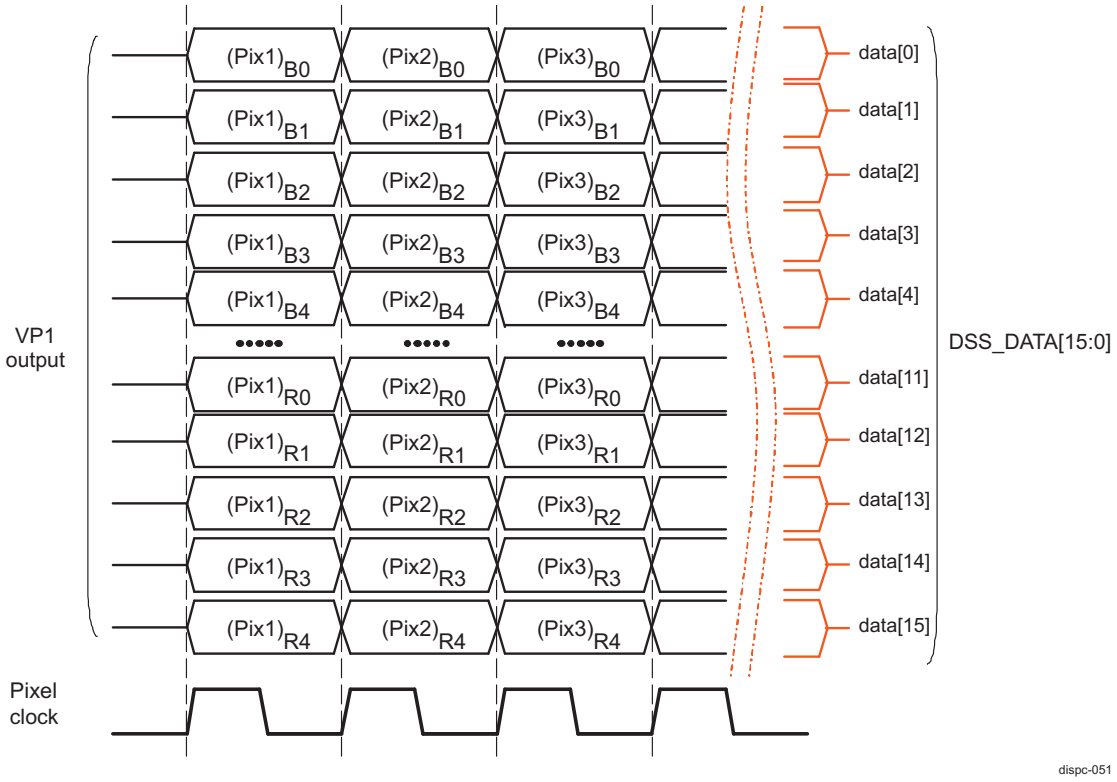


Figure 11-157. DISPC VP1 Pixel Data Color-18 Active Matrix

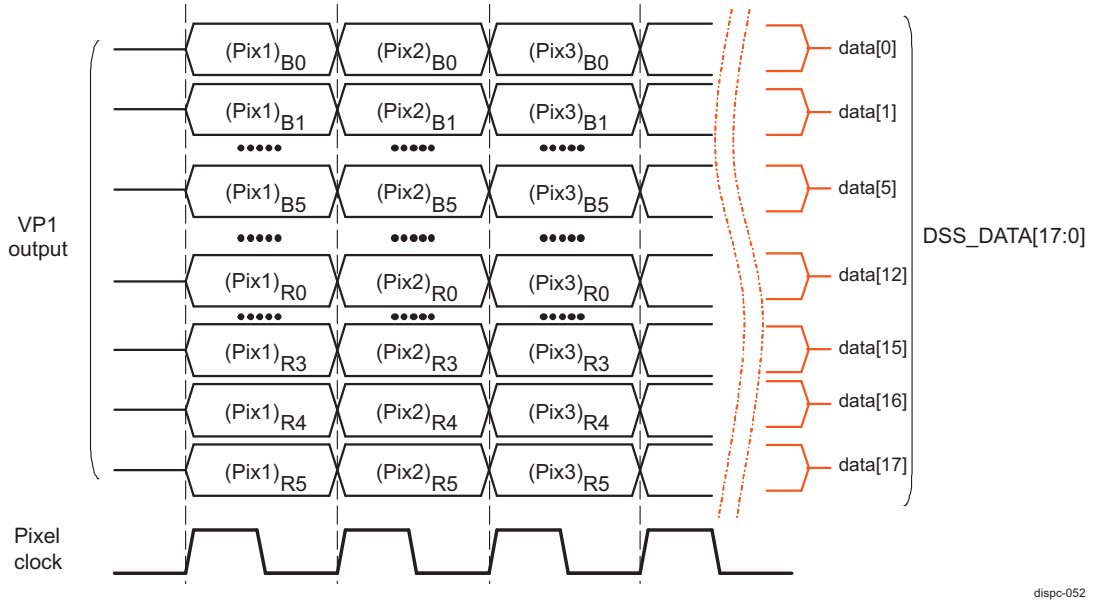
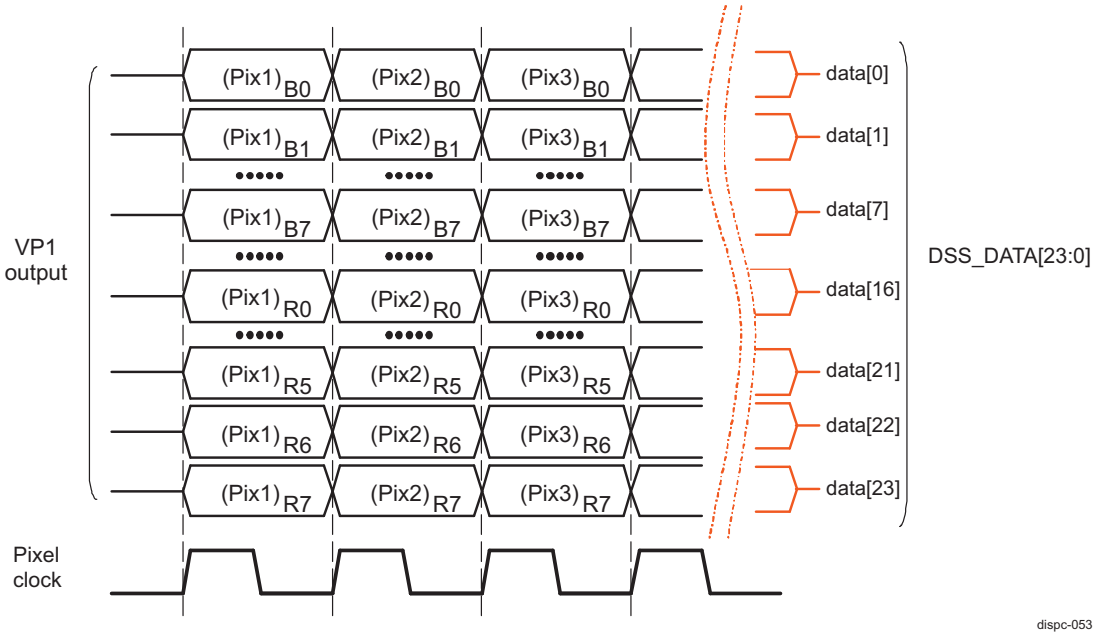


Figure 11-158. DISPC VP1 Pixel Data Color-24 Active Matrix



11.3.2.1.2 DISPC VP1 Active Marix Display Timing Diagrams

Figure 11-159 through Figure 11-162 show timing diagrams of synchronization signals and pixel clocks for active matrix panels. The DISPC directly drives these signals, which are related to the programmable fields listed in Table 11-397.

Table 11-397. DISPC VP1 Programmable Fields for Active Matrix Display

Name	Register	Description
PPL	DISPC_VP1_SIZE_SCREEN[11-0] PPL value + 1	Pixels per line
LPP	DISPC_VP1_SIZE_SCREEN[27-16] LPP value + 1	Lines per panel
HBP	DISPC_VP1_TIMING_H[31-20] HBP value + 1	Horizontal back porch
HFP	DISPC_VP1_TIMING_H[19-8] HFP value + 1	Horizontal front porch
HSW	DISPC_VP1_TIMING_H[7-0] HSW value + 1	Horizontal synchronization pulse width
VBP	DISPC_VP1_TIMING_V[31-20] VBP value	Vertical back porch
VFP	DISPC_VP1_TIMING_V[19-8] VFP value	Vertical front porch
VSW	DISPC_VP1_TIMING_V[7-0] VSW value + 1	Vertical synchronization pulse width
ALIGN	DISPC_VP1_POL_FREQ[18] ALIGN	Alignment between DISPC HSYNC and VSYNC assertion
ONOFF	DISPC_VP1_POL_FREQ[17] ONOFF	DISPC HSYNC and VSYNC pixel clock control
RF	DISPC_VP1_POL_FREQ[16] RF	DISPC HSYNC and VSYNC pixel clock edge control
IEO	DISPC_VP1_POL_FREQ[15] IEO	Invert output enable
IPC	DISPC_VP1_POL_FREQ[14] IPC	Invert DISPC PCLK
IHS	DISPC_VP1_POL_FREQ[13] IHS	Invert DISPC HSYNC
IVS	DISPC_VP1_POL_FREQ[12] IVS	Invert DISPC VSYNC

- Active matrix timing configuration 1:
  - DISPC_VP1_POL_FREQ[17] ONOFF = 0
  - DISPC_VP1_POL_FREQ[16] RF = 0

    The HSYNC and VSYNC signals are driven on the opposite edge of PCLK from the pixel data.

– DISPC_VP1_POL_FREQ[15] IEO = 0
  The DE signal is active high.
– DISPC_VP1_POL_FREQ[14] IPC = 0
  The pixel data are driven on the rising edge of PCLK.
– DISPC_VP1_POL_FREQ[13] IHS = 0
  The HSYNC signal is active high.
– DISPC_VP1_POL_FREQ[12] IVS = 0
  The VSYNC signal is active high.
– DISPC_VP1_POL_FREQ[18] ALIGN = 0
  The VSYNC and HSYNC assertion is not aligned.

Figure 11-159. DISPC Active Matrix Timing Diagram of Configuration 1 (Start of Frame)

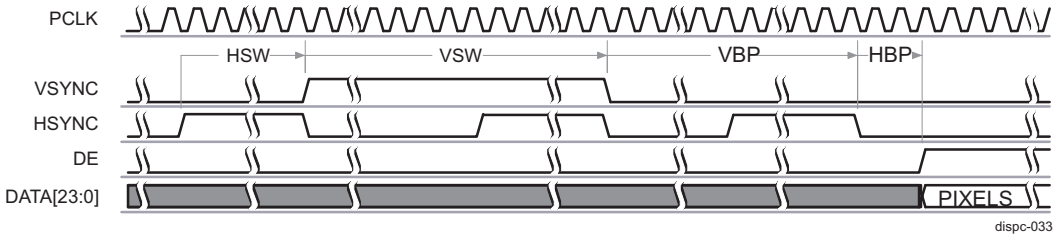


Figure 11-160. DISPC Active Matrix Timing Diagram of Configuration 1 (Between Lines)

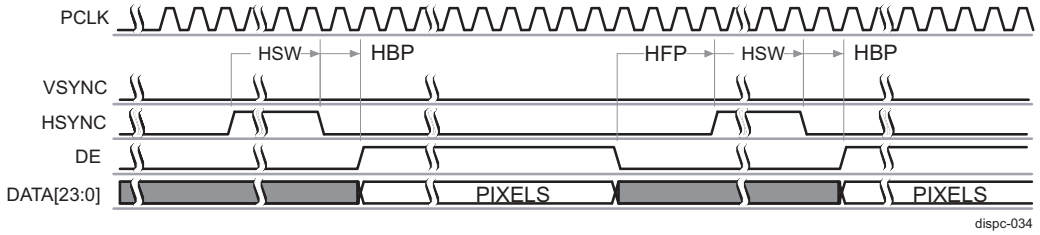


Figure 11-161. DISPC Active Matrix Timing Diagram of Configuration 1 (Between Frames)

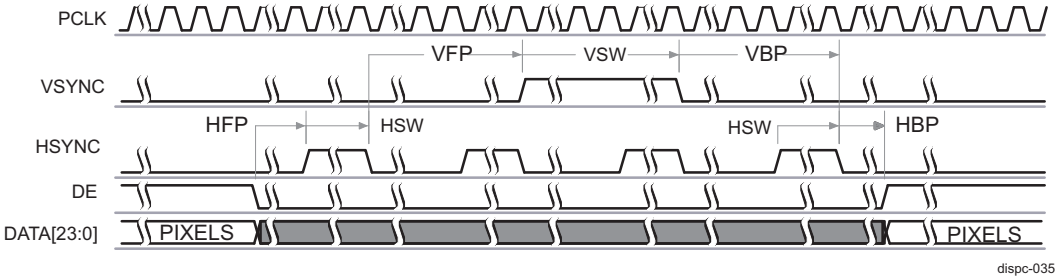
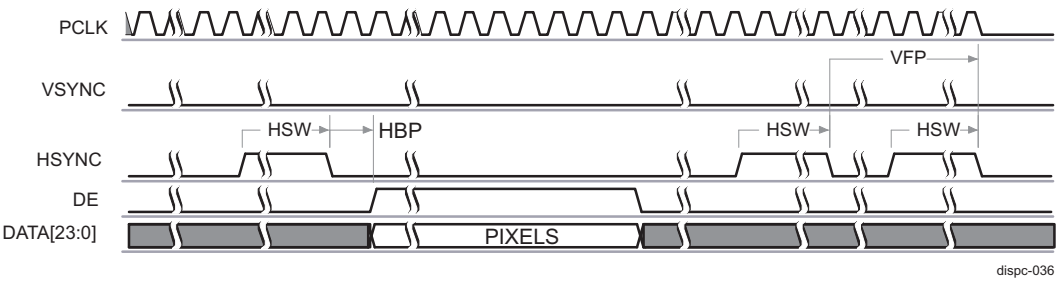


Figure 11-162. DISPC Active Matrix Timing Diagram of Configuration 1 (End of Frame)



- Active matrix timing configuration 2:

– DISPC_VP1_POL_FREQ[17] ONOFF = 1
– DISPC_VP1_POL_FREQ[16] RF = 1
  The HSYNC and VSYNC signals are driven on the rising edge of PCLK.
– DISPC_VP1_POL_FREQ[15] IEO = 1
  The DE signal is active low.
– DISPC_VP1_POL_FREQ[14] IPC = 1
  The pixel data is driven on the falling edge of PCLK.
– DISPC_VP1_POL_FREQ[13] IHS = 1
  The HSYNC signal is active low.
– DISPC_VP1_POL_FREQ[12] IVS = 1
  The VSYNC signal is active low.
– DISPC_VP1_POL_FREQ[18] ALIGN = 0
  The VSYNC and HSYNC assertion is not aligned.

Figure 11-163. DISPC Active Matrix Timing Diagram of Configuration 2 (Start of Frame)

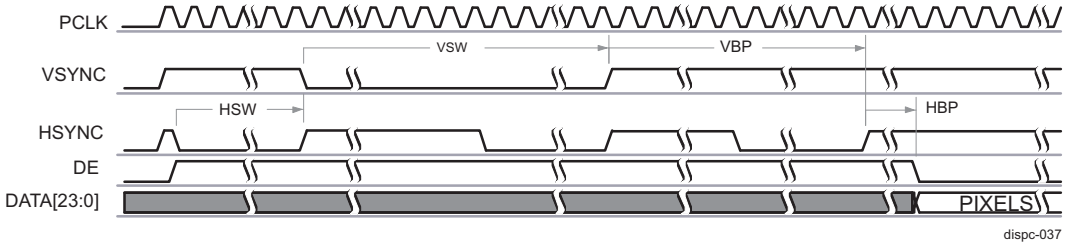


Figure 11-164. DISPC Active Matrix Timing Diagram of Configuration 2 (Between Lines)

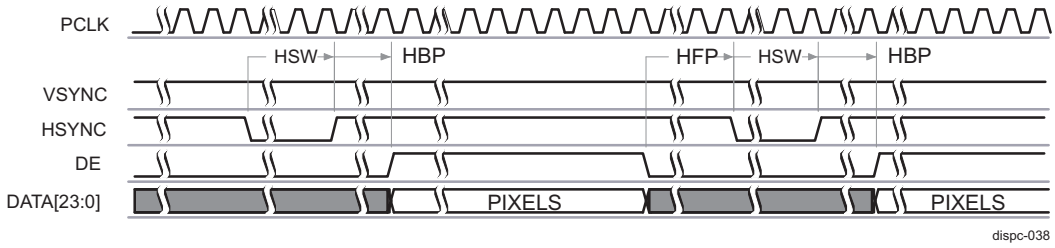


Figure 11-165. DISPC Active Matrix Timing Diagram of Configuration 2 (Between Frames)

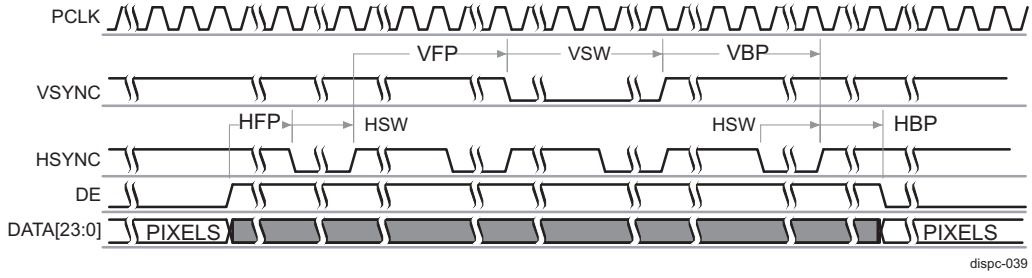
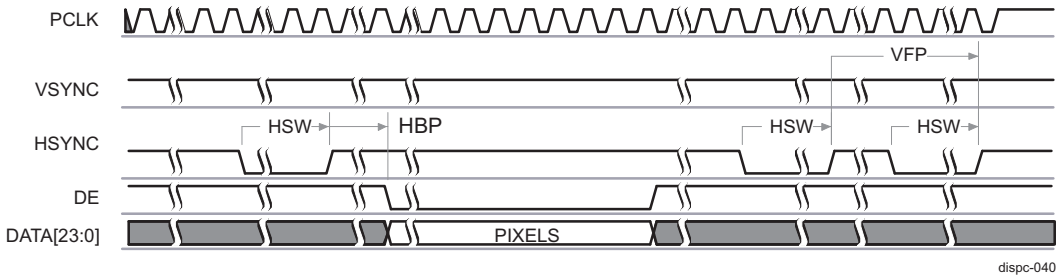


Figure 11-166. DISPC Active Matrix Timing Diagram of Configuration 2 (End of Frame)



- Active matrix timing configuration 3:
  - DISPC_VP1_POL_FREQ[17] ONOFF = 1
  - DISPC_VP1_POL_FREQ[16] RF = 1
    The HSYNC and VSYNC signals are driven on the rising edge of PCLK.
  - DISPC_VP1_POL_FREQ[15] IEO = 0
    The DE signal is active high.
  - DISPC_VP1_POL_FREQ[14] IPC = 0
    The pixel data are driven on the rising edge of PCLK.
  - DISPC_VP1_POL_FREQ[13] IHS = 0
    The HSYNC signal is active high.
  - DISPC_VP1_POL_FREQ[12] IVS = 0
    The VSYNC signal is active high.
  - DISPC_VP1_POL_FREQ[18] ALIGN = 0
    The VSYNC and HSYNC assertion is not aligned.

Figure 11-167. DISPC Active Matrix Timing Diagram of Configuration 3 (Start of Frame)

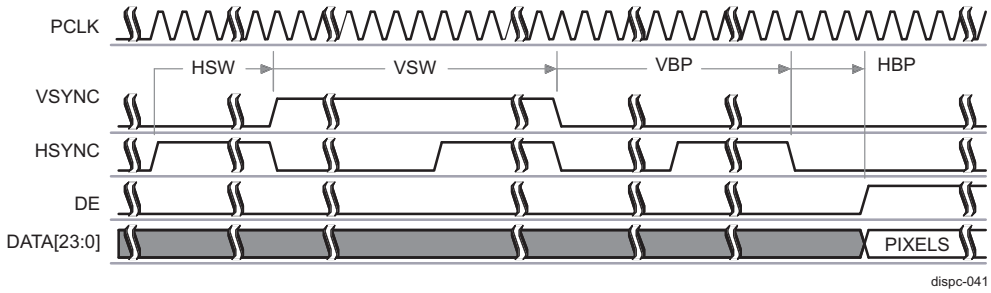


Figure 11-168. DISPC Active Matrix Timing Diagram of Configuration 3 (Between Lines)

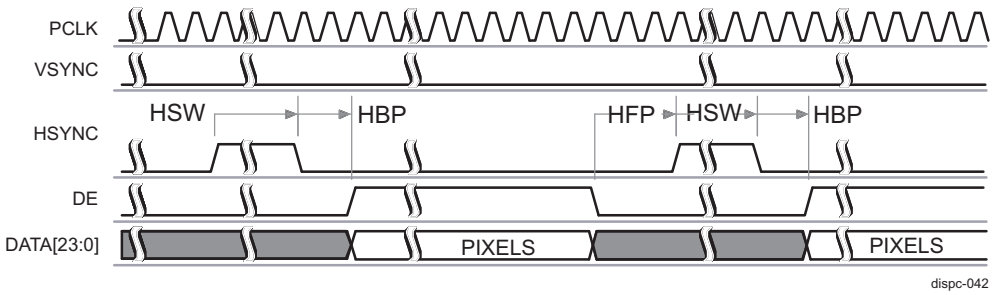


Figure 11-169. DISPC Active Matrix Timing Diagram of Configuration 3 (Between Frames)

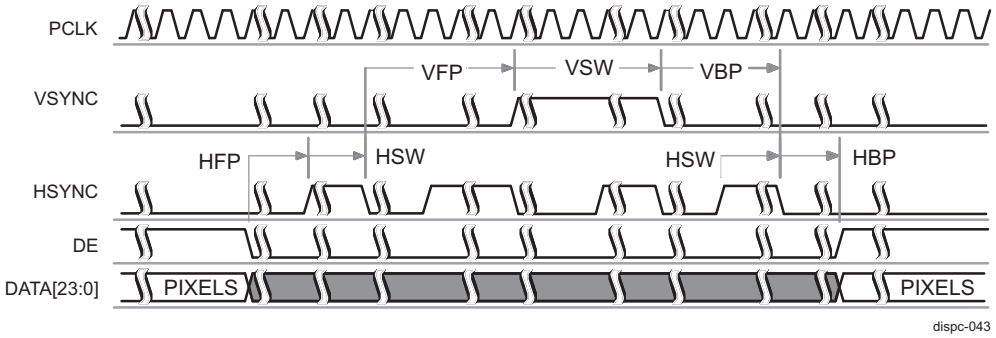
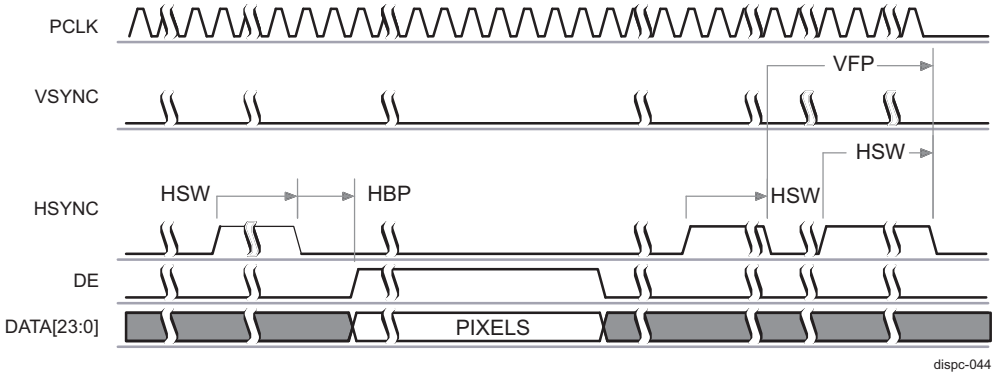


Figure 11-170. DISPC Active Matrix Timing Diagram of Configuration 3 (End of Frame)



11.3.2.2 RFBI Environment

In asynchronous RFBI configuration, the remote frame buffer (RFB) of the peripheral LCD panel is connected directly to the RFBI module of the device. RFBI controls the reads and writes to and from the RFB. The RFBI receives data from the DISPC video port, or from system memory (through the interconnect slave port), and generates the signals to control the LCD panel.

The RFBI module provides the necessary control signals to interface between the DISPC on one side and the LCD driver of the peripheral LCD panel on the other. The SoC ARMSS or DMA controller can read and write the encoded pixel values from/to the remote frame buffer inside the peripheral. In the forward direction from the SoC to the peripheral, the stream consists of commands, parameters, and pixels.

For the forward direction from the SoC to the LCD peripheral, the commands and parameters are provided using the RFBI interconnect slave port. The pixel data is provided either by:

- DISPC video port: DISPC uses its own DMA engine to read the system frame buffer, or
- SoC ARMSS or DMA controller, using the RFBI interconnect slave port

For the reverse direction from LCD peripheral to SoC, the data from the peripheral are transferred to system memory by the SoC ARMSS or DMA controller through the RFBI interconnect slave port. The ARMSS can also analyze directly the information received from the peripheral. In that case, there is not always access to the system memory when data are received from the peripheral (that is, status of the peripheral state: ON/OFF/IDLE).

The RFBI can manage only one LCD panel at a time, although a sequential configuration is also possible.

Table 11-398 describes the RFBI interface signals to/from a peripheral LCD panel.

Table 11-398. RFBI Interface Signals to/from Peripheral LCD

Signal Name	Type[1]	Description
DSS_RFBI_DATA[15:0]	I/O	RFBI I/O data

(1) I = Input; O = Output; I/O = Input/Output

Table 11-398. RFBI Interface Signals to/from Peripheral LCD (continued)

Signal Name	Type[1]	Description
DSS_RFBI_REn	O	Read access signal
DSS_RFBI_WEn	O	Write access signal
DSS_RFBI_A0	O	Command/data selection signal
DSS_RFBI_CS0	O	Chip-select (CS) signal for LCD panel 0
DSS_RFBI_CS1	O	Chip-select (CS) signal for LCD panel 1
DSS_RFBI_TEVSYNC0	I	Tearing effect (TE) synchronization signal (TE or VSYNC for LCD panel 0)
DSS_RFBI_TEVSYNC1	I	Tearing effect (TE) synchronization signal (TE or VSYNC for LCD panel 1)
DSS_RFBI_HSYNC0	I	HSYNC from LCD panel 0
DSS_RFBI_HSYNC1	I	HSYNC from LCD panel 1

- DSS_RFBI_DATA[15:0]: The pixel data comprises the RFBI pixel data (bits 15-0). A write/read command must be sent to the LCD panel to send/read the data.

  Before any data access, the application must send commands and parameters when it is necessary to configure an LCD panel. The data is used as input in read operations during production test and also to read the status of the registers in the LCD panel and pixels from the embedded frame buffer in the LCD panel.
- DSS_RFBI_REn: The read-enable signal indicates an ongoing read from the embedded memory in the LCD panel. The RFBI registers describe the behavior of the read signal (off/on/cycle time). The polarity of the read-enable signal is programmable. The read is used to obtain status/data information from the LCD panel.
- DSS_RFBI_WEn: The write-enable signal indicates an ongoing write. The RFBI registers allow flexible behavior of the write signal (programmable off/on/cycle times and signal polarity).
- DSS_RFBI_A0: The signal is asserted to indicate its status: Command or Data. The polarity is programmable and the status of the signal depends on the RFBI registers written by the application (RFBI_CMD/RFBI_READ/RFBI_STATUS/RFBI_PARAM/RFBI_DATA). The register in use by the hardware defines the status of DSS_RFBI_A0. The order of the writes/reads to RFBI registers RFBI_CMD/RFBI_READ/RFBI_STATUS/RFBI_PARAM/RFBI_DATA defines the transitions of RFBI_A0 signal.
- DSS_RFBI_CS0/CS1: The chip-select (CS) signals are asserted to indicate which LCD panel is selected and must be ready to receive/transmit commands and data. Two LCD panels can be connected at the same time to the RFBI module. When RFBI_REn or RFBI_WEn is on, the RFBI_CS0/CS1 signals must not be changed.

  To select the trigger mode, configure the RFBI_CONFIG__0/RFBI_CONFIG__1[3-2] TRIGGERMODE bit field (0x0: Internal trigger mode with the RFBI_CONTROL[4] ITE bit; 0x1: External trigger mode with the TE signal TEVSYNC; 0x2: External trigger mode with the TEVSYNC and HSYNC signals with the programmable line counter).
- DSS_RFBI_TEVSYNC0/TEVSYNC1 (used by RFBI only if RFBI_CONFIG__0/RFBI_CONFIG__1[3-2] TRIGGERMODE = 0x1 or 0x2; otherwise, this signal is ignored): Based on the trigger mode selected, the signal is the TE pulse signal or the LCD panel vertical synchronization (VSYNC) pulse signal. TE logic uses VSYNC as the synchronization signal to send the pixel to the LCD panel.
- DSS_RFBI_HSYNC0/HSYNC1 (used by RFBI only if RFBI_CONFIG__0/RFBI_CONFIG__1[3-2] TRIGGERMODE = 0x2, otherwise, this signal is ignored): The HSYNC pulse signals indicate to the RFBI when horizontal synchronization occurs. The polarity of the HSYNC signals is programmable. The minimum pulse width of the signal is two DSS_OCP_CLK cycles. TE logic uses HSYNC as a synchronization signal to send the pixel to the LCD panel.

11.3.2.2.1 RFBI Description of the Tearing Effect Pulse Signal

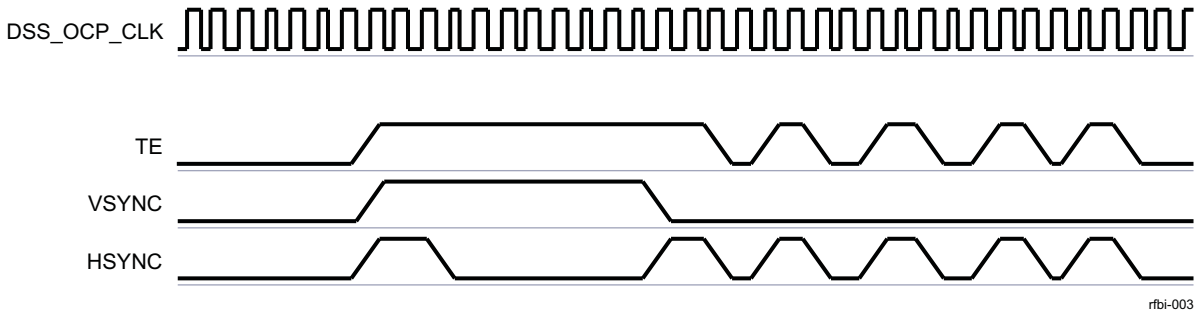
The TE signal can be generated externally by the RFBI depending on the value of the RFBI_CONFIG__0/RFBI_CONFIG__1[3-2] TRIGGERMODE bit field:

- TRIGGERMODE = 0x1: External trigger mode with the TE signal, which corresponds to the DSS_RFBI_TEVSYNC0 signal.

- TRIGGERMODE = 0x2: External trigger mode with the HSYNC/VSYNC signals. The HSYNC signal corresponds to the DSS_RFBI_HSYNC0 signal. The VSYNC signal corresponds to the DSS_RFBI_TEVSYNC0 signal.

The externally generated TE synchronization signal is a logical OR or AND operation between the HSYNC and VSYNC signals (see Figure 11-171). The logical operation (OR or AND) depends on the polarity of the HSYNC and VSYNC signals. The VSYNC signal indicates to the RFBI when vertical synchronization occurs. The HSYNC signal indicates to the RFBI when horizontal synchronization occurs.

Figure 11-171. RFBI External Generation of TE Signal Based on Logical OR Operation Between HSYNC and VSYNC (Active High)



- The TE signal is connected to DSS_RFBI_TEVSYNC0 when the RFBI_CONFIG__0/RFBI_CONFIG__1[3-2] TRIGGERMODE bit field is set to 0x1.
- The HSYNC and VSYNC signals are connected to DSS_RFBI_HSYNC0 and DSS_RFBI_VSYNC0, respectively, when the RFBI_CONFIG__0/RFBI_CONFIG__1[3-2] TRIGGERMODE bit field is set to 0x2.

The RFBI module detects the VSYNC and HSYNC pulses embedded in the received signal. VSYNC is detected based on the minimum pulse width defined by the RFBI_VSYNC_WIDTH register. VSYNC is not triggered by an inactive-to-active edge.

HSYNC is detected based on the minimum pulse width defined by the RFBI_HSYNC_WIDTH register. HSYNC is not triggered by an inactive-to-active edge.

The signal is generated from external logic based on VSYNC and HSYNC of the LCD panel. The automatic trigger can be programmed based on the external RFBI_TE signal or use the RFBI_CONTROL[4] ITE bit to start data capture (internal trigger mode).

The polarity of the TE signal is programmable in the RFBI_CONFIG__0/RFBI_CONFIG__1 register. HSYNC and VSYNC pulses embedded in the TE signal have the same polarity, which is active high for an OR-ed signal and active low for an AND-ed signal. The minimum pulse width of the signal is two DSS_OCP_CLK cycles. Hardware resets the line counter when VSYNC occurs and increments it at every HSYNC. Transfer to the LCD panel begins when the line counter reaches the programmable line number.

11.3.2.2.2 RFBI Transaction Timing Diagrams

Table 11-399 lists the programmable timing fields. Figure 11-172 through Figure 11-174 show timing diagrams of read/write transactions to the LCD panel for the RFBI. In these figures, the polarity 0 (active low) describes the DSS_RFBI_A0, DSS_RFBI_CS0/CS1, DSS_RFBI_REn, and DSS_RFBI_WEn signals.

Table 11-399. RFBI Programmable Timing Fields in RFBI Mode

Timing Name	Register Field	Description
CSOnTime	RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[3-0] CSONTIME bit field	RFBI_CS0/CS1 assertion time from start access time
CSOffTime	RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[9-4] CSOFFTIME bit field	RFBI_CS0/CS1 deassertion time from start access time
WECycleTime	RFBI_CYCLE_TIME__0/RFBI_CYCLE_TIME__1[5-0] WECYCLETIME bit field	The time when RFBI_A0 becomes valid until write cycle completion
WEOnTime	RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[13-10] WEONTIME bit field	RFBI_WEn assertion delay time from start access time

Table 11-399. RFBI Programmable Timing Fields in RFBI Mode (continued)

Timing Name	Register Field	Description
WEOffTime	RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[19-14] WEOFFTIME bit field	RFBI_WEn deassertion delay time from start access time
RECycleTime	RFBI_CYCLE_TIME__0/RFBI_CYCLE_TIME__1[11-6] RECYCLETIME bit field	The time when RFBI_A0 becomes valid until read cycle completion
REOnTime	RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[23-20] REONTIME bit field	RFBI_REn assertion delay time from start access time
REOffTime	RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[29-24] REOFFTIME bit field	RFBI_REn deassertion delay time from start access time
CSPulseWidth	RFBI_CYCLE_TIME__0/RFBI_CYCLE_TIME__1[17-12] CSPULSEWIDTH bit field	The time when write cycle time or read cycle time completes

Figure 11-172. RFBI Command Data Write

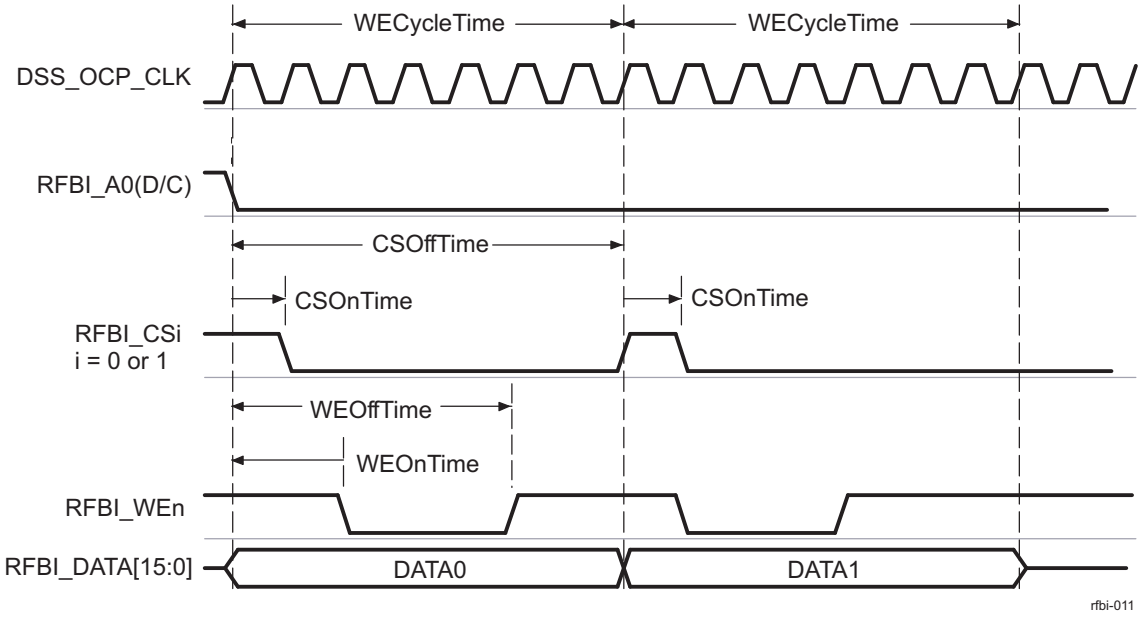


Figure 11-173. RFBI Display Data Read

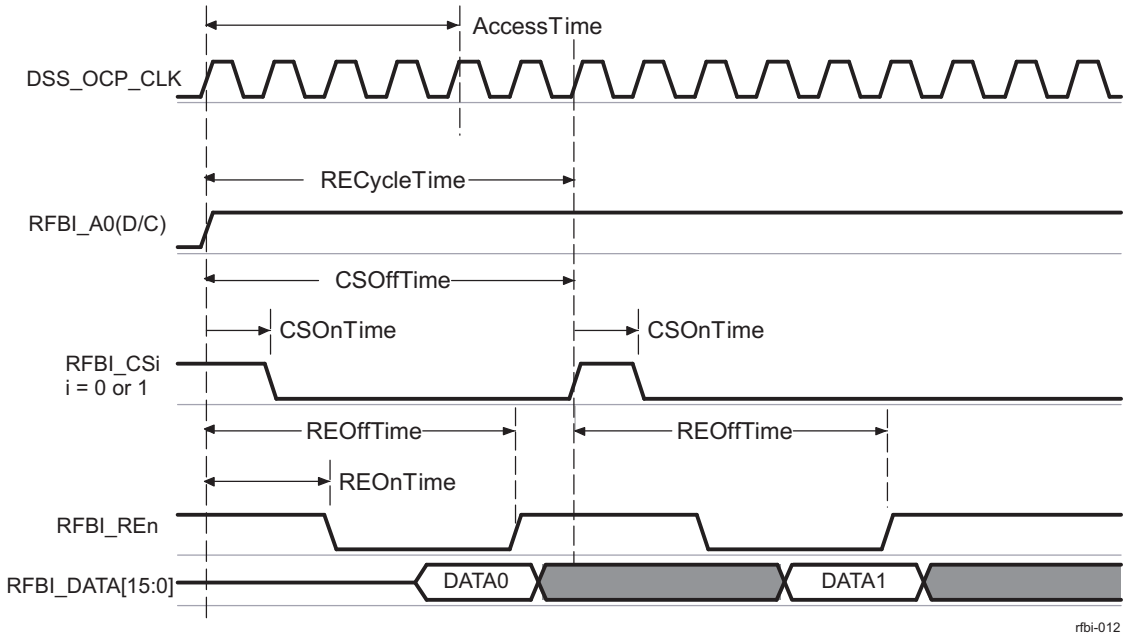
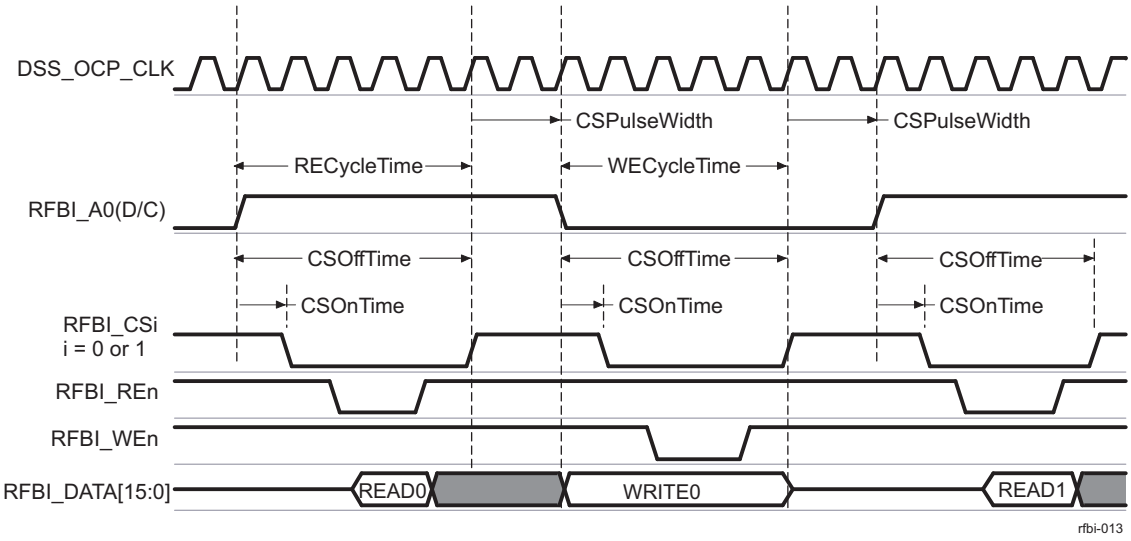


Figure 11-174. RFBI Read to Write and Write to Read



11.3.3 DSS Integration

This section describes the integration of DSS in the device, including information about clocks, resets, and hardware requests.

Figure 11-175 shows the integration of DSS in the device.

Figure 11-175. DSS Integration

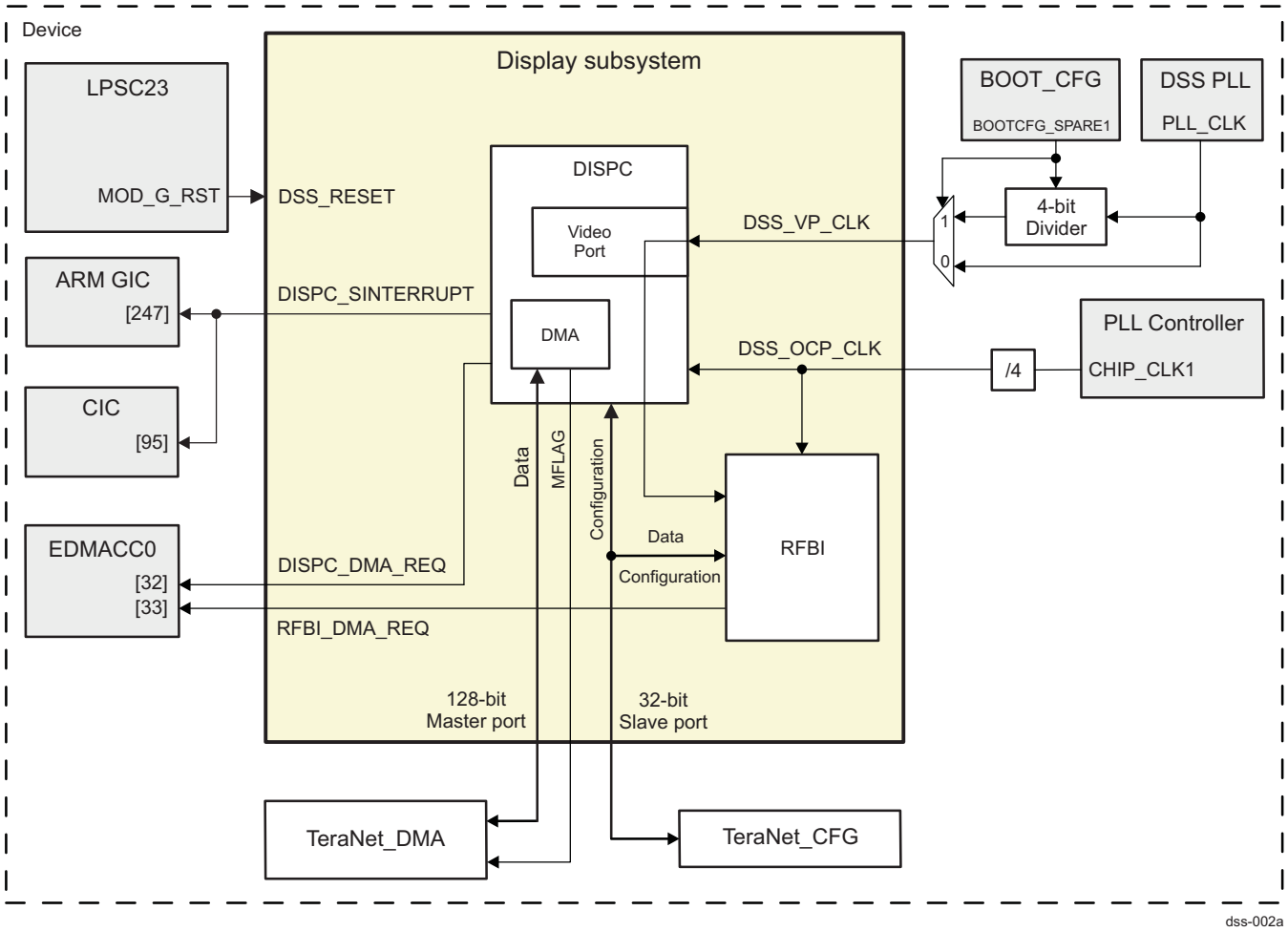


Table 11-400 summarizes the main integration attributes of DSS in the device.

Table 11-400. DSS Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
DSS	PD12	LPSC23	TeraNet_DMA: Connected to DSS 128-bit master port. Used for by DISPC DMA engine to read pixel data from system memory. TeraNet_CFG: Connected to DSS 32-bit slave port. Used for DSS, DISPC, and RFBI configuration, and data traffic via RFBI slave port.

NOTE: For more information on device and module level power management, see the corresponding sections within Chapter 5, *Device Configuration*.

For more information on device interconnects, see Section 3.1, *Interconnect*.

11.3.3.1 DSS Clocks and Resets

Table 11-401. DSS Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
DSS (DISPC and RFBI)	DSS_OCP_CLK	CHIP_CLK1 / 4	PLL Controller	Interconnect interface clock (for both DSS master port and slave port). Also the main functional clock (FCLK) inside the DISPC. Must be greater than DSS_VP_CLK for all supported display resolutions and refresh rates. RFBI module uses this clock when capturing data on the RFBI slave port data bus (through RFBI interconnect FIFO).
	DSS_VP_CLK	PLL_CLK / (4-bit clock divider value)	DSS PLL	DISPC video port (VP1) pixel clock. Its frequency is tied to the external LCD panel resolution and refresh rate. RFBI module uses this clock in order to capture the pixels from DISPC video port output, and for the generation of the STALL signal. The 4-bit clock divider value is controlled via BOOTCFG_SPARE1 register inside the device BOOT_CFG module.
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
DSS	DSS_RESET	MOD_G_RST	LPSC23	DSS asynchronous reset

The DSS is reset via a global HW reset signal (DSS_RESET), which is propagated to all DSS sub-modules.

Each DSS sub-module can also be reset by a corresponding SW reset on sub-module level.

The [0] DSS_RESETDONE and [7] RFBI_RESETDONE bits of DSS_SYSSTATUS register indicate the completion of the DSS reset following the HW reset, or after a SW reset of all its sub-modules.

NOTE: For more information on device reset and clock management, see the corresponding sections within Chapter 5, *Device Configuration*.

11.3.3.2 DSS Interrupt and DMA Requests

Table 11-402 shows the interrupt and DMA requests generated by the DISPC and RFBI modules.

Table 11-402. DSS Hardware Requests

Interrupt Requests				
Module Instance	**Event Name**	**Mapped To Input Event [Number]**		**Description**
		ARM GIC	**CIC**	
DISPC	DISPC_SINTERRUPT	[247]	[95]	DISPC interrupt
DMA Requests				
Module Instance	**Event Name**	**Mapped To Input Event [Number]**		**Description**
		EDMACC0		

Table 11-402. DSS Hardware Requests (continued)

DISPC	DISPC_DMA_REQ	[22]	Connected to DISPC line trigger DMA event. The line trigger signal is used for synchronization of a logical channel in the SoC DMA Controller, when performing data transfer. There is no real data received by the DISPC based on this DMA request. It can also be used to update a frame buffer base-address stored in the system memory.
RFBI	RFBI_DMA_REQ	[23]	RFBI FIFO DMA event. This signal is useful, if a SoC initiator (DMA controller) wants to do a direct pixel data transfer through RFBI, bypassing the DISPC.

NOTE: For more information on device interrupt controllers, see Chapter 9, *Interrupts*.

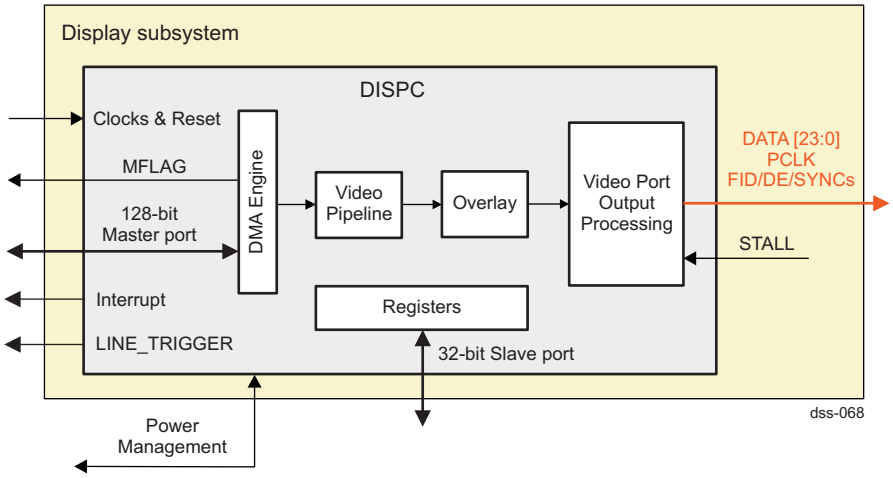
For more information on device DMA controllers, see Chapter 10, *Enhanced Direct Memory Access (EDMA) Controller*.

11.3.4 DSS Functional Description

11.3.4.1 DISPC Functional Description

11.3.4.1.1 DISPC Overview

Figure 11-176. DISPC Architecture Overview



The DISPC integrated within DSS is a fully scalable module capable of fetching pixel data from the device system memory through its single master port, performing various pixel processing, and then providing the processed pixels to an external display panel.

Several processes are configurable by the user in order to manage the video pipeline (color space conversion, up-sampling, down-sampling) and overlay features. The internal timing generator logic generates the vieo port output signals based on VESA DMT and CEA-861 standard.

The internal DMA engine tightly coupled to the DISPC is used for the pixel data transfer from system memory (frame buffer). The DISPC DMA engine is in charge of scheduling the requests.

The DISPC video port output can be connected to display peripherals either directly (for MIPI DPI 2.0, or BT.656, or BT.1120 support), or through RFBI module for MIPI DBI 2.0 support.

NOTE: DISPC does not support any tiled frame buffer, nor any compressed frame buffer. The DISPC has no capability to support by itself rotation of the frame buffer.

11.3.4.1.2 DISPC Features

The DISPC provides the following features:

- One Video Pipeline (VID1):
  - Input pixel formats supported: ARGB16-4444, ABGR16-4444, RGBA16-4444, RGB16-565, BGR16-565, ARGB16-1555, ABGR16-1555, ARGB32-8888, ABGR32-8888, BGRA32-8888, RGBA32-8888, RGB24-888, ARGB32-2101010, ABGR32-2101010, ARGB48-12121212, RGBA48-12121212, ARGB64-16161616, RGBA64-16161616, RGB565-A8, BITMAP1, BITMAP2, BITMAP4, BITMAP8, YUV422-UYVY, YUV422-YUV2, YUV420-NV12, YUV420-NV21 and in addition RGBx, xRGB, xBGR, and BGRx pixel formats defined considering that A component of RGBA, ARGB, ABGR, BGRA pixel formats is ignored by HW (e.g. ARGB->xRGB)
  - Premultiplied ARGB and RGBA formats
  - Programmable poly-phase filter:
    - Independent horizontal and vertical resampling. Upsampling up to x16, and downsampling down to 1/4
    - Maximum input width of 1280 pixels (32-bit) in 5-tap mode, and 2560 pixels (32-bit) in 3-tap

mode
- No limitation on the input height
- Supported pixel formats for scaling are all above listed formats, except BITMAP formats. The alpha channel is rescaled like the R, G and B color components. 16 phases with symmetrical coefficients are implemented.

– Programmable color space conversion from YUV4:2:2/YUV4:4:4, and YUV4:2:0 (after chroma upsampling through the scaler) into ARGB48-12121212.
– Programmable VC-1 range mapping
– Support for color look-up table (CLUT): 256 × 24-bit entries palette in RGB
– Up to 4K-by-4K frame buffer size

- One Video Port (VP1) output:
  – Up to 24 bits per pixel on the output interface, with selection between 12, 16, 18, or 24 bits
  – Programmable timing generator to support up to 150MHz pixel clock (PCLK) frequency for video formats defined in CEA-861-E and VESA-DMT standards
  – Programmable multiple cycles output format on 8/9/12/16-bit interface (TDM)
  – Configurable output mode: progressive or interlaced mode
  – Selection between RGB and YUV422 output pixel format (YUV4:2:2 only available when a BT mode is enabled)
- One Overlay Manager (OVR1):
  – Input pixel format: RGB48-12121212
  – Output pixel format: RGB48-12121212
  – Programmable background color
- Common:
  – Programmable 8-bit gamma curve support on VP1 output
  – Programmable color phase rotation (CPR)
- DMA (internal to the DISPC):
  – No support for rotation, flipping, mirroring, and memory fragmentation
  – Integrated buffers between DMA engine and VID1 pipeline
  – Programmable buffer thresholds
  – Support for MFLAG to indicate critical fullness of DMA buffer (close to underflow for video pipeline)
  – Region-Based support (same layer can display multiple windows)
- Advanced:
  – Self-refresh using the DMA buffers (outputting data on display only from the DMA buffer)

NOTE: The display resolution is programmable and can be any width in the range [1:4096] pixels. The following limitations, related to the type of display or the processing done, apply:
- Active Matrix screen + dithering forces a width multiple of 2 pixels
- Active Matrix + TDM may force a width multiple of 2 pixels

The display buffers in the system memory should consist of contiguous pixels.

11.3.4.1.3 DISPC Clock Configuration

The DISPC pixel clock (PCLK) for the Video Port (VP1) output is provided directly from the DSS_VP_CLK input clock. There is no internal DSS divisor on this clock. The PCLK transitions continuously as long as the VP1 is enabled and DISPC_VP1_CONFIG[0] PIXELGATED register bit is 0x0.

The DISPC internal functional clock (FCK_CLK) is provided by DSS_OCP_CLK. There is no internal divisor on this clock. It also acts as the interface clock for the DISPC master and slave ports.

The FCK_CLK has to be greater than the PCLK (that is, DSS_OCP_CLK must be greater than DSS_VP_CLK), in order for DISPC internal logic to function properly.

The minimum ratio between the PCLK frequency and FCK_CLK does not depend on the vertical scaling ratio. It is possible to rescale vertically between 16x and 0.25x with a clock ratio between functional clock and pixel clock of 1x. For the horizontal scaling, the minimum clock ratio does not depend on the video window width but on the display width. For example, downscaling a 1920 pixels wide video frame by 0.25 horizontally to 480 pixels and then displayed on a 1920 pixels wide screen, can be achieved by a FCK_CLK/PCLK ratio as low as 1.

The PCLK and FCK_CLK clocks are asynchronous. There is re-synchronization logic between internal logic and the VP1 output interface.

11.3.4.1.4 DISPC Software Reset

To perform a software reset on the DISPC, set the DISPC_SYSCONFIG[1] SOFTRESET bit to 0x1. The DISPC_SYSSTATUS[0] DISPC_FUNC_RESETDONE bit indicates that the software reset is complete when its value is 0x1. When the software reset completes, the DISPC_SYSCONFIG[1] SOFTRESET bit is automatically reset. Software must ensure that the software reset completes before performing DISPC operations.

The completion of the software reset for the VP logic is indicated in the DISPC_SYSSTATUS[1] DISPC_VP1_RESETDONE bit.

11.3.4.1.5 DISPC Interrupt Requests

The interrupt line, DISPC_SINTERRUPT, indicates when one or more events are detected by the hardware. Each event is independently maskable by configuring the DISPC_IRQENABLE_SET register.

To check when a particular interrupt event occurs and to reset a particular event, the DISPC_IRQSTATUS register must be accessed. This register regroups the status of internal events in the module that generate an interrupt (Read 0: no interrupt occurred; Read 1: interrupt occurred; Write 1: status bit reset).

There are two level of interrupts. The first level, controlled by the DISPC_IRQENABLE_SET, DISPC_IRQENABLE_CLR, and DISPC_IRQSTATUS registers, is used to indicate common events and is also source for the second level of interrupts. The second level of interrupts consists of status and enable interrupt registers for the VID1 pipeline and the video port.

Table 11-403. DISPC Interrupts - First Level

Interrupt Name	Description
VP1_IRQ	At least one event of the Video Port 1 interrupt events occurred
VID1_IRQ	At least one event of the VID1 pipeline interrupt events occurred

The second level of interrupts for VID1 pipeline is controlled through the DISPC_VID1_IRQENABLE and DISPC_VID1_IRQSTATUS registers.

Table 11-404. DISPC Interrupts - Second Level - VID1 Pipeline

Interrupt Name	Description
VIDENDWINDOW_IRQ	End of the VID1 window: The DMA engine has fetched all the data from memory for the VID1 for the current frame.
VIDBUFFERUNDERFLOW_IRQ	VID1 DMA buffer underflow: The input VID1 DMA buffer goes underflow. Does not necessary means that the buffer is empty (out of order refill), but simply that the required pixel is not in yet.
VIDREGIONBASEDPIPESTART_IRQ	Video window started on overlay. Used for Region-based feature.
VIDREGIONBASEDPIPEEND_IRQ	Video window ended on overlay. Used for Region-based feature.

The second level of interrupts for VP1 ouput is controlled through the DISPC_VP1_IRQENABLE and DISPC_VP1_IRQSTATUS registers.

Table 11-405. DISPC Interrupts - Second Level - VP1 Output

Interrupt Name	Description
FRAMEDONE_IRQ	Frame Done for VP1 output. After disabling the VP1 output of the DISPC, the interrupt is set when the active frame related to the VP1 has completed.
VSYNC_IRQ	VSYNC for VP1 output: VSYNC interrupt for the VP1 has occurred at the end of the frame.
VSYNC_ODD_IRQ	VSYNC for odd field. VSYNC_ODD interrupt has occurred at the end of the frame (EVSYNC received and the field polarity is odd).
PROGRAMMEDLINENUMBER_IRQ	Programmed line number: The VP1 has reached the user programmed line number.
SYNCLOST_IRQ	Synchronization lost on VP1 output: Occurs when VSYNC width/front or back porches are not wide enough to load the pipeline with data (VP1 output).

11.3.4.1.6 DISPC DMA Requests

The DISPC DMA request signal, DISPC_DMA_REQ, is not a classical one, but rather a synchronization signal between the DISPC and device DMA controller. The device DMA controller is informed that a programmable number of lines are output on the VP1, and that the system memory can be updated. This request is related to the interrupt event PROGRAMMEDLINENUMBER_IRQ described in Table 11-405. This allows the device DMA controller channel to be synchronized with the internal DMA controller of the DISPC.

In other words, it allows synchronizing a system memory frame buffer update, based on the line of the frame buffer in system memory, which is scanned by the DISPC. The DISPC_DMA_REQ request is generated at a programmable line number defined in the DISPC_VP1_LINE_NUMBER[11-0] LINENUMBER bit field. This process allows an application to use a single frame buffer and to update it after a certain number of lines are read by the DISPC.

11.3.4.1.7 DISPC DMA Engine

The DISPC DMA engine:

- Requests and supplies data from system memory to the VID1 pipeline through the interconnect, based on the configuration of the DISPC and VID1 pipeline settings.
- Is fully programmable and fetches pixel data using 1D burst.

The VID1 pipeline has a dedicated buffer and a channel with independent settings. Table 11-406 lists the default size and allocation of the DMA buffer. The allocation is done in the DISPC_GLOBAL_BUFFER register.

Table 11-406. DISPC DMA Buffer Size

Pipeline	DMA Buffer Size
VID1	4 lines × 640 × 128 bits

11.3.4.1.7.1 DISPC DMA Addressing and Bursts

For each line to be fetched, the DMA engine:

- Calculates the pixel address
- Aligns the address
- Defines the burst structure:
  - Type of burst (1D only)
  - Length of the burst

The DMA engine generates scan addresses to read data from the system memory. The base address defines the start address of the first pixel, and then the address is incremented based on the number of pixels per line, offset between two consecutive lines and number of lines. The ROW_INC register of VID1 pipeline allow to access a frame using 1D bursts (but as a two-dimensional block) by adding a fixed address offset at the end of a line. The ROW_INC can also be used to skip lines from the input frame. The byte address of each pixel in the frame buffer in the system memory is determined by:

Pixel address = Base address + x × bpp + (y × ((width × bpp) + increment))

where:

- Base address corresponds to the base address (for YUV–NV12 or YUV4:2:0-NV21 format) defined by:
  - DISPC_VID1_BA_0 to DISPC_VID1_BA_1[31-0] BA bit fields
  - DISPC_VID1_BA_UV_0 to DISPC_VID1_BA_UV_1[31-0] BA bit fields
- bpp corresponds to the number of bits per pixel defined by the DISPC_VID1_ATTRIBUTES[6-1] FORMAT bit field.
- width corresponds to the number of pixels per line defined by the DISPC_VID1_SIZE[11-0] SIZEX + 1 bit field.
- increment corresponds to the number of bytes to skip between two contiguous lines defined by the DISPC_VID1_ROW_INC[31-0] ROWINC – 1 bit field.
- x corresponds to the pixel position on the x-axis.
- y corresponds to the pixel position on the y-axis.

NOTE: For YUV420-NV12 or YUV420-NV21 format, the pixel values are defined in two buffers (Y and UV). The first one consists of Y values (8 bits for each Y sample). The second one consists of CbCr values (16 bits for each pair of CbCr samples). The base address of the Y buffers is defined in the DISPC_VID1_BA_0 to DISPC_VID1_BA_1[31-0] BA bit field. The base address of the UV buffers is defined in the DISPC_VID1_BA_UV_0 to DISPC_VID1_BA_UV_1[31-0] BA bit field.

In case of interlaced mode, DISPC_VID1_BA_0 and DISPC_VID1_BA_UV_0 registers define the base address of the even field, and DISPC_VID1_BA_1 and DISPC_VID1_BA_UV_1 registers define the base address of the odd field.

The number of bytes to skip between pixels and between lines are defined using DISPC_VID1_PIXEL_INC and DISPC_VID1_ROW_INC registers, respectivelly. They define the values to be used for the Y buffer. For the CbCr buffer, the HW calculates the corresponding values.

Table 11-407 summarizes the register settings for a simple access of a picture in the system memory.

Table 11-407. DISPC Register Settings for Accessing Image in Internal Memory

Registers	Value
DISPC_VID1_BA_0 to DISPC_VID1_BA_1	PBA, the physical base address of image in the memory
DISPC_VID1_BA_UV_0 to DISPC_VID1_BA_UV_1	PBA, the physical base address of UV buffers image in the memory
DISPC_VID1_PIXEL_INC	1 or other in pixel incremental value
DISPC_VID1_ROW_INC	1 or other in row incremental value

An interconnect request (128 bits) corresponds to one or several pixels, depending on the bits per pixel. Therefore, the DMA engine determines the appropriate burst sequence to optimize the fetching of each new line. The DMA engine must prevent a single burst from crossing two lines. The DMA engine supports only 1D burst. 1D burst is used, if the fetch data is linear in memory. The size of the burst can be one of the following values: 1x128-bit, 2x128-bit, 4x128-bit, or 8x128-bit. The default burst size at reset time is 8 × 128 bits. Because the burst size must be aligned to the burst boundary, in case of misalignment, the DMA engine may issue one or more smaller burst requests.

11.3.4.1.7.2 DISPC Read DMA Buffer

When the vertical front porch (VFP) period starts after the last horizontal front porch (HFP) of the last line, the DMA buffer is flushed according to the VP1 output associated with the VID1 pipeline. The DMA engine restarts fetching data from the memory through the device interconnect. Enabling or disabling the DISPC flushes the DMA buffer.

Programmable high and low thresholds, independent for each DMA buffer, are used by the DMA engine to start and stop requesting data to the device interconnect.

- When low threshold (set in the DISPC_VID1_BUF_THRESHOLD[15-0] BUFLOWTHRESHOLD bit field) is reached, the DMA engine starts a request on the device interconnect to fill the DMA buffer.
- When high threshold (set in the DISPC_VID1_BUF_THRESHOLD[31-16] BUFHIGHTHRESHOLD bit field) is reached, the DMA engine stops requesting encoded pixels.

NOTE: The following limitations on DISPC_VID1_BUF_THRESHOLD[15-0] BUFLOWTHRESHOLD must be considered:

- If VID1 scaler is disabled, BUFLOWTHRESHOLD can be programmed as low as interconnect latency and pixel output rate allow it
- If VID1 scaler is enabled, BUFLOWTHRESHOLD must be programmed to guarantee that at least four full lines can be stored

To avoid underflow at the beginning of a frame and have sufficient encoded pixel data to start some processing, a preloading of the DMA buffer is configurable between a fixed value of bytes or the high threshold value. The preload ensures a minimum number of pixels present in the buffer. When the preload value is reached, the associated channel must start pulling pixels out of the DMA buffer. To enable the preload based on the value entered in the DISPC_VID1_PRELOAD[11-0] PRELOAD bit field, the DISPC_VID1_BUF_THRESHOLD[19] BUFPRELOAD bit must be set to 0x0.

NOTE: When self-refresh mode is selected, meaning the data in the DMA buffer are used for multiple frames, and at the end of each frame, the DMA buffer is not flushed.

11.3.4.1.7.3 DISPC DMA MFLAG Mechanism

The MFLAG mechanism allows a dynamic increase of the priority of DISPC real-time traffic, when required, based on the fullness of the DISPC DMA read buffer.

The MFLAG mechanism is used when fullness of the DMA buffer is critical (close to underflow). The mechanism is implemented for all DMA buffer of VID1 pipeline.

Programmable buffer thresholds (hysteresis) are used to indicate when the local MFLAG signal is generated. The 1-bit MFLAG signal is generated in order to inform the system that the outstanding requests from DISPC shall be considered with higher priority in order to get faster interconnect responses. The MFLAG signal is asynchronous to any ongoing interconnect transaction.

The threshold for VID1 pipeline corresponds to the fullness of the associated DMA buffer, and is defined by two threshold parameters:

- HT_MFLAG: High threshold.
  - For read access from the VID1 pipeline, when the pipeline buffer reaches the programmed value, the associated local MFLAG signal goes low (deasserted).
  - This threshold can be programmed in the DISPC_VID1_MFLAG_THRESHOLD[31-16] HT_MFLAG bit field
- LT_MFLAG: Low threshold.
  - For read access from the VID1 pipeline, when the pipeline buffer reaches the programmed value, the associated local MFLAG signal goes high (asserted).
  - This threshold can be programmed in the DISPC_VID1_MFLAG_THRESHOLD[15-0] LT_MFLAG bit field

Summary of the MFLAG value, based on DMA read buffer fullness:

- If DMA read buffer fullness < DISPC_VID1_MFLAG_THRESHOLD[15-0] LT_MFLAG, then MFLAG = 1
- If DISPC_VID1_MFLAG_THRESHOLD[15-0] LT_MFLAG < DMA read buffer fullness < DISPC_VID1_MFLAG_THRESHOLD[31-16] HT_MFLAG, then MFLAG = 1
- If DMA read buffer fullness > DISPC_VID1_MFLAG_THRESHOLD[31-16] HT_MFLAG, then MFLAG = 0

By default, the MFLAG mechanism is disabled (DISPC_GLOBAL_MFLAG_ATTRIBUTE[1-0] MFLAG_CTRL bit field = 0x0), and the DSS MFLAG out band signal is low (de-asserted).

When the DISPC_GLOBAL_MFLAG_ATTRIBUTE[1-0] MFLAG_CTRL bit field is set to 0x2, the MFLAG mechanism is enabled, and the DSS MFLAG out band signal is dynamically set to 0 or 1, depending on DMA buffer fullness and programmed threshold levels, as explained previously in this section.

The DISPC_GLOBAL_MFLAG_ATTRIBUTE[2] MFLAG_START bit defines additional rules for the MFLAG mechanism:

- If the MFLAG_START bit is set to 0x0 (default value), then in the beginning of the frame when the DMA buffer is empty, the local MFLAG signal is kept at 0 until PRELOAD is reached (for more information on preloading, see Section 11.3.4.1.7.2, *DISPC Read DMA Buffer*. Then, based on the setting of the DISPC_GLOBAL_MFLAG_ATTRIBUTE[1-0] MFLAG_CTRL bit field, the MFLAG signal is generated.
- If the MFLAG_START bit is set to 0x1, then even in the beginning of the frame when the DMA buffer is empty, the DISPC_GLOBAL_MFLAG_ATTRIBUTE[1-0] MFLAG_CTRL bit field defines the generation of the MFLAG signal.

NOTE: On SoC level, the DSS MFLAG signal is mapped to two transaction priorities, which can be configured via the INITIATOR_PRIORITY1 register within BOOT_CFG module. For more information, see Section 5.1.3.1.4, *TeraNet Priority Registers*.

11.3.4.1.7.4 DISPC DMA Predecimation

The predecimation process consists of downscaling an image by fetching only the necessary pixels in the memory. Vertical and horizontal predecimation are possible:

- Vertical predecimation: The picture stored in memory can be predecimated vertically by skipping lines. Burst mode is used to fetch the data when skipping lines. Only the lines that will be used by the DISPC are fetched from memory; the other lines are skipped. The DMA engine sends requests only for the useful lines using 1D burst. The base address indicates the first valid pixel to fetch from memory. The number of lines to skip is set in the DISPC_VID1_ROW_INC[31-0] ROWINC bit field.
- Horizontal predecimation: When fetching data from memory, it is possible to skip 1 of 2 pixel data containers, up to 1 of 2047 pixel data containers, by setting the DISPC_VID1_PIXEL_INC[7-0] PIXELINC bit field to the number of pixel data containers to skip (n), multiplied by the size of a pixel data container (in bytes), +1.

The restriction to horizontal predecimation is that there is at least one useful pixel per 128-bit request. In that case, the DMA engine uses burst mode instead of singles to optimize the requests in terms of latency and SDRAM efficiency.

No decimation is supported when the input format is 1, 2, 4, or 8-bit BITMAP.

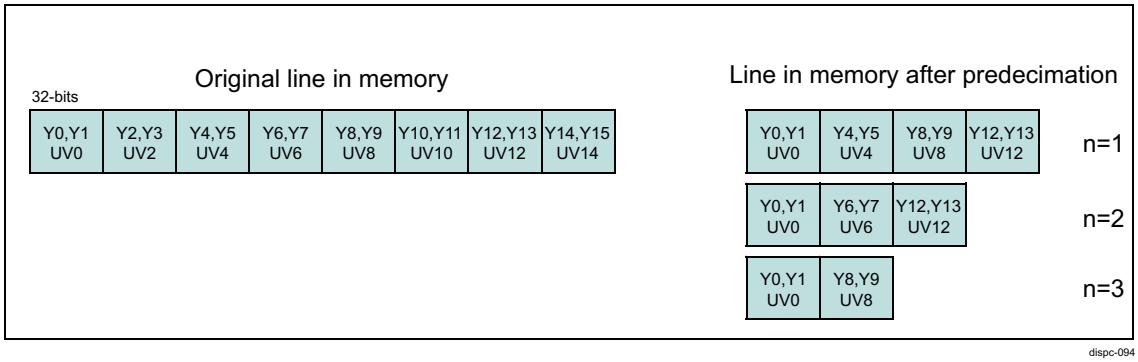
For RGB and YUV4:2:0 data formats, each pixel data container in memory holds 1 pixel. Thus, when configuring the PIXELINC bit field, the value of n equals the number of pixels to skip:

- For RGB format, one pixel data container = 32 bits = 1 pixel
- For YUV format:
  - One Y pixel data container = 8 bits = 1 pixel
  - One UV pixel data container = 16 bits = 1 pixel

For YUV4:2:2 format, each 32-bit pixel data container holds the Luma components for 2 pixels, and the Chrominance component of 1 pixel (see Figure 11-177). Therefore, for the valid values of the PIXELINC bit field in the case of the following YUV4:2:2 format, caution must be taken because n equals the number of pixel data containers to skip, and not the number of pixels:

- For n = 1, PIXELINC = 5
- For n = 2, PIXELINC = 9
- For n = 3, PIXELINC = 13
- For n = 4, PIXELINC = 17, etc.

Figure 11-177. DISPC YUV4:2:2 Predecimation



11.3.4.1.7.5 DISPC DMA Power Modes

11.3.4.1.7.5.1 DISPC DMA Low-Power Mode

The low power mode of the DISPC DMA is achieved by keeping the thresholds (bit-fields [31-16] BUFHIGHTHRESHOLD and [15-0] BUFLOWTHRESHOLD of DISPC_VID1_BUF_THRESHOLD register) farther apart. The farther they are there will be longer periods of idling on the DMA fetch interface resulting in lower power. The user needs to ensure a minimum value of BUFFER_LT, so that there is no underflow.

11.3.4.1.7.5.2 DISPC DMA Ultralow-Power Mode

In ultralow-power mode, the SoC interconnect is used to fill up the DMA buffers to store all the data required to display a full frame. The SoC interconnect is not used to fetch new pixels for the following frames. The data in the DMA buffer are reused to display on the screen.

During the time in which the frames are fetched from the internal DMA buffer, MStandby is asserted, if the DISPC_SYSCONFIG[13-12] MIDLEMODE bit field is set to 0x2 (smart-standby mode).

Two ultralow-power modes can be entered manually or automatically:

- Self-refresh mode: Starting self-refresh mode is done manually by setting the DISPC_VID1_ATTRIBUTES[15] SELFREFRESH bit to 0x1 after capturing a frame in the DMA buffers. Self-refresh mode is stopped by setting the SELFREFRESH bit to 0x0.
- Automatic self-refresh mode: By setting the DISPC_VID1_ATTRIBUTES[17] SELFREFRESHAUTO bit to 0x1, the transition from off to on self-refresh mode is done by hardware after capturing the first frame. The hardware reflects the status of the self-refresh mode by setting the SELFREFRESH bit to 0x1, which means that the data are read inside the DMA buffer without accessing the interconnect and system memory during the frame. Setting the SELFREFRESH bit to 0x0 updates the DMA buffer.

11.3.4.1.8 DISPC Region-Based Mechanism

The region-based mechanism is used in order to display multiple windows using the same video pipeline. The reconfiguration of the pipeline is performed based on a synchronization signal (interrupt) from DISPC, which corresponds to the moment when the first pixel of the window is displayed in order to update the shadow registers of the layer. When all the pixels for the current window have been output, then the pipeline uses the new configuration provided through the shadow registers in order to display the new window re-using the same layer.

It is possible to display unlimited number of windows on the screen using same layer considering the following restriction: only one window horizontally per layer at a given time.

In order to enable the feature, SW needs to set to 0x1 the DISPC_VID1_ATTRIBUTES2[24] REGION_BASED register bit.

It is SW responsibility to handle the reconfiguration of the pipeline based on the different region-based interrupts. For more information, see Section 11.3.4.1.5, *DISPC Interrupt Requests*.

11.3.4.1.9 DISPC Memory Formats

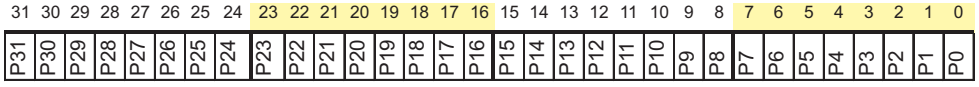
The video pipeline supports various types of memory formats, listed in Table 11-408.

For BITMAP formats the nibble mode can be enabled by setting the DISPC_VID1_ATTRIBUTES[10] NIBBLEMODE bit to 0x1.

Table 11-408. DISPC Memory Formats Support

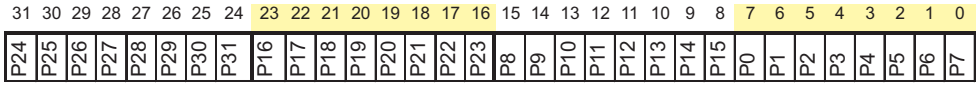
Formats			
BITMAP 1-bpp	xRGB12-4444	xRGB24-8888	UYVY 4:2:2
BITMAP 2-bpp	RGBx12-4444	RGBx24-8888	YUV2 4:2:2
BITMAP 4-bpp	ARGB16-4444	RGB24-888	YUV4:2:0 – NV12
BITMAP 8-bpp	RGBA16-4444	ARGB32-8888	YUV4:2:0 – NV21
	RGB16-565	RGBA32-8888	
	xRGB16-1555	BGRA32-8888	
	ARGB16-1555		

- BITMAP 1-bpp data memory organization (CLUT)



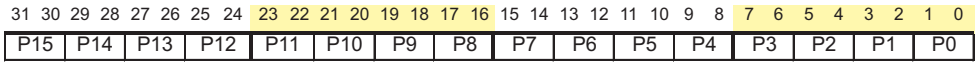
dss-T039

- BITMAP 1-bpp data memory organization (CLUT) in nibble mode



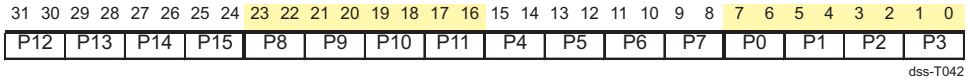
dss-T040

- BITMAP 2-bpp data memory organization (CLUT)



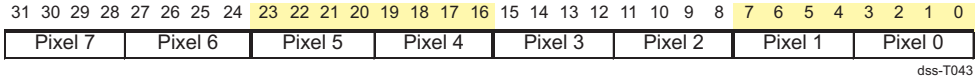
dss-T041

- BITMAP 2-bpp data memory organization (CLUT) in nibble mode



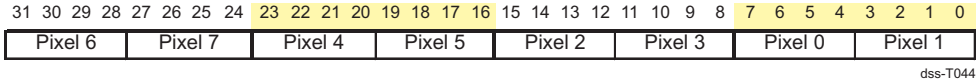
dss-T042

- BITMAP 4-bpp data memory organization (CLUT)



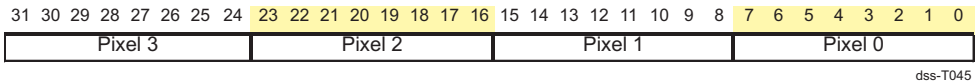
dss-T043

- BITMAP 4-bpp data memory organization (CLUT) in nibble mode



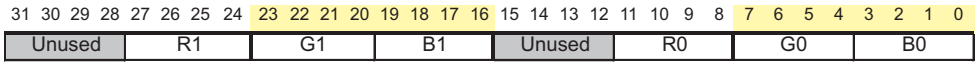
dss-T044

- BITMAP 8-bpp data memory organization (CLUT)



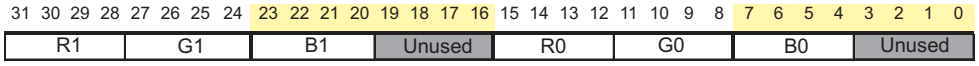
dss-T045

- xRGB12-4444 bpp data memory organization



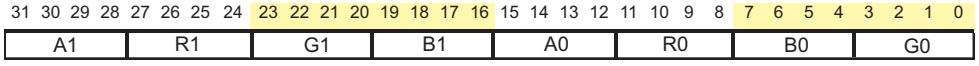
dss-T046

- RGBx12-4444 bpp data memory organization



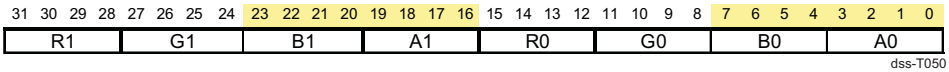
dss-T047

- ARGB16-4444 bpp data memory organization



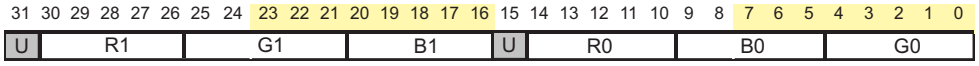
dss-T048

- RGBA16-4444 bpp data memory organization



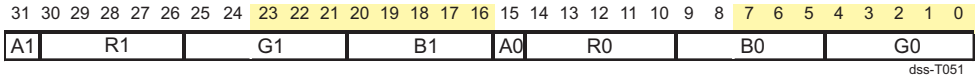
dss-T050

- xRGB16-1555 bpp data memory organization



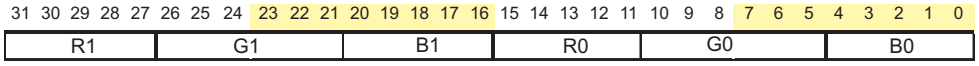
dss-T062

- ARGB16-1555 bpp data memory organization



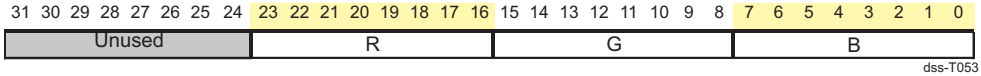
dss-T051

- RGB16-565 bpp data memory organization



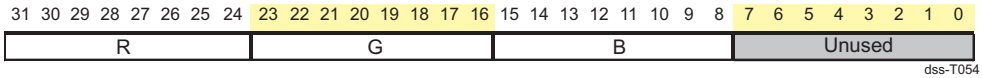
dss-T052

- xRGB24-8888 bpp data memory organization



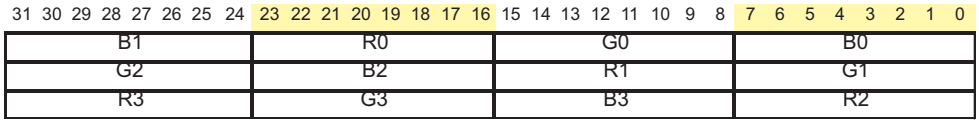
dss-T053

- RGBx24-8888 bpp data memory organization



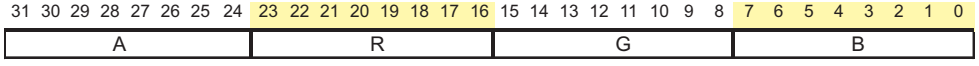
dss-T054

- RGB24-888 bpp packed data memory organization



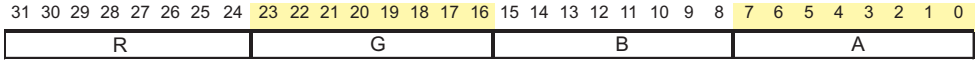
dss-T055

- ARGB32-8888 bpp data memory organization



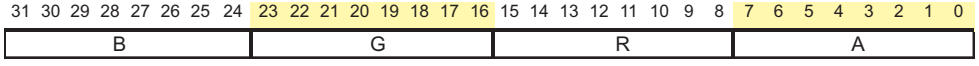
dss-T056

- RGBA32-8888 bpp data memory organization



dss-T057

- BGRA32-8888 bpp data memory organization



dss-T058

- UYVY4:2:2 data memory organization

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0	
Y1	Cr0	Y0	Cb0	+0x0
Y3	Cr2	Y2	Cb2	+0x4
Y5	Cr4	Y4	Cb4	+0x8
Y7	Cr6	Y6	Cb6	+0xC

dss-T059

- YUV2 4:2:2 data memory organization

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0	
Cr0	Y1	Cb0	Y0	+0x0
Cr2	Y3	Cb2	Y2	+0x4
Cr4	Y5	Cb4	Y2	+0x8
Cr6	Y7	Cb6	Y6	+0xC

dss-T060

- YUV4:2:0-NV12 data memory organization (same for YUV4:2:0-NV21, with only UV in reverse order)

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0	
Y3	Y2	Y1	Y0	+0x0
Y7	Y6	Y5	Y4	+0x4

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0	
Cr1	Cb1	Cr0	Cb0	+0x0 + Offset
Cr3	Cb3	Cr2	Cb2	+0x4 + Offset
Cr5	Cb5	Cr4	Cb4	+0x8 + Offset
Cr7	Cb7	Cr6	Cb6	+0xC + Offset

dss-T061

11.3.4.1.10 DISPC Video Pipeline

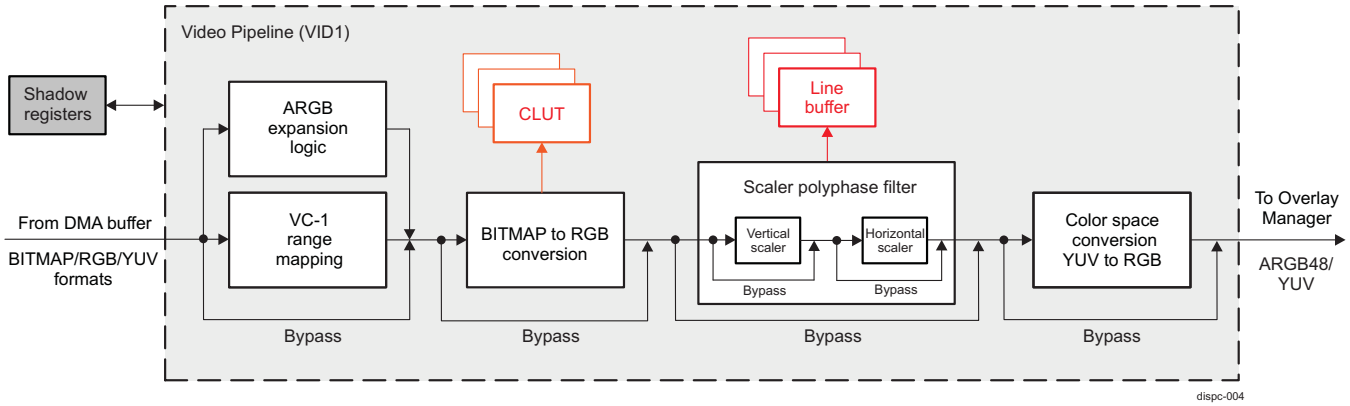
One video pipeline is available, VID1. The input of the video pipeline is connected to the video DMA buffer controller. The VID1 pixel output is always connected to the overlay manager (OVR1). The video pipeline configuration supports various BITMAP, RGB (ARGB and RGBA), and YUV formats, listed in Table 11-408.

The video pipeline consists of:

- VC-1 range mapping unit for YUV422/YUV420 input formats;
- Replication logic for ARGB input formats;
- One 256 x 24-bit entries Color Look-up Table (CLUT);
- Polyphase-filter-based resizer unit (scaler);
- Color Space Conversion (CSC) unit (YUV to RGB);

Each processing block can be independently bypassed.

Figure 11-178. DISPC Video Pipeline Configuration



The 256-entry CLUT is used to convert BITMAP (1, 2, 4, or 8-bit indexed formats) into RGB format.

NOTE: For a BITMAP format data, scaling is not supported. Scaling and color look-up table features are mutually exclusive. If color look-up feature is enabled, then video pipeline scaler has to be disabled.

For chroma sub-sampled YUV formats (YUV422 and YUV420-NV12/NV21) the scaler unit up-samples the chrominance data using both vertical and horizontal polyphase filters.

For ARGB source data with less than/equal to 10-bit component data size the replication logic (ARGB expansion) converts the data to ARGB48 by replicating the MSBs into the LSBs:

- When scaling is disabled, the resulting ARGB48 data is directly provided to VID1 pipeline output;
- When vertical scaling is engaged, the resulting ARGB48 data is first truncated to ARGB8888, and then converted to ARGB10101010 (by MSBs replication into LSBs), before being fed to the vertical scaler input;
- When vertical scaling is disabled, but horizontal scaling is engaged, the resulting ARGB48 data is directly provided to the horizontal scaler input;

The video pipeline is enabled by setting the DISPC_VID1_ATTRIBUTES[0] ENABLE bit to 0x1.

11.3.4.1.10.1 DISPC VID1 Replication Logic

The replication logic (ARGB expansion) converts ARGB pixel formats into ARGB48 format by replicating the MSBs into the LSBs. The logic is always enabled.

Table 11-409 shows how some of the ARGB formats supported by VID1 are remapped into ARGB48.

Table 11-409. DISPC VID1 Replication: ARGB Pixel Formats Remapping into ARGB48-12121212

Formats	A[11:0]	R[11:0]	G[11:0]	B[11:0]
	MSB - LSB	MSB - LSB	MSB - LSB	MSB - LSB
xRGB12-4444	111111111111	R[3:0]R[3:0]R[3:0]	G[3:0]G[3:0]G[3:0]	B[3:0]B[3:0]B[3:0]
RGBx12-4444	111111111111	R[3:0]R[3:0]R[3:0]	G[3:0]G[3:0]G[3:0]	B[3:0]B[3:0]B[3:0]
RGB16-565	111111111111	R[4:0]R[4:0]R[4:3]	G[5:0]G[5:0]	B[4:0]B[4:0]B[4:3]
xRGB16-1555	111111111111	R[4:0]R[4:0]R[4:3]	G[4:0]G[4:0]G[4:3]	B[4:0]B[4:0]B[4:3]
ARGB16-1555	AAAAAAAAAA	R[4:0]R[4:0]R[4:3]	G[4:0]G[4:0]G[4:3]	B[4:0]B[4:0]B[4:3]
ARGB16-4444	A[3:0]A[3:0]A[3:0]	R[3:0]R[3:0]R[3:0]	G[3:0]G[3:0]G[3:0]	B[3:0]B[3:0]B[3:0]
RGBA16-4444	A[3:0]A[3:0]A[3:0]	R[3:0]R[3:0]R[3:0]	G[3:0]G[3:0]G[3:0]	B[3:0]B[3:0]B[3:0]
ARGB32-8888	A[7:0]A[7:4]	R[7:0]R[7:4]	G[7:0]G[7:4]	B[7:0]B[7:4]

11.3.4.1.10.2 DISPC VID1 VC-1 Range Mapping Unit

The VC-1 range mapping unit is used when the video frame picture is decoded using a VC-1 codec by the video accelerator. It remaps the Y, Cb, and Cr components to the full range (from the reduced data range) before displaying. The unit is used primarily for YUV4:2:0-NV12 (NV21) pixel format, but also can be applied to YUV4:2:2 pixel formats (YUV2 and UYVY).

The VC-1 range mapping unit is enabled by setting the DISPC_VID1_ATTRIBUTES2[0] VC1ENABLE bit to 0x1. The VC-1 range mapping should be enabled only for 8 bits/component YUV input data.

The DISPC_VID1_ATTRIBUTES2[3:1] VC1_RANGE_Y and DISPC_VID1_ATTRIBUTES2[6:4] VC1_RANGE_CBCR bit fields are two 3-bit values programmed by the user.

The equations for the mapping process are:

$Y_{out} = CLIP(((Y_{int} – 128) \times (VC1\_RANGE\_Y + 9) + 4) / 8) + 128)$

$C_b = CLIP(((C_b – 128) \times (VC1\_RANGE\_CBCR + 9) + 4) / 8) + 128)$

$C_r = CLIP(((C_r – 128) \times (VC1\_RANGE\_CBCR + 9) + 4) / 8) + 128)$

NOTE: The input and output pixel values are unsigned (Y, Cr, and Cb).

The function CLIP () clips to 0 or 255 when minimum or maximum, respectively, is reached. Otherwise, the resulting output remains identical.

11.3.4.1.10.3 DISPC VID1 Color Look-Up Table (CLUT)

The video pipeline supports conversion of BITMAP formats (1, 2, 4, or 8-bit indexed) into RGB24 format through a Color Look-Up Table (CLUT).

The look-up table consists of 3 separate 256 x 8-bit memories and is indexed by the source BITMAP data. The table is loaded through direct register access by writing to DISPC_VID1_CLUT register.

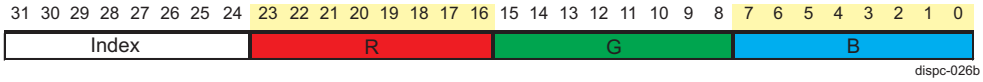
The sequence to load the table is:

1. SW writes (only writes are supported) 32-bit values using single access, or burst access in linear increment burst mode, into DISPC_VID1_CLUT register. The LSB 24 bits [23:0] are used for the value, and the MSB 8 bits [31:24] are used for the index into the table (see Figure 11-179).
2. Loop to Step 1, if there is a new access to the CLUT register. The SW can access other registers between two accesses to the CLUT register.

SW needs to ensure that there is no visible effect when modifying the table, since it is not under HW control.

The usage of the CLUT is activated when a BITMAP format is selected in the DISPC_VID1_ATTRIBUTES[6:1] FORMAT register bit-field.

Figure 11-179. DISPC VID1 CLUT Data Memory Organization



11.3.4.1.10.4 DISPC VID1 Chrominance Resampling

The chrominance resampling from 8-bpc or 10-bpc to 12-bit color components is always performed using filtering (the scaler unit filter). Chrominance resampling and rescaling can be combined to support native rescaling of YUV formats.

The usage of the scaler unit for resampling the chrominance of YUV4:2:0 and YUV4:2:2 is shown in Figure 11-180 and Figure 11-181, respectively. The settings of the scaler unit to perform chrominance resampling are described in Section 11.3.4.1.10.5, *DISPC VID1 Scaler Unit*.

Figure 11-180. DISPC VID1 YUV4:2:0 to ARGB48 Using Scaler Unit for Resampling Chrominance

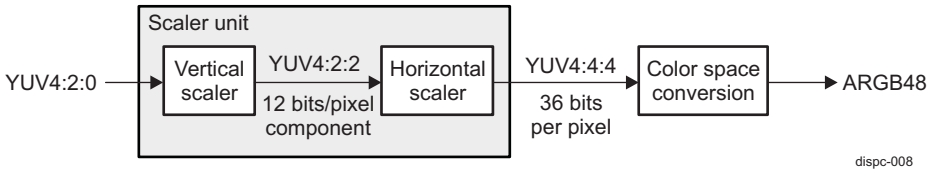
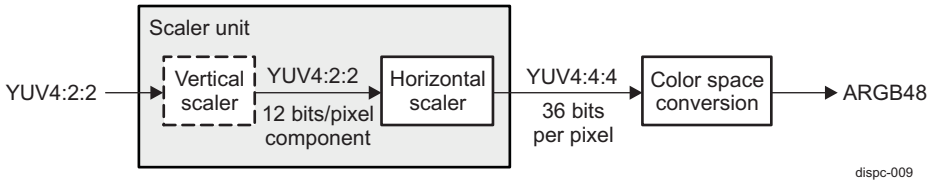


Figure 11-181. DISPC VID1 YUV4:2:2 to ARGB48 Using Scaler Unit for Resampling Chrominance



The vertical scaler is not a mandatory block for resampling chrominance of YUV4:2:2 data, as shown in Figure 11-181.

11.3.4.1.10.5 DISPC VID1 Scaler Unit

The video pipeline resizer unit (scaler) supports 8-bpc and 10-bpc source data, depending on the pixel format.

All video formats are supported, including formats with alpha channel. Alpha channel is scaled with the same parameters as RGB color components. For the YUV formats, Y and Cb/Cr are processed independently. The filter is based on a finite impulse response (FIR) filter with 16 phases. The filter is a 5-tap for horizontal filtering, and can be configured for 3 or 5 taps for vertical filtering.The filtering can be used for different processing:

- Up-sampling of the picture
- Down-sampling of the picture
- Anti-aliasing reduction
- Chrominance resampling in case of YUV data formats

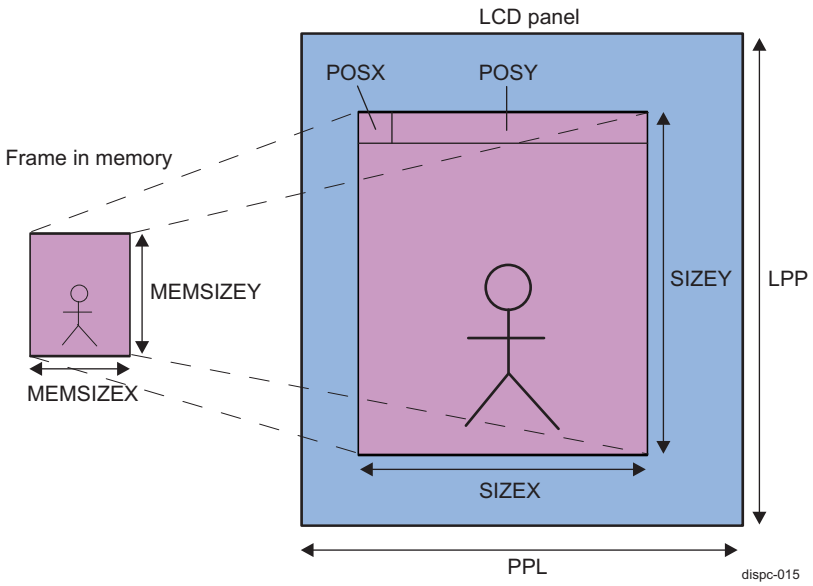
The following limitations must be considered:

- The up-sampling ratio is up to x16.
- The down-sampling ratio using 3-tap configuration is down to x0.5 for RGB format.
- The down-sampling ratio using 5-tap configuration is down to x0.25 for RGB format.

NOTE: The user must set the correct size and position of the original video before resize in order for the up-sampled/down-sampled video to be displayed inside the screen boundaries.

Figure 11-182 shows an example of video up-sampling, with corresponding video window attributes.

Figure 11-182. DISPC Video Up-sampling



The video window attributes can be configured in the following register bit-fields:

- Source data format (input to the scaler unit) in DISPC_VID1_ATTRIBUTES [6-1] FORMAT bit-field
- Video window X-position in DISPC_VID1_POSITION [11-0] POSX bit-field
- Video window Y-position in DISPC_VID1_POSITION [27-16] POSY bit-field
- Video window width in DISPC_VID1_SIZE [11-0] SIZEX bit-field
- Video window height in DISPC_VID1_SIZE [27-16] SIZEY bit-field
- Video picture width in system memory (frame buffer) in DISPC_VID1_PICTURE_SIZE [11-0] MEMSIZEX bit-field
- Video picture height in system memory (frame buffer) in DISPC_VID1_PICTURE_SIZE [27-16]

MEMSIZEY bit-field

- For configuration of LPP and PPL parameters, see Section 11.3.4.1.12.7, *DISPC VP1 Timing Generator and Panel Settings*.

For vertical up-sampling and down-sampling in a 3-tap configuration, the equations are:

For RGB formats

$$Aout(n) = \left(\sum_{i=-1}^{i=1} Ci(\Phi) \times Ain(n+i)\right) >> (Cwidth - (OutWidth - InWidth) - 1)$$

$$Rout(n) = \left(\sum_{i=-1}^{i=1} Ci(\Phi) \times Rin(n+i)\right) >> (Cwidth - (OutWidth - InWidth) - 1)$$

$$Gout(n) = \left(\sum_{i=-1}^{i=1} Ci(\Phi) \times Gin(n+i)\right) >> (Cwidth - (OutWidth - InWidth) - 1)$$

$$Bout(n) = \left(\sum_{i=-1}^{i=1} Ci(\Phi) \times Bin(n+i)\right) >> (Cwidth - (OutWidth - InWidth) - 1)$$

For YUV formats

$$Yout(n) = \left(\sum_{i=-1}^{i=1} Cyi(\Phi y) \times Yin(n+i)\right) >> (Cwidth - (OutWidth - InWidth) - 1)$$

$$Cbout(n) = \left(\sum_{i=-1}^{i=1} Cci(\Phi c) \times Cbin(n+i)\right) >> (Cwidth - (OutWidth - InWidth) - 1)$$

$$Crout(n) = \left(\sum_{i=-1}^{i=1} Cci(\Phi c) \times Crin(n+i)\right) >> (Cwidth - (OutWidth - InWidth) - 1)$$

Component	Vertical	Horizontal
Cwidth (Coefficient Width)	10 bits	10 bits
InWidth (Input Width)	10 bits	12 bits
OutWidth (Output Width)	12 bits	12 bits

dispc-013

(1)

For vertical and horizontal up-sampling and down-sampling in a 5-tap configuration, the equations are:

For RGB formats

$$Aout(n) = \left(\sum_{i=-2}^{i=2} Ci(\Phi) \times Ain(n+i)\right) >> (Cwidth - (OutWidth - InWidth) - 1)$$

$$Rout(n) = \left(\sum_{i=-2}^{i=2} Ci(\Phi) \times Rin(n+i)\right) >> (Cwidth - (OutWidth - InWidth) - 1)$$

$$Gout(n) = \left(\sum_{i=-2}^{i=2} Ci(\Phi) \times Gin(n+i)\right) >> (Cwidth - (OutWidth - InWidth) - 1)$$

$$Bout(n) = \left(\sum_{i=-2}^{i=2} Ci(\Phi) \times Bin(n+i)\right) >> (Cwidth - (OutWidth - InWidth) - 1)$$

For YUV formats

$$Yout(n) = \left(\sum_{i=-2}^{i=2} Cyi(\Phi y) \times Yin(n+i)\right) >> (Cwidth - (OutWidth - InWidth) - 1)$$

$$Cbout(n) = \left(\sum_{i=-2}^{i=2} Cci(\Phi c) \times Cbin(n+i)\right) >> (Cwidth - (OutWidth - InWidth) - 1)$$

$$Crout(n) = \left(\sum_{i=-2}^{i=2} Cci(\Phi c) \times Crin(n+i)\right) >> (Cwidth - (OutWidth - InWidth) - 1)$$

Component	Vertical	Horizontal
Cwidth (Coefficient Width)	10 bits	10 bits
InWidth (Input Width)	10 bits	12 bits
OutWidth (Output Width)	12 bits	12 bits

dispc-012

(2)

NOTE: The pixel (n + 1) is the previous pixel with respect to pixel (n). The line (n + 1) is the previous line with respect to line (n).

The coefficients Ci() depend on the phase between input and output pixels.

The coefficients are different for Y and Cr/Cb filtering because the calculations are independent due to the chrominance resampling for YUV4:2:2 and YUV4:2:0.

First, the vertical filter is applied to the encoded input pixel data, and then the horizontal filter is applied on the resulting pixel values to generate the output pixel values. The vertical input of the filter consists of (see Table 11-410):

- Six lines of 1280 × 32 bits for 5-tap configuration
- Three lines of 2560 x 32 bits for 3-tap configuration

Table 11-410. DISPC VID1 Line Buffer Width for Scaler Unit

Vertical Taps	Maximum Input Width (Pixels)
3	2560
5	1280 (1)

(1) For the 5-tap configuration, the 6th video line is used on the output of the horizontal scaler, so that horizontal down-scaling can be supported with a minimum clock ratio equal to 1. The maximum output width is limited to 1280 pixels in order to be able to support clock ratio of 1 due to the 6th video line buffer.

At the beginning of frame scaling processing, the first line may be duplicated multiple times depending on the initial vertical phase programmed for the poly-phase filter.

At the end of frame scaling processing, the last line is duplicated, if the scaling logic requires loading more lines and the last line has been reached.

Similarly, the first pixel may be duplicated multiple times depending on the initial horizontal phase programmed for the poly-phase filter. The last pixel is duplicated, if the scaling logic requires loading more pixels and the last pixel has been reached.

The programmable coefficients of the polyphase filters are signed 10-bit values (except for the central coefficient, which is unsigned). The vertical video scaler has an 10-bit input and a 12-bit output. The horizontal scaling stage takes the resulting 12-bit input and produces 12-bit output.

Figure 11-183 and Figure 11-184 show the scaler macro-architecture for components A, R, G, and B. Figure 11-185 and Figure 11-186 show the scaler macro-architecture for components Y, Cr, and Cb.

NOTE: The scaling and CSC clipping is set by the same bit, DISPC_VID1_ATTRIBUTES[11] FULLRANGE.

Figure 11-183. DISPC VID1 Macro-Architecture of the Horizontal Scaling for A, R, G, and B Components (5-tap Restriction)

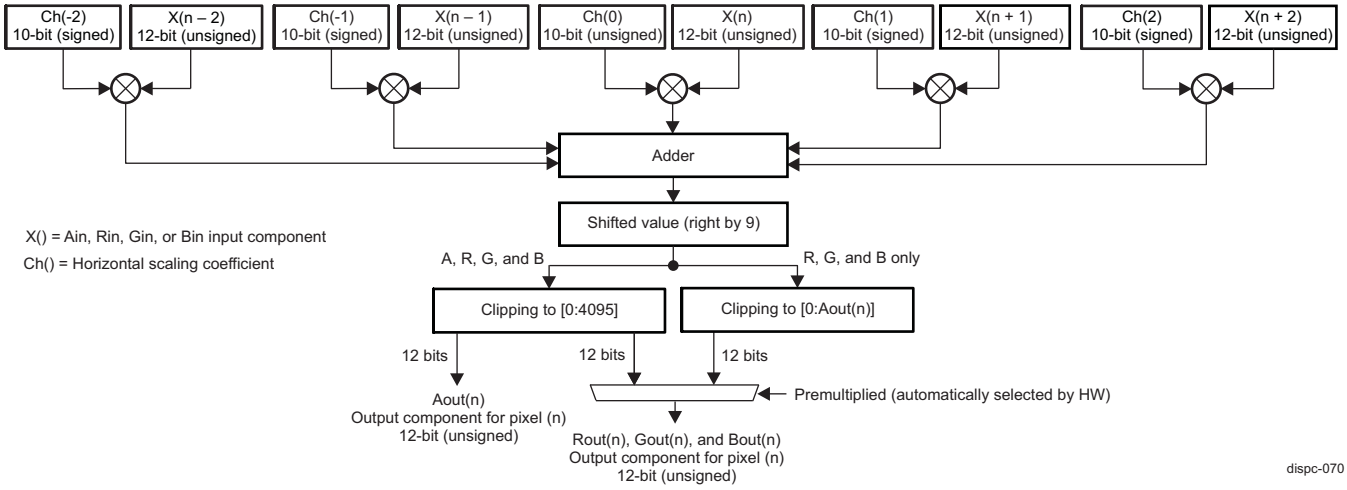


Figure 11-184. DISPC VID1 Macro-Architecture of the Vertical Scaling for A, R, G, and B Components (5 and 3 taps)

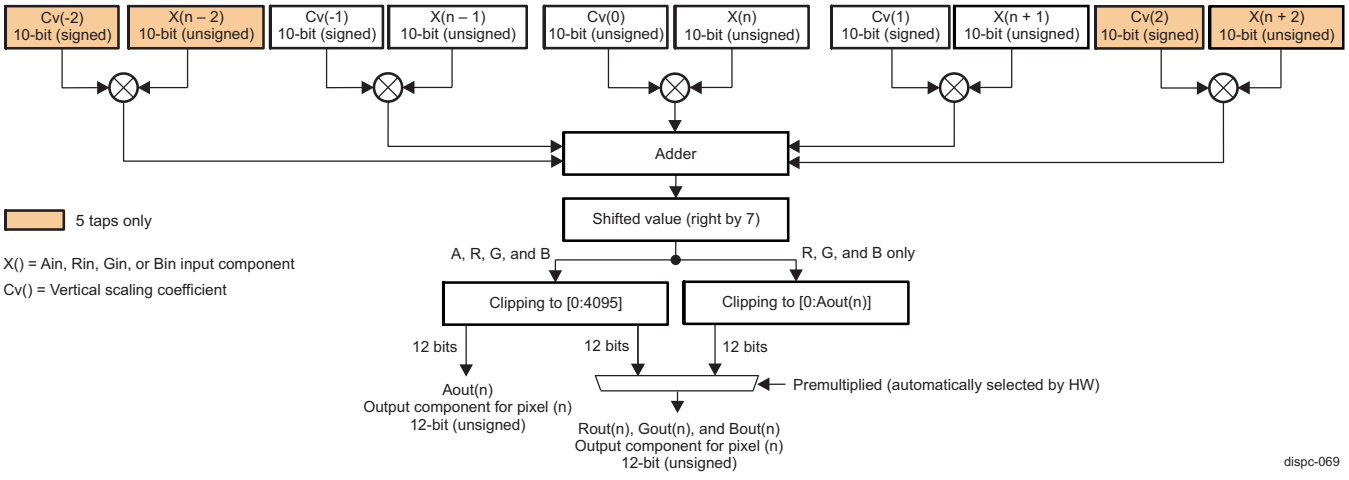


Figure 11-185. DISPC VID1 Macro-Architecture of the Horizontal Scaling for Y, Cr, and Cb Components (5-tap Restriction)

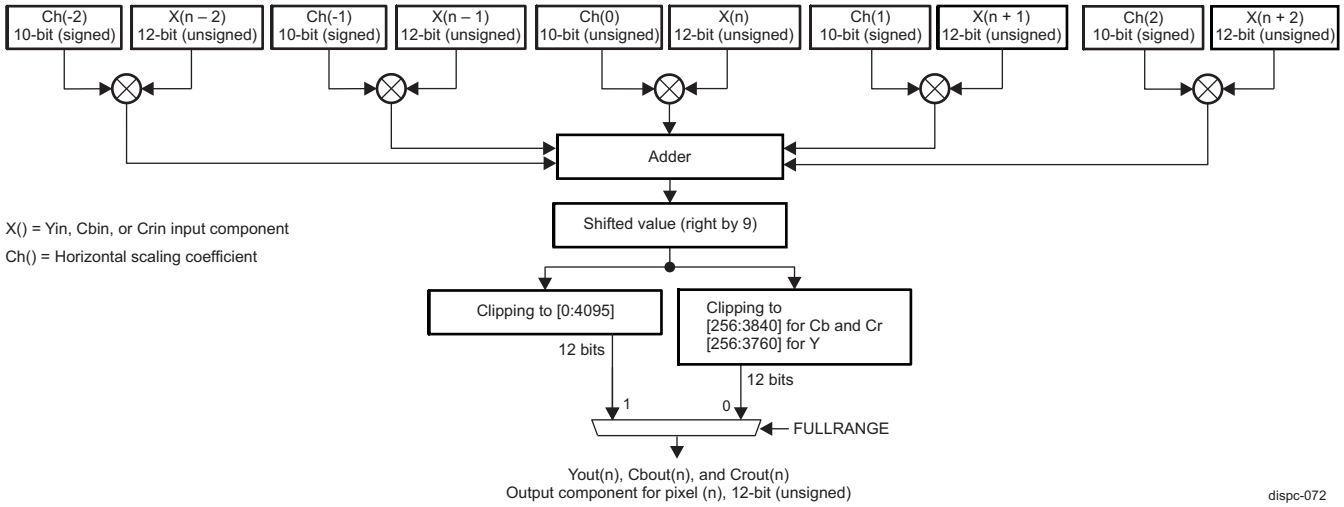


Figure 11-186. DISPC VID1 Macro-Architecture of the Vertical Scaling for Y, Cr, and Cb Components (5 and 3 taps)

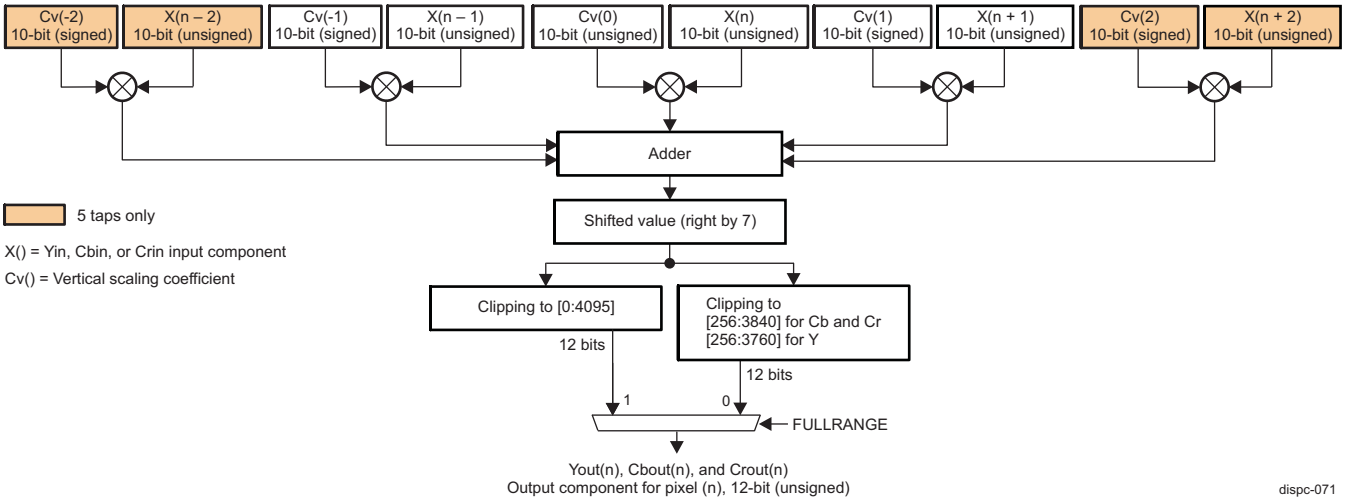


Table 11-411 list the bit-fields in the function to the coefficients for the VID horizontal scaler in the DISPC_VID1_FIR_COEF_H0_0 to DISPC_VID1_FIR_COEF_H0_8, and DISPC_VID1_FIR_COEF_H12_0 to DISPC_VID1_FIR_COEF_H12_15 registers.

Table 11-411. DISPC Register Bit-Fields Associated to Coefficients for ARGB and Y Configuration in VID Horizontal Scaler

Phases	Ch(2)	Ch(1)	Ch(0)	Ch(-1)	Ch(-2)
	Signed coefficient [29-20] FIRHC2 bitfield	Signed coefficient [19-10] FIRHC1 bitfield	Unsigned central coefficient [9-0] FIRHC0 bitfield	Signed coefifcient [19-10] FIRHC1 bitfield	Signed coefficient [29-20] FIRHC2 bitfield
0	DISPC_VID1_FIR_COEF_H12_0	DISPC_VID1_FIR_COEF_H12_0	DISPC_VID1_FIR_COEF_H0_0	DISPC_VID1_FIR_COEF_H12_0	DISPC_VID1_FIR_COEF_H12_0
1	DISPC_VID1_FIR_COEF_H12_1	DISPC_VID1_FIR_COEF_H12_1	DISPC_VID1_FIR_COEF_H0_1	DISPC_VID1_FIR_COEF_H12_15	DISPC_VID1_FIR_COEF_H12_15
2	DISPC_VID1_FIR_COEF_H12_2	DISPC_VID1_FIR_COEF_H12_2	DISPC_VID1_FIR_COEF_H0_2	DISPC_VID1_FIR_COEF_H12_14	DISPC_VID1_FIR_COEF_H12_14
3	DISPC_VID1_FIR_COEF_H12_3	DISPC_VID1_FIR_COEF_H12_3	DISPC_VID1_FIR_COEF_H0_3	DISPC_VID1_FIR_COEF_H12_13	DISPC_VID1_FIR_COEF_H12_13

Table 11-411. DISPC Register Bit-Fields Associated to Coefficients for ARGB and Y Configuration in VID Horizontal Scaler (continued)

Phases	Ch(2)	Ch(1)	Ch(0)	Ch(-1)	Ch(-2)
4	DISPC_VID1_FIR_COEF_H12_4	DISPC_VID1_FIR_COEF_H12_4	DISPC_VID1_FIR_COEF_H0_4	DISPC_VID1_FIR_COEF_H12_12	DISPC_VID1_FIR_COEF_H12_12
5	DISPC_VID1_FIR_COEF_H12_5	DISPC_VID1_FIR_COEF_H12_5	DISPC_VID1_FIR_COEF_H0_5	DISPC_VID1_FIR_COEF_H12_11	DISPC_VID1_FIR_COEF_H12_11
6	DISPC_VID1_FIR_COEF_H12_6	DISPC_VID1_FIR_COEF_H12_6	DISPC_VID1_FIR_COEF_H0_6	DISPC_VID1_FIR_COEF_H12_10	DISPC_VID1_FIR_COEF_H12_10
7	DISPC_VID1_FIR_COEF_H12_7	DISPC_VID1_FIR_COEF_H12_7	DISPC_VID1_FIR_COEF_H0_7	DISPC_VID1_FIR_COEF_H12_9	DISPC_VID1_FIR_COEF_H12_9
8	DISPC_VID1_FIR_COEF_H12_8	DISPC_VID1_FIR_COEF_H12_8	DISPC_VID1_FIR_COEF_H0_8	DISPC_VID1_FIR_COEF_H12_8	DISPC_VID1_FIR_COEF_H12_8
9	DISPC_VID1_FIR_COEF_H12_9	DISPC_VID1_FIR_COEF_H12_9	DISPC_VID1_FIR_COEF_H0_7	DISPC_VID1_FIR_COEF_H12_7	DISPC_VID1_FIR_COEF_H12_7
10	DISPC_VID1_FIR_COEF_H12_10	DISPC_VID1_FIR_COEF_H12_10	DISPC_VID1_FIR_COEF_H0_6	DISPC_VID1_FIR_COEF_H12_6	DISPC_VID1_FIR_COEF_H12_6
11	DISPC_VID1_FIR_COEF_H12_11	DISPC_VID1_FIR_COEF_H12_11	DISPC_VID1_FIR_COEF_H0_5	DISPC_VID1_FIR_COEF_H12_5	DISPC_VID1_FIR_COEF_H12_5
12	DISPC_VID1_FIR_COEF_H12_12	DISPC_VID1_FIR_COEF_H12_12	DISPC_VID1_FIR_COEF_H0_4	DISPC_VID1_FIR_COEF_H12_4	DISPC_VID1_FIR_COEF_H12_4
13	DISPC_VID1_FIR_COEF_H12_13	DISPC_VID1_FIR_COEF_H12_13	DISPC_VID1_FIR_COEF_H0_3	DISPC_VID1_FIR_COEF_H12_3	DISPC_VID1_FIR_COEF_H12_3
14	DISPC_VID1_FIR_COEF_H12_14	DISPC_VID1_FIR_COEF_H12_14	DISPC_VID1_FIR_COEF_H0_2	DISPC_VID1_FIR_COEF_H12_2	DISPC_VID1_FIR_COEF_H12_2
15	DISPC_VID1_FIR_COEF_H12_15	DISPC_VID1_FIR_COEF_H12_15	DISPC_VID1_FIR_COEF_H0_1	DISPC_VID1_FIR_COEF_H12_1	DISPC_VID1_FIR_COEF_H12_1

NOTE: The Table 11-411 cells without color are duplicated from the grey cells.

Similar table approach applies to the vertical scaler (registers DISPC_VID1_FIR_COEF_V0_0 to DISPC_VID1_FIR_COEF_V0_8, and DISPC_VID1_FIR_COEF_V12_0 to DISPC_VID1_FIR_COEF_V12_15 are used).

Similar table approach applies to the coefficients for Cb/Cr filtering in case of YUV format (registers DISPC_VID1_FIR_COEF_H0_C_0 to DISPC_VID1_FIR_COEF_H0_C_8, and DISPC_VID1_FIR_COEF_H12_C_0 to DISPC_VID1_FIR_COEF_H12_C_15, and DISPC_VID1_FIR_COEF_V0_C_0 to DISPC_VID1_FIR_COEF_V0_C_8 and DISPC_VID1_FIR_COEF_V12_C_0 to DISPC_VID1_FIR_COEF_V12_C_15) .

The VID1 scaler unit vertical or/and horizontal sampling is selected by configuring the DISPC_VID1_ATTRIBUTES[8-7] RESIZEENABLE bit field.

A set of configurations must be valid before enabling the video up-sampling and down-sampling block.

The following fields define the configuration of the video up-sampling/down-sampling block for VID1 pipeline:

- Vertical upsampling and downsampling increments the value of the [23-0] FIRVINC bit field in DISPC_VID1_FIRV and DISPC_VID1_FIRV2 registers. The unsigned integer value range is 2^{23}. Software calculates the value using the following equation:

$$FIRVINC = 2^{21} * \left(\frac{MEMSIZEY+1}{SIZEY+1}\right) \quad (3)$$

dispc-066

- Horizontal upsampling and downsampling increments the value of the the [23-0] FIRHINC bit field in the DISPC_VID1_FIRH and DISPC_VID1_FIRH2 registers. The unsigned integer value range is 2^{23}. Software calculates the value using the following equation:

$$FIRHINC = 2^{21} * \left(\frac{MEMSIZEH+1}{SIZEH+1}\right)$$

dispc-067 (4)

- Vertical up/downsampling accumulator value, [23-0] VERTICALACCU bit field in DISPC_VID1_ACCUV_0 to DISPC_VID1_ACCUV_1 and DISPC_VID1_ACCUV2_0 to DISPC_VID1_ACCUV2_1 registers: The accumulator value indicates on which phase the vertical filtering starts. The register DISPC_VID1_ACCUV_0 is used for progressive output, and for interlace output DISPC_VID1_ACCUV_0 and DISPC_VID1_ACCUV_1 registers are used. Similarly, DISPC_VID1_ACCUV2_0 and DISPC_VID1_ACCUV2_1 are used in progressive or interlace output to set the accumulator value of the Cb and Cr components when scaling YUV format.
- Vertical upsampling and downsampling line buffer configuration DISPC_VID1_ATTRIBUTES[21] VERTICALTAPS bit: The default value at reset time is 0x0 (3-tap configuration is used). If the bit field is reset, the 3-tap configuration is used.
- Horizontal upsampling and downsampling accumulator value, [23-0] HORIZONTALACCU bit field in DISPC_VID1_ACCUH_0 to DISPC_VID1_ACCUH_1, and DISPC_VID1_ACCUH2_0 to DISPC_VID1_ACCUH2_1 registers: The accumulator value indicates on which phase the horizontal filtering starts. The register DISPC_VID1_ACCUH_0 is used for progressive output, and for interlace output the DISPC_VID1_ACCUH_0 and DISPC_VID1_ACCUH_1 registers are used. Similarly, DISPC_VID1_ACCUH2_0 and DISPC_VID1_ACCUH2_1 are used in progressive or interlace output to set the accumulator value of the Cb and Cr components when scaling YUV format.

Table 11-412 lists the DISPC vertical and horizonatal accumulator values and phases.

Table 11-412. DISPC VID1 Vertical and Horizontal Accumulator Phases

Accumulator Value (MSB bits)	Phases f
0	0
256 or -3840	1
512 or -3584	2
768 or -3328	3
1024 or -3072	4
1280 or -2816	5
1536 or -2560	6
1792 or -2304	7
2048 or -2048	8
2304 or -1792	9
2560 or -1536	10
2816 or -1280	11
3072 or -1024	12
3328 or -768	13
3584 or -512	14
3840 or -256	15

- Vertical upsampling and downsampling central coefficients:
  - The vertical upsampling and downsampling central coefficients are defined in the DISPC_VID1_FIR_COEF_V0_0 to DISPC_VID1_FIR_COEF_V0_8 registers. There are 9 registers for the 16 phases with 1 coefficient for each of them. Symetrical implementation is used, so only 9 coefficients are used. Each register contains one 10-bit unsigned coefficient (the central one).
  - Four YUV, the vertical upsampling and downsampling central coefficients are set in DISPC_VID1_FIR_COEF_V0_C_0 to DISPC_VID1_FIR_COEF_V0_C_8 registers.
- Vertical upsampling and downsampling coefficients:
  - The vertical upsampling and downsampling coefficients are defined in the DISPC_VID1_FIR_COEF_V12_0 to DISPC_VID1_FIR_COEF_V12_15 registers. There are 16

registers for the 16 phases with 2 coefficient for each of them, so a total of 32 programmable coefficients for the vertical up/down-sampling block. Each register contains two 10-bit signed coefficients.

  - Four YUV, the vertical upsampling and downsampling coefficients are set in DISPC_VID1_FIR_COEF_V12_C_0 to DISPC_VID1_FIR_COEF_V12_C_15 registers.

- Horizontal upsampling and downsampling central coefficients:
  - The horizontal upsampling and downsampling central coefficients are defined in the DISPC_VID1_FIR_COEF_H0_0 to DISPC_VID1_FIR_COEF_H0_8 registers. There are 9 registers for the 16 phases with 1 coefficient for each of them. Symetrical implementation is used, so only 9 coefficients are used. Each register contains one 10-bit unsigned coefficient (the central one).
  - Four YUV, the horizontal upsampling and downsampling central coefficients are set in DISPC_VID1_FIR_COEF_H0_C_0 to DISPC_VID1_FIR_COEF_H0_C_8 registers.
- Horizontal upsampling and downsampling coefficients:
  - The horizontal upsampling and downsampling coefficients are defined in the DISPC_VID1_FIR_COEF_H12_0 to DISPC_VID1_FIR_COEF_H12_15 registers. There are 16 registers for the 16 phases with 2 coefficient for each of them, so a total of 32 programmable coefficients for the horizontal up/down-sampling block. Each register contains two 10-bit signed coefficients.
  - Four YUV, the horizontal upsampling and downsampling coefficients are set in DISPC_VID1_FIR_COEF_H12_C_0 to DISPC_VID1_FIR_COEF_H12_C_15 registers.

11.3.4.1.10.6 DISPC VID1 CSC Unit - YUV to RGB

The CSC unit converts the video-encoded pixel values from YUV4:4:4 format into ARGB48 format (12-bit value per component A, R, G, and B, with A fixed at 0xFFF).

In case of YUV4:2:0 or YUV4:2:2 formats, a chrominance resampling to YUV4:4:4 is performed before converting the YUV into RGB values (see Section 11.3.4.1.10.4, *DISPC VID1 Chrominance Resampling*). The YUV4:2:2 or YUV4:2:0 to YUV4:4:4 chrominance resampling is a pre-processing to the color space conversion.

Figure 11-187 and Figure 11-188 show the 3 × 3 11-bit coefficients used to convert from YUV4:4:4 into ARGB48. The value of A component is fixed at 0xFFF in the output. The coefficients are set according to the standard used to encode the pixel data in YUV color space. Table 11-413 summarizes the coefficients with their respective register bit fields.

Table 11-413. DISPC VID1 CSC - YUV to RGB Register Bitfield Settings

Coefficients	Bit Field Registers
R_Y	DISPC_VID1_CONV_COEF0[10-0] RY
R_{Cr}	DISPC_VID1_CONV_COEF0[26-16] RCR
R_{Cb}	DISPC_VID1_CONV_COEF1[10-0] RCB
G_Y	DISPC_VID1_CONV_COEF1[26-16] GY
G_{Cr}	DISPC_VID1_CONV_COEF2[10-0] GCR
G_{Cb}	DISPC_VID1_CONV_COEF2[26-16] GCB
B_Y	DISPC_VID1_CONV_COEF3[10-0] BY
B_{Cr}	DISPC_VID1_CONV_COEF3[26-16] BCR
B_{Cb}	DISPC_VID1_CONV_COEF4[10-0] BCB
G offset	DISPC_VID1_CONV_COEF5[31-19] GOFFSET
R offset	DISPC_VID1_CONV_COEF5[15-3] ROFFSET
B offset	DISPC_VID1_CONV_COEF6[15-3] BOFFSET

If the active range for the luminance samples (Y) is [256:3760] and [256:3840] for the chrominance samples (Cb and Cr), the range selection is done by setting the DISPC_VID1_ATTRIBUTES[11] FULLRANGE bit to 0x0. The values of R, G, and B output components are clipped to the range [0:4095].

NOTE: The scaling and CSC clipping is set by the same bit, DISPC_VID1_ATTRIBUTES[11] FULLRANGE.

Figure 11-187. DISPC VID1 YCbCr to RGB Registers (FULLRANGE = 0), 12-Bit Outputs

$$\begin{bmatrix} R_{OUT} \\ G_{OUT} \\ B_{OUT} \end{bmatrix} = \frac{1}{256} * \begin{bmatrix} R_Y & R_{cr} & R_{cb} \\ G_Y & G_{cr} & G_{cb} \\ B_Y & B_{cr} & B_{cb} \end{bmatrix} * \begin{bmatrix} Y_{IN} - 256 \\ Cr_{IN} - 2048 \\ Cb_{IN} - 2048 \end{bmatrix}$$

dispc-005

In Figure 11-187, the R, G, B offset values -256, -2048, -2048 are programmed via corresponding bit-fields in DISPC_VID1_CONV_COEF5 and DISPC_VID1_CONV_COEF6 registers (see Table 11-413, *DISPC VID1 CSC - YUV to RGB Register Bitfield Settings*).

If the active range for the luminance samples (Y) and chrominance samples (Cb and Cr) is [0:4095], the range selection is done by setting the DISPC_VID1_ATTRIBUTES[11] FULLRANGE bit to 0x1. The values of R, G, and B output components are clipped to the range [0:4095].

Figure 11-188. DISPC VID1 YCbCr to RGB Registers (FULLRANGE = 1), 12-Bit Outputs

$$\begin{bmatrix} R_{OUT} \\ G_{OUT} \\ B_{OUT} \end{bmatrix} = \frac{1}{256} * \begin{bmatrix} R_Y & R_{cr} & R_{cb} \\ G_Y & G_{cr} & G_{cb} \\ B_Y & B_{cr} & B_{cb} \end{bmatrix} * \begin{bmatrix} Y_{IN} \\ Cr_{IN} - 2048 \\ Cb_{IN} - 2048 \end{bmatrix}$$

dispc-006

In Figure 11-188, the G and B offset values -2048 and -2048 are programmed via corresponding bit-fields in DISPC_VID1_CONV_COEF5 and DISPC_VID1_CONV_COEF6 registers (see Table 11-413, *DISPC VID1 CSC - YUV to RGB Register Bitfield Settings*).

11.3.4.1.11 DISPC Overlay Manager

DISPC implements a single Overlay Manager (OVR1) with output assigned to the Video Port (VP1) (see Figure 11-176, *DISPC Architecture Overview*).

Only background color (programmable constant background color) feature is available. The default solid background color is defined in DEFAULTCOLOR bit-fields of DISPC_OVR1_DEFAULT_COLOR and DISPC_OVR1_DEFAULT_COLOR2 registers.

The output of OVR1 is connected to VP1 Color Phase Rotation (CPR) block through the Gamma table.

11.3.4.1.12 DISPC Video Port Output

DISPC implemets a single video port output (VP1). It processes data received from Overlay Manager 1 (OVR1). The VP1 output path consist of several processing blocks (see Figure 11-189):

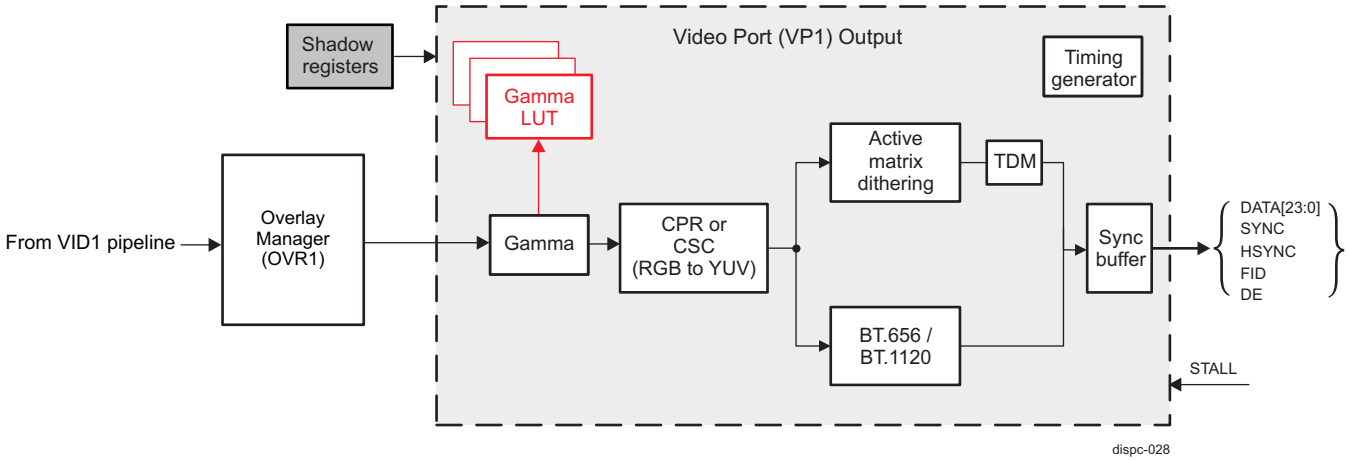
- Gamma correction unit
- Color phase rotation (CPR) (also used for RGB-to-YUV conversion)
- Active matrix dithering with TDM
- BT.656
- BT.1120
- Timing generator

The display subsystem supports active matrix display technology. The configuration of colors depends on the color depth:

- 24 bpp supports 16,777,216 colors.

- 18 bpp supports 262,144 colors.
- 16 bpp supports 65,536 colors.
- 12 bpp supports 4096 colors.

Figure 11-189. DISPC VP1 Output Architecture



NOTE: In BT.656 mode only bits 9 through 0 are used from DATA[23:0] data bus.

11.3.4.1.12.1 DISPC VP1 Gamma Correction Unit

When performing gamma correction, the selected encoded pixel values by the overlay manager from the video path are sent to the gamma curve table. Each component of encoded pixel value is used as a pointer to index 1 out of 256 x 24-bit gamma curve entries in the table. Each 8-bit component is replaced with the 8-bit table value corresponding to R, G, or B component. The table is loaded by software via the DISPC_VP1_GAMMA_TABLE register. It is possible to load only part of the table. For each access to the table, the 24-bit value is associated with index in the table by concatenating the 24-bit value (LSB of 32-bit access) and the 8-bit index value (MSB of the 32-bit access).

The sequence to load the gamma table is:

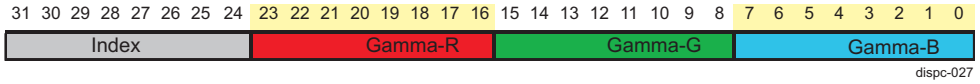
1. SW writes (only writes are supported) 32-bit gamma correction values using single access, or burst access in streaming mode, into DISPC_VP1_GAMMA_TABLE register. The LSB 24 bits [23:0] are used for the value, and the MSB 8 bits [31:24] are used for the index into the table.
2. Loop to Step 1, if there is a new access to the gamma table register. The SW can access other registers between two accesses to the gamma table register.

SW needs to ensure that there is no visible effect when modifying the table, since it is not under HW control.

The usage of the gamma table is activated by setting DISPC_VP1_CONFIG[2] GAMMAENABLE register bit to 0x1.

Figure 11-190 describes the format of one of the gamma curve values in the memory.

Figure 11-190. DISPC Data Memory Organization for Gamma Mode in VP1 Output



NOTE: Because gamma registers are not 32-byte aligned, they cannot be accessed using EDMA constant mode addressing. For direct memory access to multiple words to/from these registers, the EDMA should be programmed to do multiple 4-byte accesses to the register address in increment mode (DAM/SAM = 0) by setting the appropriate ACNT=4 and BIDX=0x0.

11.3.4.1.12.2 DISPC VP1 Color Phase Rotation Unit

The Color Phase Rotation (CPR) unit can be used to correct the display output colorimetry in case of nonpure white backlight.

The CPR is enabled by setting the DISPC_VP1_CONFIG[15] CPR bit to 0x1. The coefficients are programmed in the following registers:

- Red 10-bit signed coefficients in DISPC_VP1_CPR_COEF_R
- Green 10-bit signed coefficients in DISPC_VP1_CPR_COEF_G
- Blue 10-bit signed coefficients in DISPC_VP1_CPR_COEF_B

The CPR logic is integrated after the overlay manager while using the gamma correction and before the spatial/temporal dithering. The CPR can be selected to correct the nonpure white backlight of the display module by using a programmable matrix to convert the 36-bit RGB pixel value into a new 30-bit RGB pixel value. The matrix is programmed through a set of nine 10-bit signed coefficients. The output of the calculation is clipped to [0:4095]. The CPR is processed by the equation shown in Figure 11-191. Table 11-414 lists all coefficients with their respective register bitfields for settings.

Figure 11-191. DISPC VP1 CPR Matrix

$$\begin{bmatrix} R_{out} \\ G_{out} \\ B_{out} \end{bmatrix} = \begin{bmatrix} Rr & Rg & Rb \\ Gr & Gg & Gb \\ Br & Bg & Bb \end{bmatrix} * \begin{bmatrix} R_{in} \\ G_{in} \\ B_{in} \end{bmatrix}$$

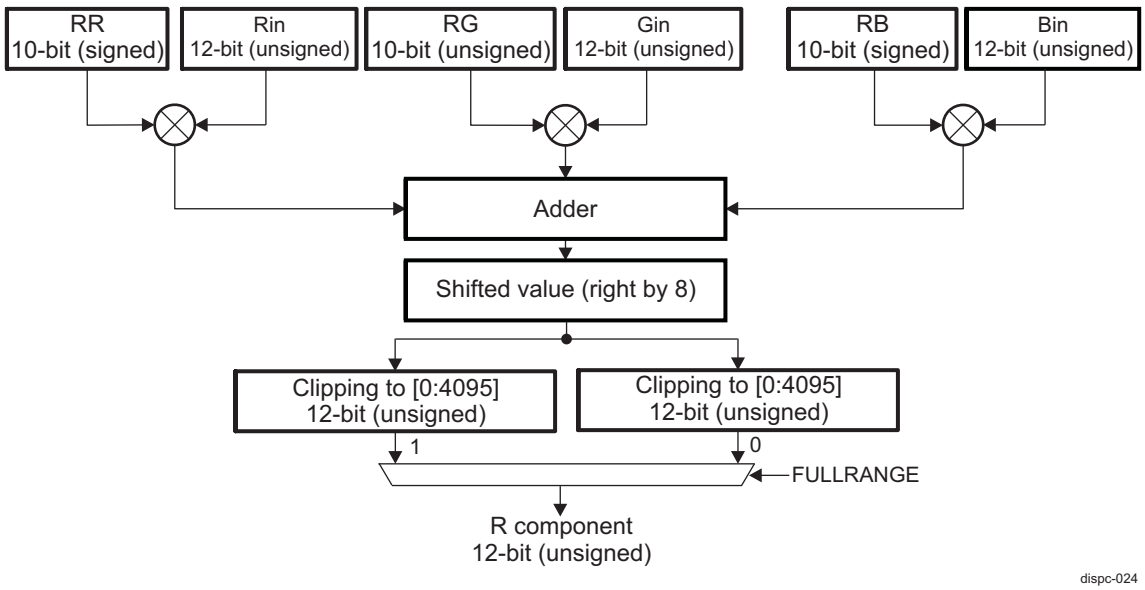
dispc-023

Table 11-414. DISPC VP1 CPR, or RGB to YUV Conversion Coefficients with Associated Register Bitfields

Registers	Bit Field	Color Phase Rotation	RGB to YUV
DISPC_VP1_CPR_COEF_R	RR	Rr	Yr
	RG	Rg	Yg
	RB	Rb	Yb
DISPC_VP1_CPR_COEF_G	GR	Gr	Cbr
	GG	Gg	Cbg
	GB	Gb	Cbb
DISPC_VP1_CPR_COEF_B	BR	Br	Crr
	BG	Bg	Crg
	BB	Bb	Crb

Figure 11-192 shows the CPR macro-architecture.

Figure 11-192. DISPC VP1 CPR Macro-Architecture



11.3.4.1.12.3 DISPC VP1 Color Space Conversion

The CPR block can be used to perform Color Space Conversion (CSC). Table 11-414 lists the associated coefficients with their respective register bit-fields. The CSC logic uses the CPR registers for the coefficients. Only the upper 10 bits of each color component are used for the CSC processing. The CSC processing is enabled by setting the DISPC_VP1_CONFIG[24] COLORCONVENABLE register bit to 0x1.

The color space conversion on the VP1 output can be selected in order to convert from 30-bit RGB to YUV444. The matrix is programmed through a set of nine 10-bit signed coefficients. The result is clipped to [64:940] for the luminance and [64:960] for the chrominance, when full-range equals zero (DISPC_VP1_CONFIG[25] FULLRANGE = 0).

Figure 11-193. DISPC VP1 CSC RGB to YUV Registers (FullRange=0)

$$\begin{bmatrix} Y_{OUT} \\ Cr_{OUT} \\ Cb_{OUT} \end{bmatrix} = \frac{1}{256} * \begin{bmatrix} Y_R & Y_G & Y_B \\ Cr_R & Cr_G & Cr_B \\ Cb_R & Cb_G & Cb_B \end{bmatrix} * \begin{bmatrix} R_{IN} \\ G_{IN} \\ B_{IN} \end{bmatrix} + \begin{bmatrix} 64 \\ 512 \\ 512 \end{bmatrix}$$

dispc-098

If the programmed active range for the luminance samples (Y) and chrominance samples (Cb and Cr) is [0:1023], the values Y, Cb, and Cr are clipped to the range [0:1023]. The following equation gives the 11-bit coefficients of the RGB to YUV color space conversion, for full-range (DISPC_VP1_CONFIG[25] FULLRANGE = 1)

Figure 11-194. DISPC VP1 CSC RGB to YUV Registers (FullRange=1)

$$\begin{bmatrix} Y_{OUT} \\ Cr_{OUT} \\ Cb_{OUT} \end{bmatrix} = \frac{1}{256} * \begin{bmatrix} Y_R & Y_G & Y_B \\ Cr_R & Cr_G & Cr_B \\ Cb_R & Cb_G & Cb_B \end{bmatrix} * \begin{bmatrix} R_{IN} \\ G_{IN} \\ B_{IN} \end{bmatrix} + \begin{bmatrix} 0 \\ 512 \\ 512 \end{bmatrix}$$

dispc-099

In order to provide YUV422 data to the BT-656 or BT-1120 blocks, the YUV444 format is converted to YUV422 by sub-sampling the chrominance values. The HW does this by averaging two-by-two the chrominance samples. The conversion from YUV444 to YUV422 is performed when the color space conversion is enabled.

11.3.4.1.12.4 DISPC VP1 BT.656 and BT.1120 Modes

The VP1 output can be configured in BT.656 or BT.1120 mode. The following standards are not supported in BT.656 mode:

- BT.470 (Support for conventional Analog TV Systems)
- BT.803 (The avoidance of interference generated by digital television studio equipment)
- BT.1364 (Format of ancillary data signals carried in digital component studio interfaces)

Unsupported formats when BT.1120 mode is used:

- BT.1364 (Support for Ancillary Data during blanking period)

BT.656/BT.1120 modes use embedded EAV/SAV syncs.

Enabling BT.656 or BT.1120 format for VP1 ouput is done by setting DISPC_VP1_CONFIG[20] BT656ENABLE or [21] BT1120ENABLE register bits, respectively

NOTE: It is not posssible to enable BT.656 and BT.1120 mode simultaneously on the same output.

Figure 11-195 shows signal mapping on DSS_DATA[23:0] data bus. Bits 9 to 0 are used in BT.656 mode (10-bit).

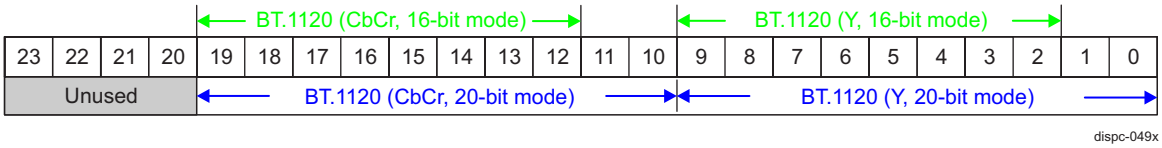
Figure 11-195. DISPC VP1 Data Mapping in BT.656 Mode

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Unused														BT.656									

NOTE: For compatibility with existing 8-bit interfaces, the two LSB are ignored, and only bits [9-2] are used.

Figure 11-196 shows signal mapping on DSS_DATA[23:0] data bus for BT.1120 mode. Bits [19-10] (CbCr) and [9-0] (Y) are used in 20-bit mode. Bits [19-12] (CbCr) and [9-2] (Y) are used in 16-bit mode (YCbCr 4:2:2).

Figure 11-196. DISPC VP1 Data Mapping in BT.1120 Mode



VP1 supports progressive output for embedded syncs interface, when it is configured in BT.656/BT.1120 modes. For more information on timings configuration, see Section 11.3.4.1.12.7, *DISPC VP1 Timing Generator and Panel Settings*.

11.3.4.1.12.4.1 DISPC BT Mode Blanking

During the transmission of the video signal, the portion of the stream in-between active video data segments is known as the horizontal blanking interval.

Strictly speaking this entire region is the blanking interval, but this interval also includes the EAV and SAV codes. The remaining bytes of information in a digital blanking interval are filled with values corresponding to the blanking levels of the Cb, Y and Cr signals respectively, and in accordance with the standard multiplex sequence for the stream (CbYCrY..). The blanking levels are as follows:

- Cb = 80h
- Y = 10h
- Cr = 80h.

  The sequence in the BLANKING region of the data stream is therefore: *80h, 10h, 80h, 10h.....80h, 10h*

For more details on setting the blanking timing values for BT.1120 and/or BT.656 mode, see Section 11.3.4.1.12.7, *DISPC Timing Generator and Panel Settings*.

11.3.4.1.12.4.2 DISPC BT Mode EAV and SAV

The End of Active Video (EAV) and Start of Active Video (SAV) parts of the stream are timing codes. Their function can be summarized as follows:

- EAV – marks the end of the active video data within the current line and therefore also the start of the subsequent line.
- SAV – heralds the start of the active video data within the current line.

These codes are embedded within the BT.656 video data stream, thereby eliminating the need for additional timing signals (HSYNC, VSYNC) to be included as part of the interface.

Both EAV and SAV codes are comprised of a sequence of four bytes (FFh – 00h – 00h - XY). The first three bytes in the sequence constitute a fixed preamble. The fourth byte, contains information about the field being transmitted (Field 1 or Field 2 in an interlaced video signal), the state of field blanking (Vertical) and the state of line blanking (Horizontal). The bit assignment for this byte of the code is shown in Figure 11-197, with the function of each bit described inTable 11-415.

Figure 11-197. DISPC BT Mode Bit-Assignment for the Fourth Byte of EAV/SAV Codes

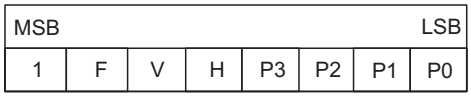


Table 11-415. DISPC BT Mode Bit Function

Bit	Symbol	Function
7	1	Always set to '1'.
6	F	Field bit. 0 – Field 1 1 – Field 2
5	V	Vertical Blanking Status bit. This bit goes High during a vertical field blanking interval, otherwise it remains Low.
4	H	Horizontal Blanking Status bit. 0 – byte is part of SAV code (i.e. stream is entering an active video data region for the current line) 1 – byte is part of EAV code (i.e. stream has entered a horizontal blanking interval - start of a new line)
3	P3	Protection bit 3
2	P2	Protection bit 2
1	P1	Protection bit 1
0	P0	Protection bit 0

The protection bits allow for detection and correction of 1-bit errors and the detection of 2-bit errors. The status of P3, P2, P1 and P0 depend on the states of bits F, V and H. This dependency is shown in Table 11-416.

Table 11-416. DISPC BT Mode Status of Protection Bits as F, V and H Vary

F	V	H	P3	P2	P1	P0
0	0	0	0	0	0	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	0
1	0	0	0	1	1	1
1	0	1	1	0	1	0
1	1	0	1	1	0	0

Table 11-416. DISPC BT Mode Status of Protection Bits as F, V and H Vary (continued)

F	V	H	P3	P2	P1	P0
1	1	1	0	0	0	1

11.3.4.1.12.5 DISPC VP1 Spatial/Temporal Dithering

The spatial/temporal dithering logic can be selected to enhance the quality of the active matrix outputs. The dithering logic is integrated after the CPR and before the TDM. The encoded pixel values are used by spatial/temporal dithering logic to display the data in a lower color depth on the display panel. The spatial/temporal dithering algorithm is based on the (x,y) pixel position and frame rate control. The dithering logic can process the pixels over one frame, two frames, or four frames. The number of frames is selected by setting the DISPC_VP1_CONTROL[31-30] SPATIALTEMPORALDITHERINGFRAMES bit field. In the case of a single frame, only spatial processing is applied. In case of multiple frames, spatial and temporal processing are applied to the pixels. The spatial/temporal dithering logic is enabled by setting the DISPC_VP1_CONTROL[7] STDITHERENABLE bit to 0x1.

NOTE:

- If the interface data bus is smaller than the pixel format size and spatial/temporal dithering is not enabled, the MSBs of the pixel color components are output on the interface data bus.
- If the interface data bus is larger than the pixel format size, by programming the pixel components replication active/inactive, the MSB is replicated to the LSB of the interface data bus.

11.3.4.1.12.6 DISPC VP1 Multiple Cycle Output Format (TDM)

The pixels are formatted on one or multiple cycles (a maximum of three cycles). On three cycles, two pixels can concatenate and send to the panel. The cycle format is selected through the DISPC_VP1_CONTROL[24-23] TDMCYCLEFORMAT bit field. The number of bits for each cycle is set in the DISPC_VP1_DATA_CYCLE_0 register for the first cycle, the DISPC_VP1_DATA_CYCLE1 register for the second cycle, and the DISPC_VP1_DATA_CYCLE2 register for the third cycle. The interface data bus width, when TDM mode is enabled (DISPC_VP1_CONTROL[20] TDMENABLE = 1), can be 8, 9, 12, or 16 bits, configurable through the DISPC_VP1_CONTROL[22-21] TDMPARALLELMODE bit field.

When the TDM is disabled (DISPC_VP1_CONTROL[20] TDMENABLE = 0), the DISPC outputs the pixels using the conventional formats: active matrix display monochrome/color. When TDM is disabled, the interface data bus width is configured through the DISPC_VP1_CONTROL[10-8] DATALINES bit field.

When using TDM mode, only up to 24-bit per pixel can be output on the interface. For higher color depth, only the upper bits are kept before converting each pixel into TDM output.

Figure 11-198 through Figure 11-201 show various examples of TDM settings in the function of pixel data formats and the interface data bus width.

Figure 11-198. DISPC VP1 TDM 8-Bit Interface Settings

	24-bpp		
	1st cycle	2nd cycle	3rd cycle
Data[7]	R0[7]	G0[7]	B0[7]
Data[6]	R0[6]	G0[6]	B0[6]
Data[5]	R0[5]	G0[5]	B0[5]
Data[4]	R0[4]	G0[4]	B0[4]
Data[3]	R0[3]	G0[3]	B0[3]
Data[2]	R0[2]	G0[2]	B0[2]
Data[1]	R0[1]	G0[1]	B0[1]
Data[0]	R0[0]	G0[0]	B0[0]

DISPC_VP1_CONTROL.TDMCYCLEFORMAT = 0x2
DISPC_VP1_DATA_CYCLE0 = 0x00000008
DISPC_VP1_DATA_CYCLE1 = 0x00000008
DISPC_VP1_DATA_CYCLE2 = 0x00000008

	18-bpp		
	1st cycle	2nd cycle	3rd cycle
Data[7]	R0[5]	G0[3]	x
Data[6]	R0[4]	G0[2]	x
Data[5]	R0[3]	G0[1]	x
Data[4]	R0[2]	G0[0]	x
Data[3]	R0[1]	B0[5]	x
Data[2]	R0[0]	B0[4]	x
Data[1]	G0[5]	B0[3]	B0[1]
Data[0]	G0[4]	B0[2]	B0[0]

DISPC_VP1_CONTROL.TDMCycleFormat = 0x2
DISPC_VP1_DATA_CYCLE0 = 0x00000008
DISPC_VP1_DATA_CYCLE1 = 0x00000008
DISPC_VP1_DATA_CYCLE2 = 0x00000002

	16-bpp	
	1st cycle	2nd cycle
Data[7]	R0[4]	G0[2]
Data[6]	R0[3]	G0[1]
Data[5]	R0[2]	G0[0]
Data[4]	R0[1]	B0[4]
Data[3]	R0[0]	B0[3]
Data[2]	G0[5]	B0[2]
Data[1]	G0[4]	B0[1]
Data[0]	G0[3]	B0[0]

DISPC_VP1_CONTROL.TDMCYCLEFORMAT = 0x1
DISPC_VP1_DATA_CYCLE0 = 0x00000008
DISPC_VP1_DATA_CYCLE1 = 0x00000008

	12-bpp	
	1st cycle	2nd cycle
Data[7]	R0[3]	x
Data[6]	R0[2]	x
Data[5]	R0[1]	x
Data[4]	R0[0]	x
Data[3]	G0[3]	B0[3]
Data[2]	G0[2]	B0[2]
Data[1]	G0[1]	B0[1]
Data[0]	G0[0]	B0[0]

DISPC_VP1_CONTROL.TDMCYCLEFORMAT = 0x1
DISPC_VP1_DATA_CYCLE0 = 0x00000008
DISPC_VP1_DATA_CYCLE1 = 0x00000004

dispc-057

Figure 11-199. DISPC VP1 TDM 9-Bit Interface Settings

	24-bpp		
	1st cycle	2nd cycle	3rd cycle
Data[8]	R0[7]	G0[6]	x
Data[7]	R0[6]	G0[5]	x
Data[6]	R0[5]	G0[4]	x
Data[5]	R0[4]	G0[3]	B0[5]
Data[4]	R0[3]	G0[2]	B0[4]
Data[3]	R0[2]	G0[1]	B0[3]
Data[2]	R0[1]	G0[0]	B0[2]
Data[1]	R0[0]	B0[7]	B0[1]
Data[0]	G0[7]	B0[6]	B0[0]

DISPC_VP1_CONTROL.TDMCycleFormat = 0x2
DISPC_VP1_DATA_CYCLE0 = 0x00000009
DISPC_VP1_DATA_CYCLE1 = 0x00000009
DISPC_VP1_DATA_CYCLE2 = 0x00000006

	18-bpp	
	1st cycle	2nd cycle
Data[8]	R0[5]	G0[2]
Data[7]	R0[4]	G0[1]
Data[6]	R0[3]	G0[0]
Data[5]	R0[2]	B0[5]
Data[4]	R0[1]	B0[4]
Data[3]	R0[0]	B0[3]
Data[2]	G0[5]	B0[2]
Data[1]	G0[4]	B0[1]
Data[0]	G0[3]	B0[0]

DISPC_VP1_CONTROL.TDMCYCLEFORMAT = 0x1
DISPC_VP1_DATA_CYCLE0 = 0x00000009
DISPC_VP1_DATA_CYCLE1 = 0x00000009

	16-bpp	
	1st cycle	2nd cycle
Data[8]	R0[4]	x
Data[7]	R0[3]	x
Data[6]	R0[2]	G0[1]
Data[5]	R0[1]	G0[0]
Data[4]	R0[0]	B0[4]
Data[3]	G0[5]	B0[3]
Data[2]	G0[4]	B0[2]
Data[1]	G0[3]	B0[1]
Data[0]	G0[2]	B0[0]

DISPC_VP1_CONTROL.TDMCYCLEFORMAT = 0x1
DISPC_VP1_DATA_CYCLE0 = 0x00000009
DISPC_VP1_DATA_CYCLE1 = 0x00000007

	12-bpp	
	1st cycle	2nd cycle
Data[8]	R0[3]	x
Data[7]	R0[2]	x
Data[6]	R0[1]	x
Data[5]	R0[0]	x
Data[4]	G0[3]	x
Data[3]	G0[2]	x
Data[2]	G0[1]	B0[2]
Data[1]	G0[0]	B0[1]
Data[0]	B0[3]	B0[0]

DISPC_VP1_CONTROL.TDMCYCLEFORMAT = 0x1
DISPC_VP1_DATA_CYCLE0 = 0x00000009
DISPC_VP1_DATA_CYCLE1 = 0x00000003

dispc-058

Figure 11-200. DISPC VP1 TDM 12-Bit Interface Settings

	24-bpp	
	1st cycle	2nd cycle
Data[11]	R0[7]	G0[3]
Data[10]	R0[6]	G0[2]
Data[9]	R0[5]	G0[1]
Data[8]	R0[4]	G0[0]
Data[7]	R0[3]	B0[7]
Data[6]	R0[2]	B0[6]
Data[5]	R0[1]	B0[5]
Data[4]	R0[0]	B0[4]
Data[3]	G0[7]	B0[3]
Data[2]	G0[6]	B0[2]
Data[1]	G0[5]	B0[1]
Data[0]	G0[4]	B0[0]

DISPC_VP1_CONTROL.TDMCYCLEFORMAT = 0x1
DISPC_VP1_DATA_CYCLE0 = 0x0000000C
DISPC_VP1_DATA_CYCLE1 = 0x0000000C

	18-bpp		
	1st cycle	2nd cycle	3rd cycle
Data[11]	R0[5]	B0[5]	G1[5]
Data[10]	R0[4]	B0[4]	G1[4]
Data[9]	R0[3]	B0[3]	G1[3]
Data[8]	R0[2]	B0[2]	G1[2]
Data[7]	R0[1]	B0[1]	G1[1]
Data[6]	R0[0]	B0[0]	G1[0]
Data[5]	G0[5]	R1[5]	B1[5]
Data[4]	G0[4]	R1[4]	B1[4]
Data[3]	G0[3]	R1[3]	B1[3]
Data[2]	G0[2]	R1[2]	B1[2]
Data[1]	G0[1]	R1[1]	B1[1]
Data[0]	G0[0]	R1[0]	B1[0]

DISPC_VP1_CONTROL.TDMCYCLEFORMAT = 0x3
DISPC_VP1_DATA_CYCLE0 = 0x0000000C
DISPC_VP1_DATA_CYCLE1 = 0x00060606
DISPC_VP1_DATA_CYCLE2 = 0x000C0000

	16-bpp	
	1st cycle	2nd cycle
Data[11]	R0[4]	x
Data[10]	R0[3]	x
Data[9]	R0[2]	x
Data[8]	R0[1]	x
Data[7]	R0[0]	x
Data[6]	G0[5]	x
Data[5]	G0[4]	x
Data[4]	G0[3]	x
Data[3]	G0[2]	B0[3]
Data[2]	G0[1]	B0[2]
Data[1]	G0[0]	B0[1]
Data[0]	B0[4]	B0[0]

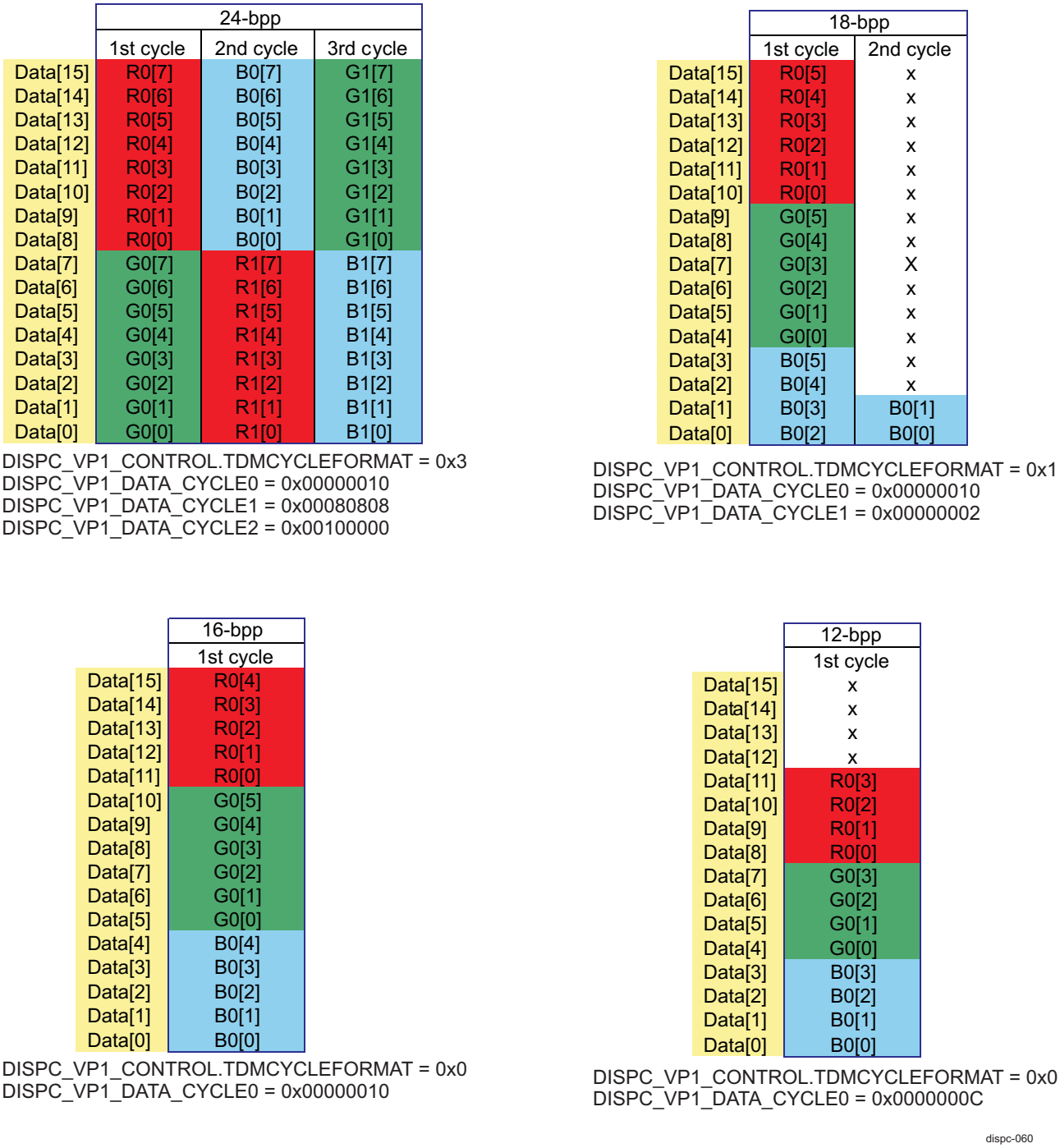
DISPC_VP1_CONTROL.TDMCYCLEFORMAT = 0x1
DISPC_VP1_DATA_CYCLE0 = 0x0000000C
DISPC_VP1_DATA_CYCLE1 = 0x00000004

	12-bpp
	1st cycle
Data[11]	R0[3]
Data[10]	R0[2]
Data[9]	R0[1]
Data[8]	R0[0]
Data[7]	G0[3]
Data[6]	G0[2]
Data[5]	G0[1]
Data[4]	G0[0]
Data[3]	B0[3]
Data[2]	B0[2]
Data[1]	B0[1]
Data[0]	B0[0]

DISPC_VP1_CONTROL.TDMCYCLEFORMAT = 0x0
DISPC_VP1_DATA_CYCLE0 = 0x0000000C

dispc-059

Figure 11-201. DISPC VP1 TDM 16-Bit Interface Settings



11.3.4.1.12.7 DISPC VP1 Timing Generator and Panel Settings

The VP1 output has a dedicated timing generator supporting progressive and interlaced mode.

The size of the display panel is defined by:

- Number of lines, DISPC_VP1_SIZE_SCREEN[27-16] LPP bit field, with a value from 1 to 4096
- Number of pixels per line, DISPC_VP1_SIZE_SCREEN[11-0] PPL bit field, with a value from 1 to 4096

Standard HSYNC/VSYNC timing generation are programmable for VP1:

- Horizontal front porch is set in the DISPC_VP1_TIMING_H[19-8] HFP bit field.
- Horizontal back porch is set in the DISPC_VP1_TIMING_H[31-20] HBP bit field.
- Horizontal synchronization pulse width is set in the DISPC_VP1_TIMING_H[7-0] HSW bit field.
- Vertical front porch is set in the DISPC_VP1_TIMING_V[19-8] VFP bit field.
- Vertical back porch is set in the DISPC_VP1_TIMING_V[31-20] VBP bit field.
- Vertical synchronization pulse width is set in the DISPC_VP1_TIMING_V[7-0] VSW bit field.

When the output is in BT.1120 or BT.656 mode, the following timing constants are mapped onto the DISPC_VP1_TIMING_H and DISPC_VP1_TIMING_V registers:

- Progressive mode:
  - Horizontal blanking (12 bits) is set in the {DISPC_VP1_TIMING_V[3-0] VSW, DISPC_VP1_TIMING_H[7-0] HSW} bit fields; (up to 2048 bytes of horizontal blanking supported).
  - Vertical frame blanking No 1 is set in the DISPC_VP1_TIMING_V[19-8] VFP bit field.
  - Vertical frame blanking No 2 is set in the DISPC_VP1_TIMING_V[31-20] VBP bit field.
  - Number of lines is set in the DISPC_VP1_SIZE_SCREEN[27-16] LPP bit field
  - Number of pixels per line is set in the DISPC_VP1_SIZE_SCREEN[11-0] PPL bit field.
- Interlaced mode:
  - Horizontal blanking (12 bits) is set in the {DISPC_VP1_TIMING_V[3-0] VSW, DISPC_VP1_TIMING_H[7-0] HSW} bit fields; (up to 2048 bytes of horizontal blanking supported).
  - Vertical field blanking No 1 for Even Field is set in the DISPC_VP1_TIMING_H[19-8] HFP bit field.
  - Vertical field blanking No 2 for Even Field is set in the DISPC_VP1_TIMING_H[31-20] HBP bit field.
  - Vertical field blanking No 1 for Odd Field is set in the DISPC_VP1_TIMING_V[19-8] VFP bit field.
  - Vertical field blanking No 2 for Odd Field is set in the DISPC_VP1_TIMING_V[31-20] VBP bit field.
  - Number of lines per field (even) is set in the DISPC_VP1_SIZE_SCREEN[27-16] LPP bit field.
  - Delta number of odd field compared to even field (in a single line) is set in the DISPC_VP1_SIZE_SCREEN[15-14] DELTA_LPP bit field.
  - Number of pixels per line is set in the DISPC_VP1_SIZE_SCREEN[11-0] PPL bit field.

NOTE: The DELTA_LPP field only controls the output channel and not the size of the field fetched from the frame buffer in memory. This field shall be set to zero for YUV420 format.

Horizontal/vertical synchronization and output enable signals polarity are programmable by setting the DISPC_VP1_POL_FREQ[12] IVS, DISPC_VP1_POL_FREQ[13] IHS, and DISPC_VP1_POL_FREQ[15] IEO bits. These signals can be gated by setting the DISPC_VP1_CONFIG[7] VSYNCGATED and DISPC_VP1_CONFIG[6] HSYNCGATED bits. In addition, the alignment between VSYNC and HSYNC signals can be programmed via the DISPC_VP1_POL_FREQ[18] ALIGN bit.

The latch of data can be driven on the rising or falling edge of the pixel clock by setting the DISPC_VP1_POL_FREQ[14] IPC register bit. The drive of the SYNC and VSYNC signals in the function of the pixel clock is done by setting the DISPC_VP1_POL_FREQ[16] RF bit.

The timing generation of FID signal is defined by the DISPC_VP1_CONFIG[22] OUTPUTMODEENABLE bitfield. The selection can be changed only if the VP output is disabled.

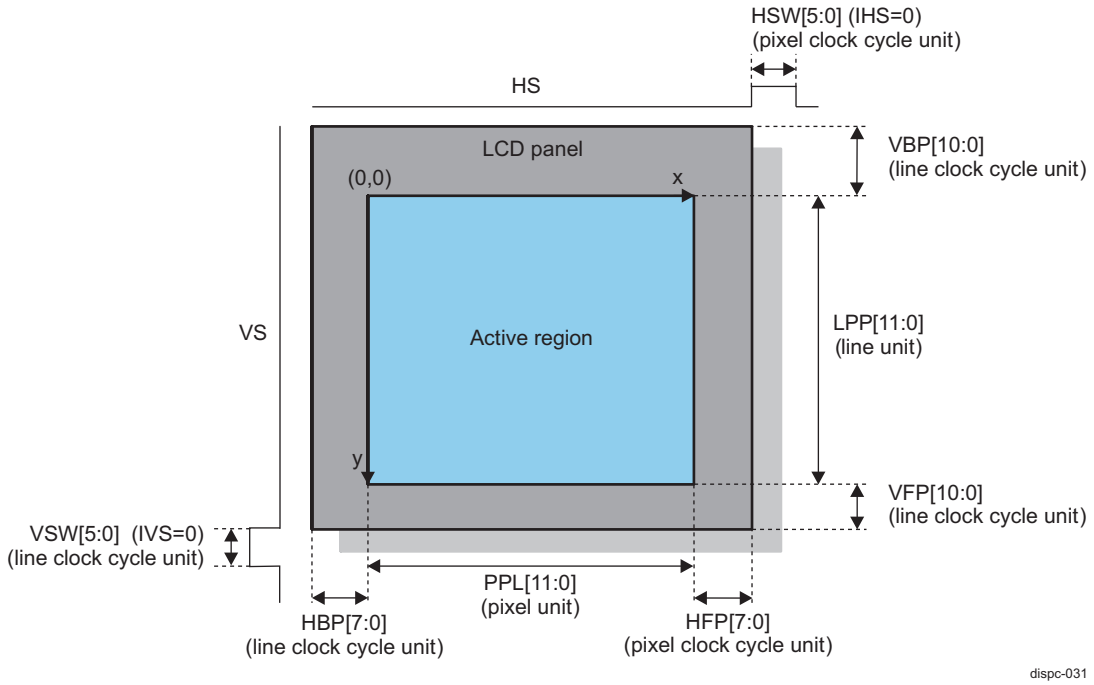
- 0x0 (default): Progressive mode (FID signal is inactive: low level)
- 0x1: Interlaced mode (FID signal transition from low to high and from high to low to indicate on the VP interface the field polarity and internally the FID to be used for the registers with dual value: one for odd field and another one for even field). The FID signal toggles on the rising edge of the VSW pulse.

When in interlaced mode, the DISPC_VP1_CONFIG[23] FIDFIRST bit indicates which field is output first:

- 0x0: Even field first (FID = 0)
- 0x1: Odd field first (FID = 1)

Figure 11-202 shows the timing values description.

Figure 11-202. DISPC VP1 Timing Values (Display Screen)

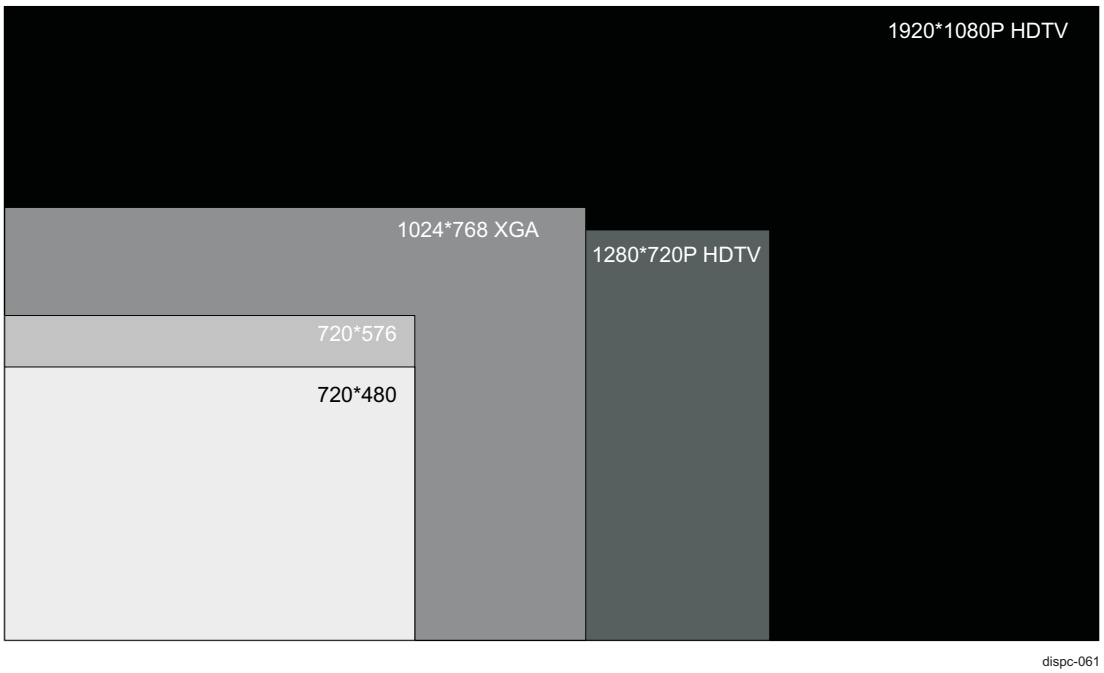


The pixel clock can be gated by setting the DISPC_VP1_CONFIG[5] PIXELCLOCKGATED bit to 0x1.

11.3.4.1.12.8 DISPC VP1 Configuration for TV Support

Figure 11-203 shows the TV formats supported.

Figure 11-203. DISPC Example TV Timing Formats



The size of a frame (field, if interlaced mode) is defined by the fields:

- Number of lines, DISPC_VP1_SIZE_SCREEN[27-16] LPP bit field, with a value from 1 to 4096

- Number of pixels per line, DISPC_VP1_SIZE_SCREEN[11-0] PPL bit field, with a value from 1 to 4096
- Delta size between odd/even field, DISPC_VP1_SIZE_SCREEN[15-14] DELTA_LPP bit field. This bit field controls only the output channel and not the size of the data field fetched from the frame buffer in memory.

The hold time of the pixels on the data bus is determined in clock cycles by the DISPC_VP1_CONTROL[16-14] HT bit field. The default value at reset time is 0x0 (one cycle).

- Table 11-417 indicates the DISPC_VP1_SIZE_SCREEN register values for PPL and LPP for each HD standard.

Table 11-417. DISPC VP1 PPL and LLP Value for HD Standard

Standards		Active Pixels/Line	Active Lines	Digital Clock DISPC_VP1_CONTROL [16-14] HT = 0x0	DISPC_VP1_SIZE_SCREEN Register Value
HDTV	720p	1280	720	74.25/74.125 MHz (60/59.99..frames/s)	0x02CF 04FF
	1080i	1920	540	74.25/74.125 MHz (60/59.99..frames/s)	0x021B 077F
	1080p	1920	1080	148.5/148.25 MHz (60/59.99..frames/s)	0x0437 077F

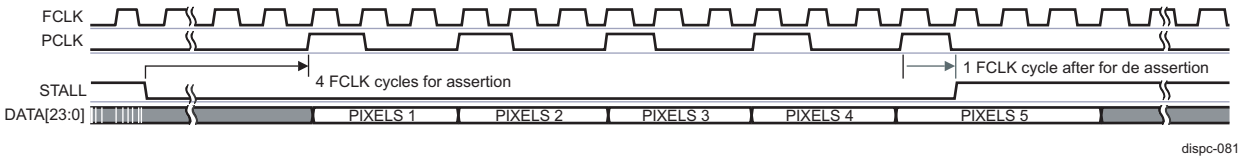
11.3.4.1.13 DISPC Stall Mode

Stall mode is used to indicate when the DISPC must stop sending data on the video port output interface to avoid an overflow on the interface level. Stall mode is available when the RFBI interfaces are used. The RFBI module asserts the stall signal to stop data output by the DISPC. It is deasserted by RFBI to indicate when new data must be output by the DISPC. Figure 11-204 shows the RFBI data stall mode activated.

Stall mode is selected by setting the DISPC_VP1_CONTROL[11] STALLMODE bit.

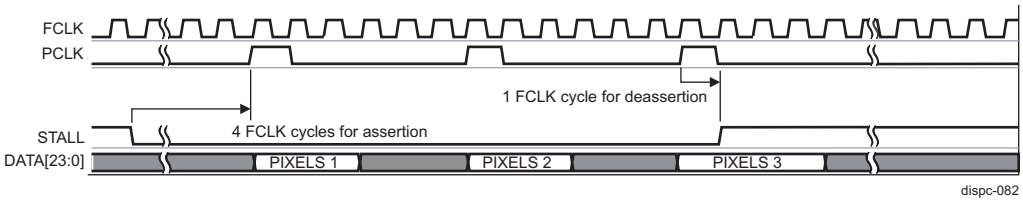
NOTE: The DISPC_VP1_CONTROL[5] GOBIT bit must not be set, but the DISPC configuration (DMA engine, VID1 pipeline) must be set before enabling the VP1 output by setting the DISPC_VP1_CONTROL[0] VPENABLE bit. In stall mode the VPENABLE bit initates the transfer.

Figure 11-204. DISPC RFBI Data Stall Signal Diagram



To avoid an underflow of the DMA buffer, the DMA buffer handshake must be enabled by setting the DISPC_VP1_CONFIG[16] BUFFERHANDCHECK bit. The fullness of the buffers associated to the VID1 pipeline used for the VP1 output is checked before providing data to the pipeline when the STALL signal is inactive. It prevents emptying the DMA buffer when the RFBI module requests data and there is not enough data in the DMA buffer of the DISPC. This feature must be enabled only when stall mode is used (the STALLMODE bit set to 0x1). When the DMA buffer handshake is activated, the pixel transfer to the RFBI module, outside a STALL period, can be stopped and restarted when there are enough data in the DMA buffer. The DMA buffer handshake ensures that the underflow does not occur for the VID1 pipeline associated with the VP1 output when RFBI interface is used. The rate of the data transfer to the RFBI is, then, fully dependent on the state of the DMA buffer. Figure 11-205 shows the RFBI data stall with FIFO handcheck mode activated.

Figure 11-205. DISPC RFBI Data Stall Signal Diagram with Handcheck



One STALL signal is generated by the RFBI. The RFBI module asserts the STALL signal when no more data must be output by the DISPC. STALL is deasserted to indicate when new data must be output by the DISPC.

11.3.4.1.14 DISPC Shadow Registers

Some DISPC registers are termed *shadow registers*. The shadow registers allow the software to modify them at any time, without direct effect on the DISPC hardware configuration. When all the values for a given configuration are written into the shadow registers, software must set only 1 bit to validate the configuration. When the hardware reaches the end of the current frame and sees that the bit has been set by software, the new configuration is now the configuration used by the hardware.

The DISPC_VP1_CONTROL[5] GOBIT bit enables the hardware to use the new configuration, for all shadow registers of the VID1 pipeline connected to the VP1 output.

11.3.4.2 RFBI Functional Description

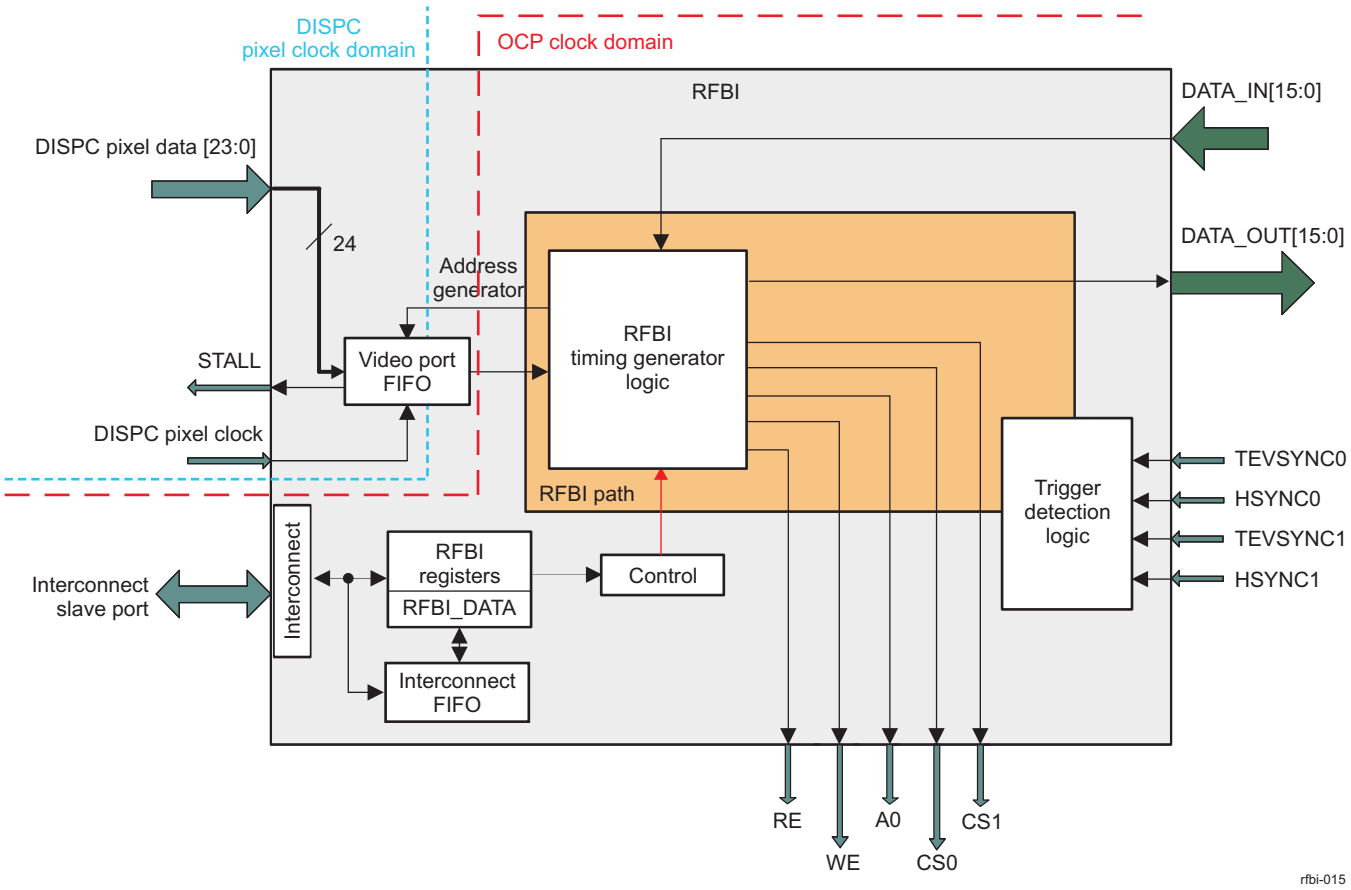
11.3.4.2.1 RFBI Block Diagram

The RFBI can capture the output pixel from the DISPC, or system memory, and send the data to the RFB in the LCD panel. The user application configures the RFBI, sends commands, reads data, and configures the DISPC to send data fetched from the system memory by the DISPC DMA engine. The commands/data are sent using an 8-, 9-, 12- or 16-bit parallel interface.

The DISPC can be configured to send the data in 12-, 16-, 18-, or 24-bpp format. In the RFBI video port FIFO, the encoded pixel values are in an LSB alignment independent of the endianness in system memory.

Figure 11-206 is an overview of the RFBI architecture.

Figure 11-206. RFBI Architecture Overview



11.3.4.2.2 RFBI Clock Configuration

For more information about the RFBI clocks, see Section 11.3.3, *DSS Integration*.

11.3.4.2.3 RFBI Software Reset

To perform a software reset, set the RFBI_SYSCONFIG[1] SOFTRESET bit to 1. The RFBI_SYSSTATUS[0] RESETDONE bit indicates that the software reset is complete when its value is 1. When the software reset completes, the RFBI_SYSCONFIG[1] SOFTRESET bit is automatically reset. Software must ensure that the software reset completes before performing any RFBI operation.

11.3.4.2.4 RFBI Interrupt Requests

The RFBI does not generate interrupts, but DISPC interrupts must be properly configured, when DISPC is connected to the RFBI.

For more information on DISPC interrupts, see Section 11.3.4.1.5,*DISPC Interrupt Requests*.

11.3.4.2.5 RFBI DMA Requests

Table 11-418 lists event flags and their mask that can cause RFBI DMA requests.

Table 11-418. RFBI DMA Requests Events

DMA Request Name	DMA Request Mask	Description
RFBI_DMA_REQ	RFBI_CONTROL[7] DISABLE_DMA_REQ	DMA request used to transfer data from system memory (via SoC DMA controller) to the RFBI interconnect FIFO

11.3.4.2.6 RFBI Video Port FIFO

11.3.4.2.6.1 RFBI Video Port Description

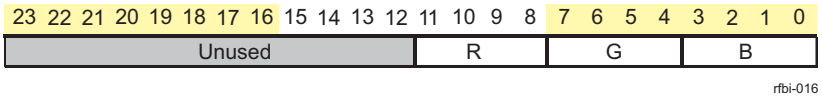
The input video port FIFO receives data from the DISPC at the pixel clock. The data in the video port FIFO are read by the RFBI and sent to the LCD panel. The video port FIFO is 24-bits wide and each pixel in 12-, 16-, 18-, and 24-bpp format is stored in the video port FIFO using one 24-bit value aligned on the 24-bit LSB. Section 11.3.4.2.9, *RFBI Output Parallel Modes*, provides examples of various output configurations based on the interface width (up to 16 bits) and pixel format output (up to 24 bits). Setting the RFBI_CONTROL[1] RFBIMODE bit to 0 directs the SoC ARMSS to send commands, parameters, and data from the input video port FIFO.

11.3.4.2.6.2 RFBI Input Formats

The encoded pixel formats supported at the RFBI input interface connected to the DISPC are: RGB12-444, RGB16-565, RGB18-666, and RGB24-888. This input data width is controlled by the RFBI_CONFIG__0/RFBI_CONFIG__1[6-5] DATATYPE bit field.

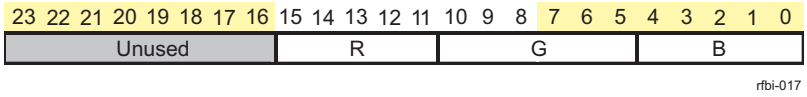
The following graphics describe the bit representation of each format in the video port FIFO. The output of the DISPC is aligned on the LSB of the interface. The pixels are then formatted in accordance with the configuration of the output interfaces (multiple cycles). For more information, see Section 11.3.4.2.9.1, *Cycle Mode Selection*.

- RGB12-444 format:



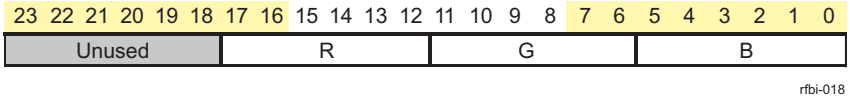
rfbi-016

- RGB16-565 format:



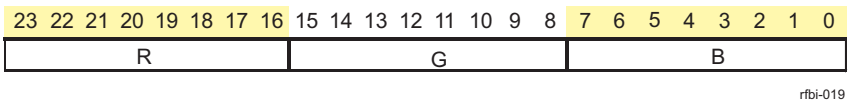
rfbi-017

- RGB18-666 format:



rfbi-018

- RGB24-888 format:



rfbi-019

NOTE: These pixel formats are also supported when data are provided from the slave port interconnect (for writing parameters).

11.3.4.2.6.3 RFBI Stall Mechanism

The internal STALL signal is generated by the RFBI video port FIFO to indicate when the DISPC must stop sending data on DISPC VP1 interface. This stall signal is used for DISPC VP1 output only when the DISPC is configured in stall mode. The stall mode is activated by setting the DISPC_VP1_CONTROL[11] STALLMODE bit.

NOTE: When the DISPC is configured in stall mode, the minimum transfer size is 1 byte.

The STALL signal allows the RFBI to reformat the data.

STALL is asserted when at least one of the following cases occurs:

- Default status when there is no data to capture from the DISPC
- High FIFO threshold reached
- End of transfer (number of data to output)
- RFBI reset
- RFBI_CONTROL[0] ENABLE bit reset to 0x0

STALL is deasserted when the RFBI_CONTROL[0] ENABLE bit is set to 0x1 and at least one of the following cases occurs:

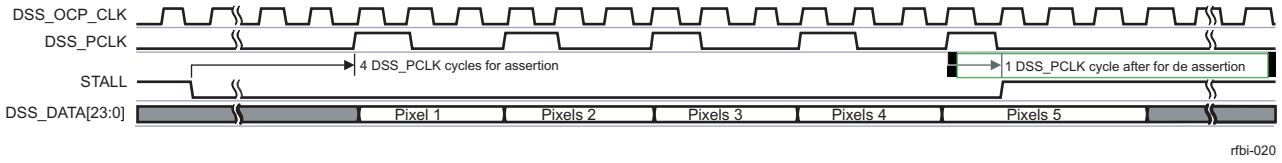
- Low FIFO threshold reached
- External TE occurs and the RFBI_CONFIG__0/RFBI_CONFIG__1[3-2] TRIGGERMODE bit field is set to 0x1 for automatic external trigger (start of the transfer, the FIFO pointers are reset, the FIFO is empty).
- RFBI_CONTROL[4] ITE bit set to 0x1 by users (start of transfer, the FIFO pointers are reset, the FIFO is empty)

The RFBI asserts the STALL signal to stop data output by the DISPC. It is deasserted to indicate when new data must be output by the DISPC.

11.3.4.2.6.3.1 RFBI Data Stall Diagram without DISPC DMA Buffer Handshake

Figure 11-207 shows the RFBI data stall diagram when the DSS_OCP_CLK clock is 3 times faster than the pixel clock.

Figure 11-207. RFBI Data Stall Signal Diagram

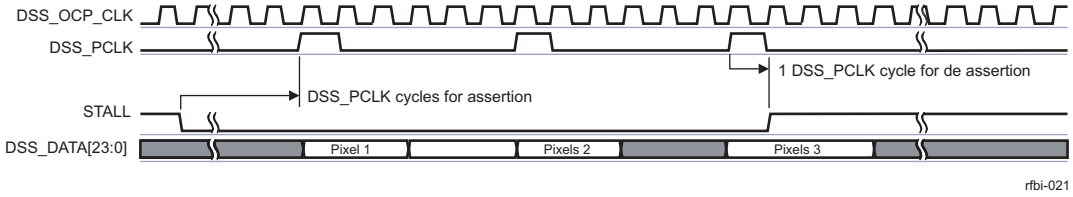


11.3.4.2.6.3.2 RFBI Data Stall Diagram with DISPC DMA Buffer Handshake

To avoid underflow of the DISPC DMA buffer, the DMA buffer handshake feature can be enabled by setting the DISPC_VP1_CONFIG[16] BUFFERHANDCHECK bit to 1. The fullness of the FIFOs associated with the VID1 pipeline used for the DISPC VP1 output is checked when the STALL signal is inactive before providing data to the pipeline. This prevents emptying the FIFO when the RFBI requests data and there is not enough data in the DISPC DMA buffer. This feature must be enabled only when the stall mode is used (DISPC_VP1_CONTROL[11] STALLMODE bit set to 1).

When the DMA buffer handshake feature is activated, the pixel transfer to the RFBI during STALL inactivity period can be stopped (no PCLK pulse) and restarted when there is enough data in the DMA buffer. The DMA buffer handshake ensures that underflow cannot occur for the VID1 pipeline associated with the DISPC VP1 output in stall mode. Figure 11-208 shows the RFBI data stall with the DMA buffer handshake mode activated, and the DSS_OCP_CLK clock is 3 times faster than the pixel clock.

Figure 11-208. RFBI Data Stall Signal Diagram with Handshake



11.3.4.2.7 RFBI Interconnect FIFO

11.3.4.2.7.1 RFBI FIFO Description

The RFBI interconnect FIFO receives the data from DSS slave port. The data in the interconnect FIFO are read by the RFBI and sent to the LCD panel. The address of the RFBI_DATA register should be used to access the interconnect FIFO. The interconnect FIFO is 32-bits wide. The size of the interconnect FIFO is 24 words of 32 bits (that is, 24 words of RFBI_DATA). The RFBI DMA request can be used to transfer data using the SoC DMA controller from system memory to the interconnect FIFO.

11.3.4.2.7.2 RFBI Using DMA Request with Interconnect FIFO

11.3.4.2.7.2.1 RFBI Threshold for DMA Request Generation

The RFBI_CONTROL[6-5] HIGHTHRESHOLD bit field defines the threshold to be used for the generation of the DMA request to receive data into the interconnect FIFO (24 × 32 FIFO depth) through the address of the register RFBI_DATA. The SoC DMA controller configuration regarding the size of the burst depends on the value of the RFBI_CONTROL[6-5] HIGHTHRESHOLD bit field. The supported values are 4 × 32, 8 × 32, and 16 × 32. For example, when 4 words of 32 bits are set for the threshold, the SoC DMA controller sends a burst of 4 × 32 bits only. The SoC DMA controller receives the DMA request and is in charge of providing the correct number of bytes.

11.3.4.2.7.2.2 RFBI Disabling DMA Request

If the RFBI_CONTROL[7] DISABLE_DMA_REQ bit is reset, the DMA request is generated when there is enough room in the interconnect FIFO to accept the full burst. In case the RFBI receives writes slave port requests to the RFBI_DATA location when the interconnect FIFO is full, the request is not accepted. The RFBI waits for a free entry in the interconnect FIFO to accept the slave port request.

If the RFBI_CONTROL[7] DISABLE_DMA_REQ bit is set, the DMA request is not generated and the threshold value is ignored.

NOTE: Software can access the RFBI_DATA location without using the DMA request and without programming the high threshold value.

11.3.4.2.7.2.3 RFBI Smart DMA Request Mode

If the RFBI_CONTROL[8] SMART_DMA_REQ bit is reset (smart DMA mode disabled), the DMA request is asserted and deasserted depending on the interconnect FIFO space, even if MIdlereq is high in smart-idle mode and the entire burst gets error responses from the RFBI. The RFBI waits for a free entry in the interconnect FIFO to accept the burst request.

If the RFBI_CONTROL[8] SMART_DMA_REQ bit is set (smart DMA mode enabled), the DMA request is deasserted after two DSS_OCP_CLK cycles, if it has been asserted for more than or equal to two DSS_OCP_CLK cycles and MIdlereq is high in smart-idle mode. If asserting time is less than two DSS_OCP_CLK cycles, no more burst requests are accepted even if space is available in the interconnect FIFO.

11.3.4.2.8 RFBI Timing Generator

11.3.4.2.8.1 RFBI Configuration Selection

The RFBI_CONTROL[3-2] CONFIGSELECT bit field selects the chip-select, CS0 or CS1, and corresponding configuration, 0 or 1.

The registers associated to the selected configuration are used to output the data to the LCD panel. If both CS0 and CS1 are selected, the configuration for the CS0 is used (except for the polarity of CS1 signal defined by the second configuration) and both devices connected to the CS signals are driven in parallel. In Read mode, if both CS0 and CS1 are set, only CS0 is asserted in order to read data from the device connected on the CS0. In Write mode, with both CS0 and CS1 selected, the RFBI is able to write into the two devices at the same time.

Configuration 0 (with CS0 selected) uses the following registers:

- RFBI_CONFIG__0
- RFBI_ONOFF_TIME__0
- RFBI_CYCLE_TIME__0
- RFBI_DATA_CYCLE1__0
- RFBI_DATA_CYCLE2__0
- RFBI_DATA_CYCLE3__0

Configuration 1 (with CS1 selected) uses the following registers:

- RFBI_CONFIG__1
- RFBI_ONOFF_TIME__1
- RFBI_CYCLE_TIME__1
- RFBI_DATA_CYCLE1__1
- RFBI_DATA_CYCLE2__1
- RFBI_DATA_CYCLE3__1

11.3.4.2.8.2 RFBI Read/Write

Depending on the status of A0, WE, and RE signals (asserted or deasserted), the commands and display/parameter data are written to the panel (handled by the state-machine for the commands/parameter data and stored in memory for the display data), or the display data/status values are read from the LCD panel (status and display data in the LCD panel memory). The polarity of A0 (RFBI_A0 signal), WE (RFBI_WEn signal), RE (RFBI_REn signal), and CS (RFBI_CS0/CS1 signals) is programmable.

Table 11-419 describes the read/write function.

Table 11-419. RFBI Read/Write Function Description

A0 (RFBI_A0)	WE (RFBI_WEn)	RE (RFBI_REn)	Function Description
1	0	1	Display data write, parameter data write
1	1	0	Display data read
0	1	0	Status read
0	0	1	Command data write

A minimum of CS cycle time, as defined in Table 11-420, is required to keep the RFBI_CS0/CS1 signals asserted between write transfers of multiple pixels.

Table 11-420 lists the minimum cycle time for RFBI_CS0/CS1, depending on the source of pixels (DISPC or RFBI slave port) and the cycle format (1pixel/cycle, 1pixel/2cycles, or 1pixel/3cycles).

Table 11-420. RFBI Minimum Cycle Time for CS/WE Always Asserted

RFBI Pixel Source	RFBI_CONFIG__0/ RFBI_CONFIG__1 [10-9] CYCLEFORMAT	RFBI_CONFIG__0/ RFBI_CONFIG__1 [8-7] PORTFORMAT	Minimum Cycle Time (in number of DSS_OCP_CLK cycles)
Slave port interconnect	1 pixel/cycle	1 pixel	5
	1 pixel/2 cycles	1 pixel	4
	1 pixel/3 cycles	1 pixel	4
	1 pixel/cycle	2 pixels	4
	1 pixel/2 cycles	2 pixels	4
	1 pixel/3 cycles	2 pixels	4
DISPC video port output	1 pixel/cycle	N/A	4
	1 pixel/2 cycles	N/A	3
	1 pixel/3 cycles	N/A	3
	2 pixel/3 cycles	N/A	6

11.3.4.2.8.3 RFBI State-Machine

The RFBI_A0, RFBI_REn, and RFBI_WEn signals are asserted and deasserted based on the register accessed (RFBI_CMD, RFBI_PARAM, RFBI_DATA, RFBI_READ, or RFBI_STATUS). When the RFBI_SYSSTATUS[8] BUSY bit is set by hardware, any access to the registers is stalled, except for the RFBI_DATA register.

The RFBI_SYSSTATUS[9] BUSYRFBIDATA bit indicates whether there are still pending data in the interconnect FIFO associated with the RFBI_DATA register only.

- Command register, RFBI_CMD

  Write one command at a time by writing through the slave port interconnect into the RFBI_CMD register. If the previous command is not processed, the RFBI_SYSSTATUS[8] BUSY bit is set by hardware and access to writing a new command is stalled.

- Parameter register, RFBI_PARAM

  Write one parameter at a time by writing in the RFBI_PARAM register.

  If the previous parameter is not processed, the RFBI_SYSSTATUS[8] BUSY bit is set by hardware and access to writing a new parameter is stalled.

- Data register, RFBI_DATA

  Write one or two pixels at a time by writing in the RFBI_DATA register.

  The pixels are formatted based on the specified cycle format. If two pixels are written into the 32-data register, the RFBI_CONFIG__0/RFBI_CONFIG__1[8-7] PORTFORMAT bit field indicates the number of pixels for each slave port access to the register and the order of the pixels.

  If the previous data are not processed, the RFBI_SYSSTATUS[8] BUSY bit is set by hardware and any access for writing new data is stalled. When the RFBI_SYSSTATUS[8] BUSY bit is reset by hardware, access is not stalled.

- Read/status register, RFBI_READ/RFBI_STATUS

  Send through the command and parameter registers the correct information to receive data in the data or status register. The read data from the LCD panel is initiated by writing into the RFBI_READ or RFBI_STATUS registers. In this case, the RFBI_SYSSTATUS[8] BUSY bit is set until the data are available in the register.

  When the RFBI_SYSSTATUS[8] BUSY bit is set by hardware, the read or write access is stalled until the register is updated with a new value from the LCD panel. To avoid the stall, the software can poll the RFBI_SYSSTATUS[8] BUSY bit until it is reset by hardware. To receive the data, send the

appropriate command/parameters.

11.3.4.2.8.4 RFBI Timings

The timing registers can be accessed only when there is no transaction in progress (based on the value of the RFBI_CONTROL[3-2] CONFIGSELECT bit field). Granularity is defined using the RFBI_CONFIG__0/RFBI_CONFIG__1[4] TIMEGRANULARITY bit. This feature allows the extension of programmable ranges of timing parameters for the RFBI interface. See Table 11-421, *RFBI Timings Configuration*, for the configuration values of the timing bits.

- Chip-select assertion/deassertion time (CSOnTime/CSOffTime)

  RFBI_A0 setup time to chip-select assertion is assured by the programmable chip-select assertion time from the start access time:
  RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[3-0] CSONTIME bit field

  The chip-select deassertion time from the start access time is programmable:
  RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[9-4] CSOFFTIME bit field

  CAUTION

  Configuring RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[3-0] CSONTIME = [9-4] CSOFFTIME = 0 is not supported and must be avoided. This configuration creates contention on the bus and progressively damages the LCD panel.

- Chip-select pulse width (CSPulseWidth)

  The total chip-select pulse width is the time when write cycle time or read cycle time completes and is programmable:
  RFBI_CYCLE_TIME__0/RFBI_CYCLE_TIME__1[17-12] CSPULSEWIDTH bit field

  It applies on the read-to-write, write-to-read, read-to-read, and write-to-write access based on:

  – The RFBI_CYCLE_TIME__0/RFBI_CYCLE_TIME__1[19] RRENABLE bit: Read-to-read access
  – The RFBI_CYCLE_TIME__0/RFBI_CYCLE_TIME__1[20] WWENABLE bit: Write-to-write access
  – The RFBI_CYCLE_TIME__0/RFBI_CYCLE_TIME__1[18] RWENABLE bit: Read-to-write access
  – The RFBI_CYCLE_TIME__0/RFBI_CYCLE_TIME__1[21] WRENABLE bit: Write-to-read access

  By default, it applies to any access (read-to-read, read-to-write, write-to-read, write-to-write) when the chip-select CS0 is activated by setting the RFBI_CONTROL[3-2] CONFIGSELECT bit field to 0x1.

- Access time

  Access time is the time delay between A0 assertion to data sampling before RE signal deassertion; access time is programmable:
  RFBI_CYCLE_TIME__0/RFBI_CYCLE_TIME__1[27-22] ACCESSTIME bit field

  When reading the data on the bus, the data are sampled at the end of the access time, which occurs before the end of the read off time (RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[29-24] REOFFTIME).

- Write-enable cycle time (WECycleTime)

  The total write-enable cycle time is the time when A0 becomes valid until write cycle completion; the write-enable cycle time is programmable:
  The RFBI_CYCLE_TIME__0/RFBI_CYCLE_TIME__1[5-0] WECYCLETIME bit field

- Write-enable assertion/deassertion time (WEOnTime/WEOffTime)

  The WE assertion delay time from start access time is programmable:
  RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[13-10] WEONTIME bit field

  The WE deassertion delay time from the start access time is programmable:
  RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[19-14] WEOFFTIME bit field

- Read-enable cycle time (RECycleTime)

The total read-enable cycle time is the time when A0 becomes valid until read cycle completion; the read-enable cycle time is programmable:
The RFBI_CYCLE_TIME__0/RFBI_CYCLE_TIME__1[11-6] RECYCLETIME bit field

- Read-enable assertion/deassertion time (REOnTime/REOffTime)

  The RE assertion delay time from the start access time is programmable:
  RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[23-20] REONTIME bit field

  The RE deassertion delay time from the start access time is programmable:
  RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[29-24] REOFFTIME bit field

At cycle time completion (read access or write access), all control signals (RFBI_CS0/CS1, RFBI_WEn, and RFBI_REn) are deasserted regardless of their deassertion time parameter values, if they are not deasserted already.

However, an exception to this forced deassertion exists when a pipelined request to CS is pending. Also, a control signal with deassertion time parameters equal to the cycle time parameter is not necessarily deasserted when a pipelined request to the same chip-select or different chip-select is pending. This prevents any unnecessary glitch transitions.

If no inactive cycles are required between successive accesses to the same chip-select (the RFBI_CYCLE_TIME__0/RFBI_CYCLE_TIME__1[17-12] CSPULSEWIDTH bit field = 0), and if assertion time parameters associated with the following access equal 0, the asserted control signals (RFBI_CS0/CS1, RFBI_WEn, and RFBI_REn) stay asserted. This applies only to write-to-write access combination. In case of read-to-write, read-to-read, or write-to-read sequences, the RFBI_CS0/CS1, RFBI_WEn, and RFBI_REn signals are always deasserted.

Table 11-421 lists the configuration values for each timing bit.

Table 11-421. RFBI Timings Configuration

Timing Configuration Bits	Granularity[1]	
	One	Two
RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[3-0] CSONTIME	0 to 15	0 to 30
RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[9-4] CSOFFTIME	0 to 63	0 to 126
RFBI_CYCLE_TIME__0/RFBI_CYCLE_TIME__1[17-12] CSPULSEWIDTH	0 to 63	0 to 126
RFBI_CYCLE_TIME__0/RFBI_CYCLE_TIME__1[27-22] ACCESSTIME	0 to 63	0 to 126
RFBI_CYCLE_TIME__0/RFBI_CYCLE_TIME__1[5-0] WECYCLETIME	3 to 63	4 to 126
RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[13-10] WEONTIME	1 to 15	2 to 30
RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[19-14] WEOFFTIME	2 to 63	2 to 126
RFBI_CYCLE_TIME__0/RFBI_CYCLE_TIME__1[11-6] RECYCLETIME	3 to 63	4 to 126
RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[23-20] REONTIME	1 to 15	2 to 30
RFBI_ONOFF_TIME__0/RFBI_ONOFF_TIME__1[29-24] REOFFTIME	2 to 63	2 to 126

(1) Number of DSS_OCP_CLK cycles. The granularity can be configured using the RFBI_CONFIG__0/RFBI_CONFIG__1[4] TIMEGRANULARITY bit.

11.3.4.2.9 RFBI Output Parallel Modes

11.3.4.2.9.1 RFBI Cycle Mode Selection

In the following text and graphics RFBI_CONFIG refers to RFBI_CONFIG__0 or RFBI_CONFIG__1 registers. RFBI_DATA_CYCLE1 refers to RFBI_DATA_CYCLE1__0 and RFBI_DATA_CYCLE1__1 registers. RFBI_DATA_CYCLE2 refers to RFBI_DATA_CYCLE2__0 and RFBI_DATA_CYCLE2__1 registers. RFBI_DATA_CYCLE3 refers to RFBI_DATA_CYCLE3__0 and RFBI_DATA_CYCLE3__1 registers.

The RFBI_CONFIG__0/RFBI_CONFIG__1[10-9] CYCLEFORMAT bit field defines the number of cycles to output a pixel. The number of cycles determines the number of registers used to format the data in the interconnect FIFO with the appropriate number of bits (starting from the LSB) and with the alignment on the interface. The data are formatted based on the configuration of the RFBI_DATA_CYCLE registers.

- One cycle: RFBI_CONFIG__0/RFBI_CONFIG__1[10-9] CYCLEFORMAT bit field = 0x00 and the RFBI_DATA_CYCLE1 registers
- Two cycles: RFBI_CONFIG__0/RFBI_CONFIG__1 [10-9] CYCLEFORMAT bit field = 0x01 and the RFBI_DATA_CYCLE1 and RFBI_DATA_CYCLE2 registers
- Three cycles: RFBI_CONFIG__0/RFBI_CONFIG__1[10-9] CYCLEFORMAT bit field = 0x10 and the RFBI_DATA_CYCLE1, RFBI_DATA_CYCLE2, and RFBI_DATA_CYCLE3 registers

Figure 11-209 through Figure 11-212 list the bits position of the pixel during the cycles for an 8-, 9-, and 16-bit parallel output, respectively.

Figure 11-209. RFBI 8-Bit Interface Settings

	24-bpp		
	1st cycle	2nd cycle	3rd cycle
Data[7]	R0[7]	G0[7]	B0[7]
Data[6]	R0[6]	G0[6]	B0[6]
Data[5]	R0[5]	G0[5]	B0[5]
Data[4]	R0[4]	G0[4]	B0[4]
Data[3]	R0[3]	G0[3]	B0[3]
Data[2]	R0[2]	G0[2]	B0[2]
Data[1]	R0[1]	G0[1]	B0[1]
Data[0]	R0[0]	G0[0]	B0[0]

RFBI_CONFIG.CYCLEFORMAT = 0x2
RFBI_DATA_CYCLE1 = 0x00000008
RFBI_DATA_CYCLE2 = 0x00000008
RFBI_DATA_CYCLE3 = 0x00000008

	18-bpp		
	1st cycle	2nd cycle	3rd cycle
Data[7]	R0[5]	G0[3]	x
Data[6]	R0[4]	G0[2]	x
Data[5]	R0[3]	G0[1]	x
Data[4]	R0[2]	G0[0]	x
Data[3]	R0[1]	B0[5]	x
Data[2]	R0[0]	B0[4]	x
Data[1]	G0[5]	B0[3]	B0[1]
Data[0]	G0[4]	B0[2]	B0[0]

RFBI_CONFIG.CYCLEFORMAT = 0x2
RFBI_DATA_CYCLE1 = 0x00000008
RFBI_DATA_CYCLE2 = 0x00000008
RFBI_DATA_CYCLE3 = 0x00000002

	16-bpp	
	1st cycle	2nd cycle
Data[7]	R0[4]	G0[2]
Data[6]	R0[3]	G0[1]
Data[5]	R0[2]	G0[0]
Data[4]	R0[1]	B0[4]
Data[3]	R0[0]	B0[3]
Data[2]	G0[5]	B0[2]
Data[1]	G0[4]	B0[1]
Data[0]	G0[3]	B0[0]

RFBI_CONFIG.CYCLEFORMAT = 0x1
RFBI_DATA_CYCLE1 = 0x00000008
RFBI_DATA_CYCLE2 = 0x00000008

	12-bpp	
	1st cycle	2nd cycle
Data[7]	R0[3]	x
Data[6]	R0[2]	x
Data[5]	R0[1]	x
Data[4]	R0[0]	x
Data[3]	G0[3]	B0[3]
Data[2]	G0[2]	B0[2]
Data[1]	G0[1]	B0[1]
Data[0]	G0[0]	B0[0]

RFBI_CONFIG.CYCLEFORMAT = 0x1
RFBI_DATA_CYCLE1 = 0x00000008
RFBI_DATA_CYCLE2 = 0x00000004

rfbi-026

Figure 11-210. RFBI 9-Bit Interface Settings

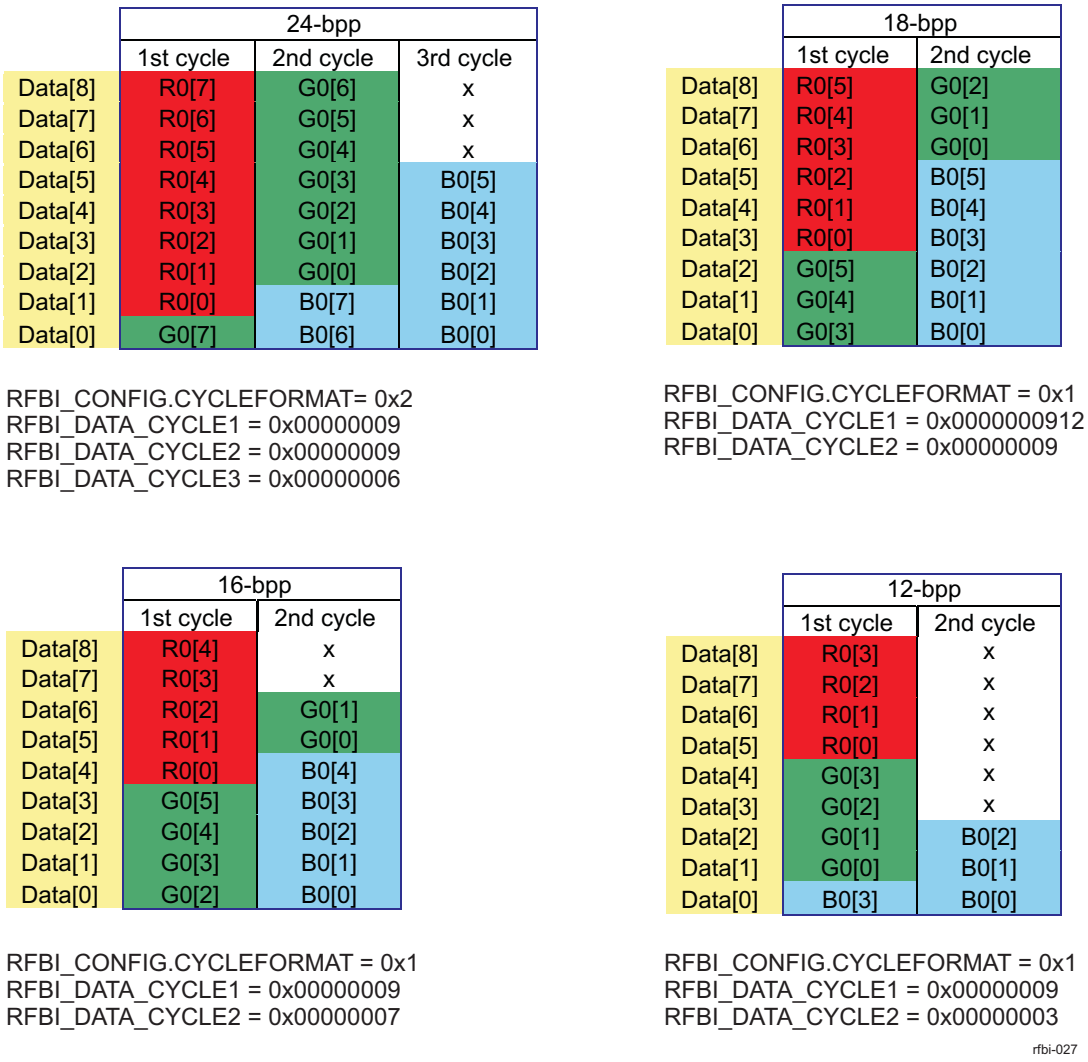


Figure 11-211. RFBI 12-Bit Interface Settings

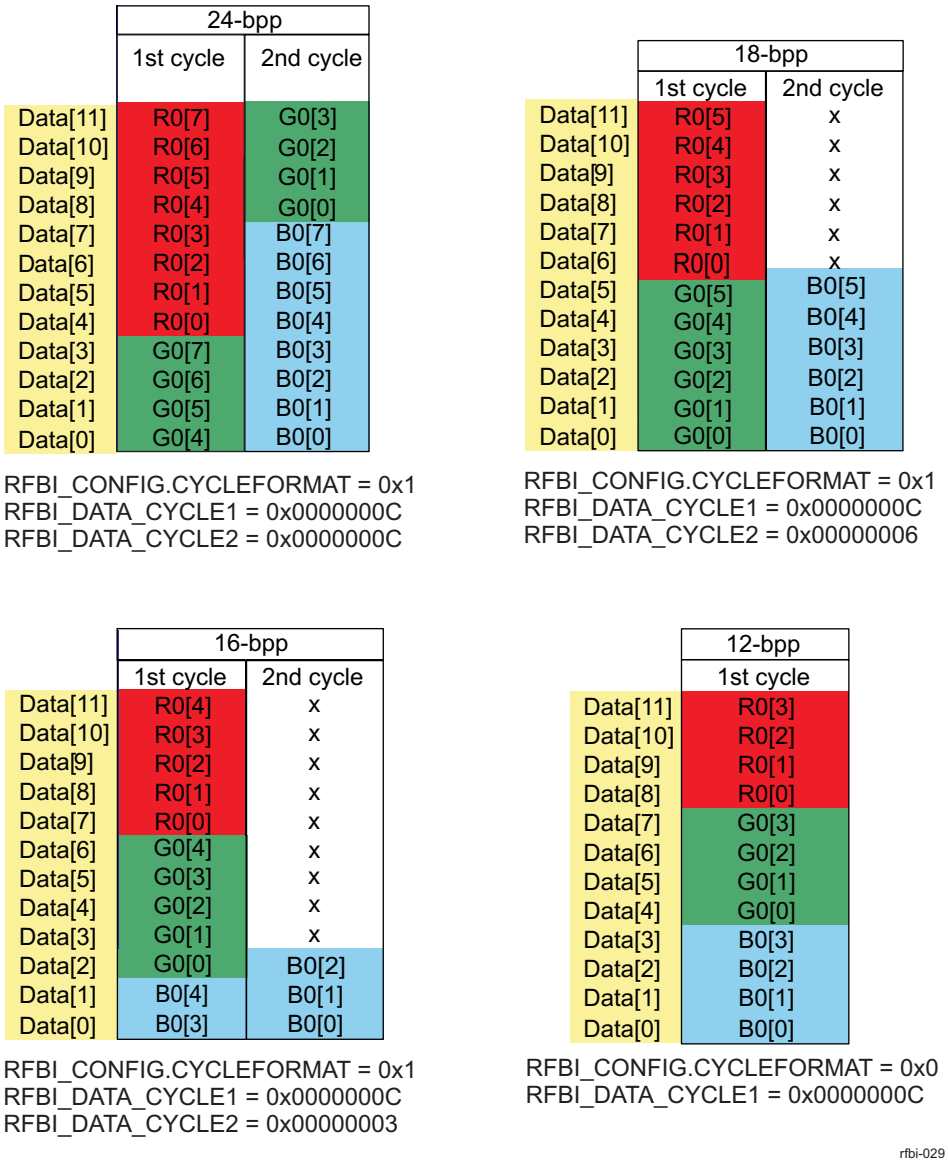
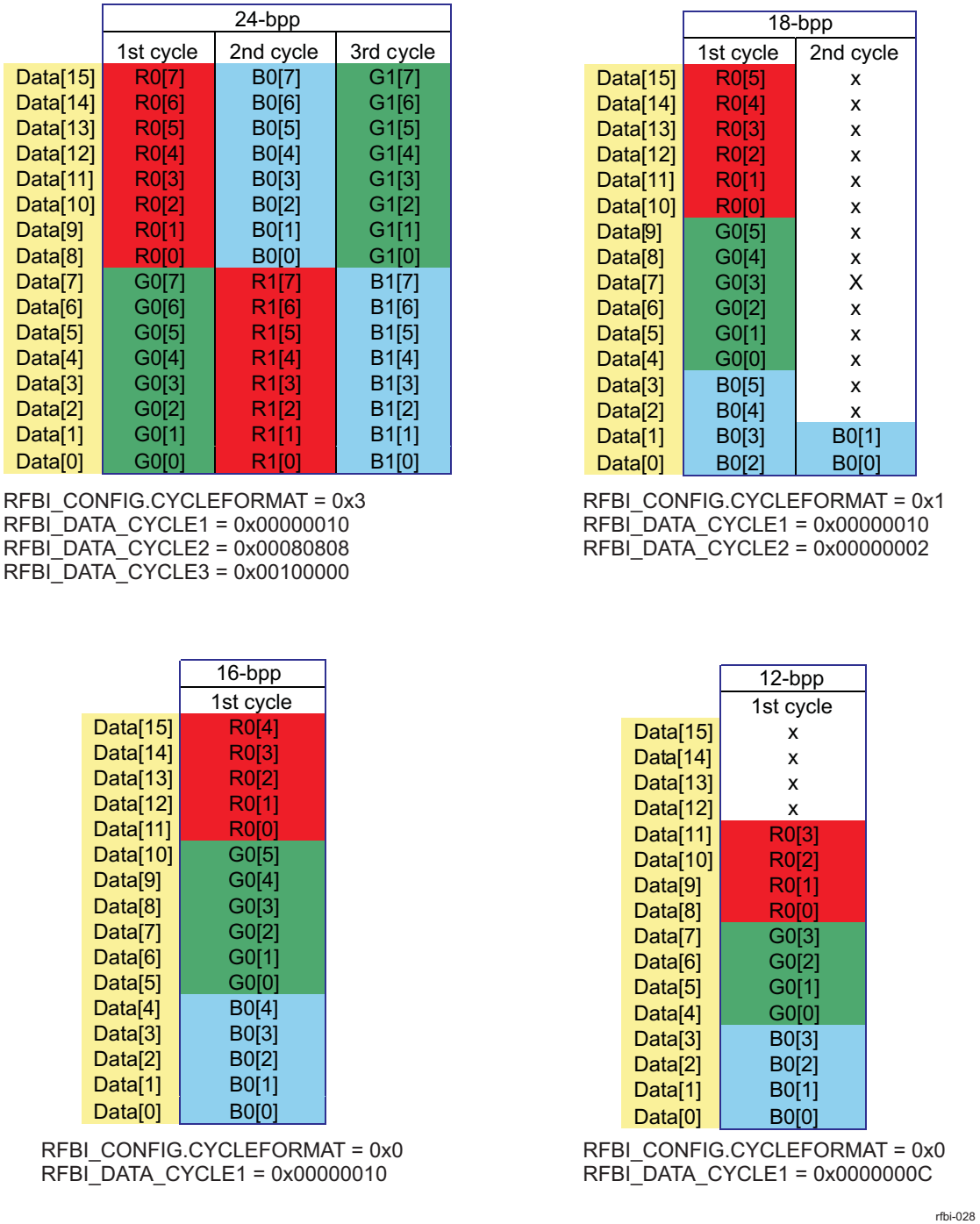


Figure 11-212. RFBI 16-Bit Interface Settings



11.3.4.2.9.2 RFBI Unmodified Bits

In a cycle, if every bit in the interface does not have a pixel value, the status of the unused bits can be programmed to be 0, 1, or the previous value (I/O power consumption optimization). Based on the configuration, the undefined bits for each cycle are defined with the previous values of the bits at the same position in the previous cycle, 0s, or 1s (the unused bits can be at any position). The RFBI_CONFIG__0/RFBI_CONFIG__1[12-11] UNUSEDBITS bit field is used.

11.3.4.2.9.3 RFBI Number of Pixels to Transfer

The RFBI_PIXEL_CNT[31-0] PIXELCNT bit field indicates the number of pixels to be transferred to the LCD panel. The value can be changed only when the RFBI_CONTROL[0] ENABLE bit is reset.

During the transfer, the hardware decrements the register when a pixel is sent to the RFB. When the RFBI_CONTROL[0] ENABLE bit is set and a new value is written in the RFBI_PIXEL_CNT register, and the current value in the register is a not 0 (the remaining number of pixels to transfer), the ongoing transfer is aborted.

11.3.4.2.10 RFBI Trigger Detection Logic

11.3.4.2.10.1 RFBI Trigger Mode

Setting the RFBI_CONFIG__0/RFBI_CONFIG__1[3-2] TRIGGERMODE bit field configures the different trigger modes:

- Internal trigger mode with the internal programmable RFBI_CONTROL[4] ITE bit
- External trigger mode with the external TE signal (RFBI_TEVSYNC)
- External trigger mode with the external VSYNC/HSYNC signals

11.3.4.2.10.2 RFBI Internal Trigger Mode

Set the RFBI_CONTROL[4] ITE bit to start capturing the data from the DISPC. The DISPC must be configured in the RFBI mode to account for the STALL signal. Setting the trigger mode to external (RFBI_CONFIG__0/RFBI_CONFIG__1[3-2] TRIGGERMODE bit field set to 0x1 or 0x2) causes the RFBI_CONTROL[4] ITE bit to be ignored. The chip-select CS0 must be selected (RFBI_CONTROL[3-2] CONFIGSELECT set to 0x1) when this bit is set by users.

11.3.4.2.10.3 RFBI External Trigger Mode

There are two external trigger modes:

- TE only: VSYNC and HSYNC are merged by logical OR operation and are detected through VSYNC and HSYNC pulse widths.
- HSYNC/VSYNC: This mode uses the two external signals, which are detected through VSYNC and HSYNC pulse widths.

11.3.4.2.10.3.1 RFBI Programmable Line Number

When the trigger mode is set to external trigger mode with HSYNC and VSYNC or the TE, hardware resets the line counter when the VSYNC occurs and, after a programmable number of lines programmed by the user in the RFBI_LINE_NUMBER[10-0] LINENUMBER bit field (the HSYNC pulse occurs for every line), the transfer to the LCD panel begins. When the programmable line number is 0, only the VSYNC pulse indicates the beginning of the transfer in both modes: HSYNC/VSYNC and TE (logical OR operation between HSYNC and VSYNC).

11.3.4.2.10.3.2 RFBI VSYNC Pulse Width (Minimum Value)

The RFBI_VSYNC_WIDTH[15-0] MINVSYNCPULSEWIDTH bit field defines the minimum number of DSS_OCP_CLK cycles of the VSYNC pulse for detection on VSYNC. It allows differentiation between VSYNC and HSYNC, which are OR-ed on the same signal, and is also used in HSYNC/VSYNC mode on the two separate input lines.

- The VSYNC pulse width must be equal to at least two DSS_OCP_CLK cycles when HSYNC is not present (TE trigger mode only).
- The VSYNC pulse width must be equal to at least four DSS_OCP_CLK cycles when HSYNC is present (TE trigger mode only).

11.3.4.2.10.3.3 RFBI HSYNC Pulse Width (Minimum Value)

The RFBI_HSYNC_WIDTH[15-0] MINHSYNCPULSEWIDTH bit field defines the minimum number of DSS_OCP_CLK cycles of the HSYNC pulse for detection on HSYNC. It allows differentiation between VSYNC and HSYNC, which are OR-ed on the same signal, and is also used in VSYNC/HSYNC mode on the separate two input lines. To be detected, the HSYNC/VSYNC pulse width must always be equal to at least two DSS_OCP_CLK cycles. See Table 11-422.

Table 11-422. RFBI Minimum Pulse Width (HSYNC/VSYNC)

Configuration Bits	TE Mode	HSYNC/VSYNC Mode
RFBI_HSYNC_WIDTH[15-0] MINHSYNCPULSEWIDTH field value	2	2
RFBI_VSYNC_WIDTH[15-0] MINVSYNCPULSEWIDTH field value	4	2

The pulse received by the RFBI must be at least two DSS_OCP_CLK cycles to be detected. For the TE mode, since the minimum value to differentiate VSYNC and HSYNC is two DSS_OCP_CLK cycles, the VSYNC pulse width must be at least four DSS_OCP_CLK cycles and the HSYNC pulse width must be at least two DSS_OCP_CLK cycles.

11.3.4.2.10.4 RFBI Hardware Status Features

Table 11-423 lists the RFBI hardware status features.

Table 11-423. RFBI Hardware Status Features

Feature	Type	Register/Bit Field	Description
Data pending	Status	RFBI_SYSSTATUS[9] BUSYRFBIDATA	It is set to 1 when some data are pending to be processed from the interconnect FIFO.
Slave port bus busy	Status	RFBI_SYSSTATUS[9] BUSY	It is set to 1 when the slave port bus is busy and access to some RFBI registers (RFBI_CMD, RFBI_STATUS, RFBI_PARAM, and RFBI_READ) is stalled.

11.3.5 DSS Registers

This section describes the DSS registers.

11.3.5.1 DSSUL_0_CFG Registers

Table 11-425 lists the memory-mapped registers for the DSSUL_0_CFG. All register offset addresses not listed in Table 11-425 should be considered as reserved locations and the register contents should not be modified.

This section describes the DSSUL_0_CFG instances registers.

Table 11-424. DSSUL_0_CFG Instances

Instance	Base Address
DSSUL_0_CFG	0254 0000h

Table 11-425. DSSUL_0_CFG Registers

Offset	Acronym	Register Name	DSSUL_0_CFG Physical Address	Section
0h	DSS_REVISION	This register contains the DSS revision number.	0254 0000h	Section 11.3.5.1.1
10h	DSS_SYSCONFIG	This register controls the various parameters of the OCP interface	0254 0010h	Section 11.3.5.1.2
14h	DSS_SYSSTATUS	This register provides status information about the module.	0254 0014h	Section 11.3.5.1.3
18h	DSS_RFBI_CTRL	This register contains control bits corresponding to the RFBI instance in DSS	0254 0018h	Section 11.3.5.1.4
1Ch	DSS_DPI_CTRL	This register contains control bits corresponding to the DPI interface in DSS	0254 001Ch	Section 11.3.5.1.5
40h	DSS_DEBUG_CFG	Debug configuration. **Feature is not supported in this family of devices.**	0254 0040h	Section 11.3.5.1.6

11.3.5.1.1 DSS_REVISION Register (Offset = 0h) [reset = 72h]

DSS_REVISION is shown in Figure 11-213 and described in Table 11-427.

Return to Summary Table.

This register contains the DSS revision number.

Table 11-426. DSS_REVISION Instances

Instance	Physical Address
DSSUL_0_CFG	0254 0000h

Figure 11-213. DSS_REVISION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REVISION																															
R-72h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-427. DSS_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REVISION	R	72h	TI internal data. Identifies revision of peripheral.

Table 11-428. Register Call Summary for DSS_REVISION

DSSUL_0_CFG Registers

- DSS_REVISION Register (Offset = 0h) [reset = 72h]: [0]
- DSSUL_0_CFG Registers: [0]

11.3.5.1.2 DSS_SYSCONFIG Register (Offset = 10h) [reset = 2h]

DSS_SYSCONFIG is shown in Figure 11-214 and described in Table 11-430.

Return to Summary Table.

This register controls the various parameters of the OCP interface

Table 11-429. DSS_SYSCONFIG Instances

Instance	Physical Address
DSSUL_0_CFG	0254 0010h

Figure 11-214. DSS_SYSCONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							RESERVED
R-0h							R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	RESERVED		SIDLEMODE	
R-0h			R-0h	R-0h		R/W-2h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-430. DSS_SYSCONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	RESERVED	R	0h	Reserved
15-5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3-2	RESERVED	R	0h	Reserved
1-0	SIDLEMODE	R/W	2h	0h (R/W) = Force-idle. An idle request is acknowledged unconditionally. 1h (R/W) = No-idle. An idle request is never acknowledged. **Feature is not supported in this family of devices.** 2h (R/W) = Smart-idle. Acknowledgement to an idle request is given based on the internal activity of the module. 3h (R/W) = Reserved

Table 11-431. Register Call Summary for DSS_SYSCONFIG

Display Subsystem (DSS)
DSSUL_0_CFG Registers • DSSUL_0_CFG Registers: [0] • DSS_SYSCONFIG Register (Offset = 10h) [reset = 2h]: [0]

11.3.5.1.3 DSS_SYSSTATUS Register (Offset = 14h) [reset = 0h]

DSS_SYSSTATUS is shown in Figure 11-215 and described in Table 11-433.

Return to Summary Table.

This register provides status information about the module.

Table 11-432. DSS_SYSSTATUS Instances

Instance	Physical Address
DSSUL_0_CFG	0254 0014h

Figure 11-215. DSS_SYSSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RFBI_RESETDONE	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	DSS_RESETDONE
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-433. DSS_SYSSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	RFBI_RESETDONE	R	0h	Reset status of RFBI module 0h (R) = Internal module reset is on-going 1h (R) = Reset completed
6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	DSS_RESETDONE	R	0h	Reset status of DISPC/DSS 0h (R) = Internal module reset is on-going 1h (R) = Reset completed

Table 11-434. Register Call Summary for DSS_SYSSTATUS

DSSUL_0_CFG Registers • DSSUL_0_CFG Registers: [0] • DSS_SYSSTATUS Register (Offset = 14h) [reset = 0h]: [0]
DSS Integration • DSS Clocks and Resets: [0]

11.3.5.1.4 DSS_RFBI_CTRL Register (Offset = 18h) [reset = 0h]

DSS_RFBI_CTRL is shown in Figure 11-216 and described in Table 11-436.

Return to Summary Table.

This register contains control bits corresponding to the RFBI instance in DSS

Table 11-435. DSS_RFBI_CTRL Instances

Instance	Physical Address
DSSUL_0_CFG	0254 0018h

Figure 11-216. DSS_RFBI_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RFBI_ENABLE
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-436. DSS_RFBI_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	RFBI_ENABLE	R/W	0h	RFBI Enable. This bit is used to control the RFBI module. 0h (R/W) = Disable 1h (R/W) = Enable

Table 11-437. Register Call Summary for DSS_RFBI_CTRL

Display Subsystem (DSS) • DSS Environment: [0]
DSSUL_0_CFG Registers • DSSUL_0_CFG Registers: [0] • DSS_RFBI_CTRL Register (Offset = 18h) [reset = 0h]: [0]

11.3.5.1.5 DSS_DPI_CTRL Register (Offset = 1Ch) [reset = 1h]

DSS_DPI_CTRL is shown in Figure 11-217 and described in Table 11-439.

Return to Summary Table.

This register contains control bits corresponding to the DPI interface in DSS

Table 11-438. DSS_DPI_CTRL Instances

Instance	Physical Address
DSSUL_0_CFG	0254 001Ch

Figure 11-217. DSS_DPI_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							DPI_ENABLE
R-0h							R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-439. DSS_DPI_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	DPI_ENABLE	R/W	1h	Enable DPI interface. This bit is used to enable the DPI interface. 0h (R/W) = Disable - PCLK is gated 1h (R/W) = Enable

Table 11-440. Register Call Summary for DSS_DPI_CTRL

Display Subsystem (DSS) • DSS Environment: [0][1]
DSSUL_0_CFG Registers • DSS_DPI_CTRL Register (Offset = 1Ch) [reset = 1h]: [0] • DSSUL_0_CFG Registers: [0]

11.3.5.1.6 DSS_DEBUG_CFG Register (Offset = 40h) [reset = 0h]

DSS_DEBUG_CFG is shown in Figure 11-218 and described in Table 11-442.

Return to Summary Table.

Debug configuration.

Feature is not supported in this family of devices.

Table 11-441. DSS_DEBUG_CFG Instances

Instance	Physical Address
DSSUL_0_CFG	0254 0040h

Figure 11-218. DSS_DEBUG_CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													CFG		
R-0h																													R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-442. DSS_DEBUG_CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2-0	CFG	R/W	0h	Defines which debug bus to provide on the DSS debug bus connected at the top. Only value 0 can be used in this family of devices. 0h (R/W) = Select DISPC Debug bus

Table 11-443. Register Call Summary for DSS_DEBUG_CFG

DSSUL_0_CFG Registers

- DSSUL_0_CFG Registers: [0]
- DSS_DEBUG_CFG Register (Offset = 40h) [reset = 0h]: [0]

11.3.5.2 DISPC_COMMON Registers

Table 11-445 lists the memory-mapped registers for the DISPC_COMMON. All register offset addresses not listed in Table 11-445 should be considered as reserved locations and the register contents should not be modified.

This section describes the DISPC_COMMON instances registers.

Table 11-444. DISPC_COMMON Instances

Instance	Base Address
DISPC_COMMON	0255 0000h

Table 11-445. DISPC_COMMON Registers

Offset	Acronym	Register Name	DISPC_COMMON Physical Address	Section
0h	DISPC_REVISION	This register contains the IP revision code	0255 0000h	Section 11.3.5.2.1
4h	DISPC_SYSCONFIG	This register allows to control various parameters of the OCP interface.	0255 0004h	Section 11.3.5.2.2
8h	DISPC_SYSSTATUS	This register provides status information about the module, excluding the interrupt status information.	0255 0008h	Section 11.3.5.2.3
20h	DISPC_IRQ_EOI	End Of Interrupt number	0255 0020h	Section 11.3.5.2.4
24h	DISPC_IRQSTATUS_RAW	Per-end of group (31 down to 0) internal signaling raw interrupt status vector, line #0. Raw status is set even if end of group (31 down to 0) interrupt is not enabled. Write 1 to set the (raw) status, mostly for debug.	0255 0024h	Section 11.3.5.2.5
28h	DISPC_IRQSTATUS	Per-end of group (31 down to 0) internal signaling 'enabled' interrupt status vector, line #0. Enabled status isn't set unless event is enabled. Write 1 to clear the status after interrupt has been serviced (raw status gets cleared, i.e. even if not enabled).	0255 0028h	Section 11.3.5.2.6
2Ch	DISPC_IRQENABLE_SET	Per-end of group (31 down to 0) internal event interrupt enable bit vector, line #0. Write 1 to set (enable interrupt). Readout equal to corresponding _CLR register.	0255 002Ch	Section 11.3.5.2.7
30h	DISPC_IRQENABLE_CLR	See description of corresponding _SET register.	0255 0030h	Section 11.3.5.2.8
34h	DISPC_IRQWAKEEN	IRQ wake up register. **Note: Feature is not supported in this family of devices.**	0255 0034h	Section 11.3.5.2.9
40h	DISPC_GLOBAL_MFLAG_ATTRIBUTE	MFLAG control register	0255 0040h	Section 11.3.5.2.10
44h	DISPC_GLOBAL_BUFFER	The register configures the DMA buffers allocations to the pipeline.	0255 0044h	Section 11.3.5.2.11
48h	DISPC_BA0_FLIPIMMEDIATE_EN	**Note: Feature is not supported in this family of devices.**	0255 0048h	Section 11.3.5.2.12
4Ch	DISPC_DBG_CONTROL	DISPC debug bus control register.	0255 004Ch	Section 11.3.5.2.13
50h	DISPC_DBG_STATUS	DISPC debug status register.	0255 0050h	Section 11.3.5.2.14
54h	DISPC_CLKGATING_DISABLE	Register to control clock gating at DISPC sub-module level	0255 0054h	Section 11.3.5.2.15

11.3.5.2.1 DISPC_REVISION Register (Offset = 0h) [reset = 82h]

DISPC_REVISION is shown in Figure 11-219 and described in Table 11-447.

Return to Summary Table.

This register contains the IP revision code

Table 11-446. DISPC_REVISION Instances

Instance	Physical Address
DISPC_COMMON	0255 0000h

Figure 11-219. DISPC_REVISION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							
R-0h																								R-82h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-447. DISPC_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
7-0	REV	R	82h	TI internal data. Identifies revision of peripheral.

Table 11-448. Register Call Summary for DISPC_REVISION

DISPC_COMMON Registers

- DISPC_COMMON Registers: [0]
- DISPC_REVISION Register (Offset = 0h) [reset = 82h]: [0]

11.3.5.2.2 DISPC_SYSCONFIG Register (Offset = 4h) [reset = 2011h]

DISPC_SYSCONFIG is shown in Figure 11-220 and described in Table 11-450.

Return to Summary Table.

This register allows to control various parameters of the OCP interface.

Table 11-449. DISPC_SYSCONFIG Instances

Instance	Physical Address
DISPC_COMMON	0255 0004h

Figure 11-220. DISPC_SYSCONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		MIDLEMODE		RESERVED		CLOCKACTIVITY	
R-0h		R/W-2h		R-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		WARMRESET	SIDLEMODE		ENWAKEUP	SOFTRESET	AUTOIDLE
R-0h		R/W-0h	R/W-2h		R/W-0h	R/W-0h	R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-450. DISPC_SYSCONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Write 0's for future compatibility. Reads returns 0
13-12	MIDLEMODE	R/W	2h	Master interface power management, standby/wait control 0h (R/W) = Force-standby.MStandby is only asserted when the module is disabled. MStandby is only asserted when the module is disabled. 1h (R/W) = No-Standby:MStandby is never asserted. **Note: Feature not supported in this family of devices.** 2h (R/W) = Smart-Standby.MStandby is asserted based on the internal activity of the module 3h (R/W) = Reserved
11-10	RESERVED	R	0h	Write 0's for future compatibility. Reads returns 0
9-8	CLOCKACTIVITY	R/W	0h	Clocks activity during wake up mode period 0h (R/W) = OCP and Functional clocks can be switched off 1h (R/W) = Functional clocks can be switched off and OCP clocks are mantained during wake up period 2h (R/W) = OCP clocks can be switched off and Functional clocks are maintained during wake up period 3h (R/W) = OCP and Functional clocks are maintained during wake up period
7-6	RESERVED	R	0h	Write 0's for future compatibility. Reads returns 0
5	WARMRESET	R/W	0h	Warm reset. Set this bit to 1 triggers a module warm reset. The bit is automatically reset by the hardware. During reads, it always returns 0. The warm reset keep the configuration registers unchanged. 0h (R/W) = Normal mode 1h (R/W) = The warmreset is set

Table 11-450. DISPC_SYSCONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	SIDLEMODE	R/W	2h	Slave interface power management, Idle req/ack control 0h (R/W) = Force-idle.An idle request is acknowledged unconditionally 1h (R/W) = No-idle. An idle request is never acknowledged . **Note: Feature not supported in this family of devices.** 2h (R/W) = Smart-idle. Acknowledgement to an idle request is given based on the internal activity of the module. 3h (R/W) = Reserved
2	ENWAKEUP	R/W	0h	WakeUp feature control. **Note: Feature not supported in this family of devices.** 0h (R/W) = Wakeup is disabled 1h (R/W) = Wakeup is enabled
1	SOFTRESET	R/W	0h	Software reset. Set this bit to 1 to trigger a module reset. The bit is automatically reset by the hardware. During reads, it always returns 0. 0h (R/W) = Normal mode 1h (R/W) = The module is reset
0	AUTOIDLE	R/W	1h	Internal OCP clock gating strategy 0h (R/W) = OCP clock is free-running 1h (R/W) = Automatic OCP clocks gating strategy is applied, based on the OCP interface activity. Automatic functional clock gating is also applied to the functional clock based on the module activity (for instance DISPC_ pipe _ATTRIBUTES.ENABLE)

Table 11-451. Register Call Summary for DISPC_SYSCONFIG

DSS Functional Description • DISPC Software Reset: [0][1] • DISPC DMA Power Modes: [0]
Display Subsystem (DSS)
DISPC_COMMON Registers • DISPC_COMMON Registers: [0] • DISPC_SYSCONFIG Register (Offset = 4h) [reset = 2011h]: [0]

11.3.5.2.3 DISPC_SYSSTATUS Register (Offset = 8h) [reset = 0h]

DISPC_SYSSTATUS is shown in Figure 11-221 and described in Table 11-453.

Return to Summary Table.

This register provides status information about the module, excluding the interrupt status information.

Table 11-452. DISPC_SYSSTATUS Instances

Instance	Physical Address
DISPC_COMMON	0255 0008h

Figure 11-221. DISPC_SYSSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	RESERVED	RESERVED	DISPC_VP1_RESETDONE	DISPC_FUNC_RESETDONE
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-453. DISPC_SYSSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	DISPC_VP1_RESETDONE	R	0h	Reset status of DISPC VP1 pixel clock domain 0h (R) = Internal module reset is on-going 1h (R) = Reset completed
0	DISPC_FUNC_RESETDONE	R	0h	Reset status of DISPC Functional clock domain 0h (R) = Internal module reset is on-going 1h (R) = Reset completed

Table 11-454. Register Call Summary for DISPC_SYSSTATUS

DSS Functional Description • DISPC Software Reset: [0][1]
DISPC_COMMON Registers • DISPC_COMMON Registers: [0] • DISPC_SYSSTATUS Register (Offset = 8h) [reset = 0h]: [0]

11.3.5.2.4 DISPC_IRQ_EOI Register (Offset = 20h) [reset = 0h]

DISPC_IRQ_EOI is shown in Figure 11-222 and described in Table 11-456.

Return to Summary Table.

End Of Interrupt number (for H08 interrupt)

Table 11-455. DISPC_IRQ_EOI Instances

Instance	Physical Address
DISPC_COMMON	0255 0020h

Figure 11-222. DISPC_IRQ_EOI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							LINE_NUMBER
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-456. DISPC_IRQ_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	LINE_NUMBER	R/W	0h	Software End Of Interrupt (EOI) control. Write number of interrupt output. The IP has 1 interrupt compliant with H08. 0h (R/W) = Reads always 0 (no EOI memory)

Table 11-457. Register Call Summary for DISPC_IRQ_EOI

DISPC_COMMON Registers

- DISPC_COMMON Registers: [0]
- DISPC_IRQ_EOI Register (Offset = 20h) [reset = 0h]: [0]

11.3.5.2.5 DISPC_IRQSTATUS_RAW Register (Offset = 24h) [reset = 0h]

DISPC_IRQSTATUS_RAW is shown in Figure 11-223 and described in Table 11-459.

Return to Summary Table.

Per-end of group (31 down to 0) internal signaling raw interrupt status vector, line #0. Raw status is set even if end of group (31 down to 0) interrupt is not enabled. Write 1 to set the (raw) status, mostly for debug.

Table 11-458. DISPC_IRQSTATUS_RAW Instances

Instance	Physical Address
DISPC_COMMON	0255 0024h

Figure 11-223. DISPC_IRQSTATUS_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		WAKEUP_IRQ	RESERVED	WB_IRQ	RESERVED	RESERVED	RESERVED
R-0h		R/W-0h	R-0h	R/W-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
VID1_IRQ	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	VP1_IRQ
R/W-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-459. DISPC_IRQSTATUS_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13	WAKEUP_IRQ	R/W	0h	Wake-up. **Note: Feature is not supported in this family of devices.** 0h (R/W) = No event pending 1h (R/W) = IRQ event pending
12	RESERVED	R	0h	Reserved
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	VID1_IRQ	R/W	0h	VID1 IRQ STATUS register indicates the video pipeline 1 interrupt events 0h (R/W) = No event pending 1h (R/W) = IRQ event pending
6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved

Table 11-459. DISPC_IRQSTATUS_RAW Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	RESERVED	R	0h	Reserved
0	VP1_IRQ	R/W	0h	VP1 IRQ STATUS register indicates the Video Port 1 interrupt events 0h (R/W) = No event pending 1h (R/W) = IRQ event pending

Table 11-460. Register Call Summary for DISPC_IRQSTATUS_RAW

DISPC_COMMON Registers

- DISPC_COMMON Registers: [0]
- DISPC_IRQSTATUS_RAW Register (Offset = 24h) [reset = 0h]: [0]

11.3.5.2.6 DISPC_IRQSTATUS Register (Offset = 28h) [reset = 0h]

DISPC_IRQSTATUS is shown in Figure 11-224 and described in Table 11-462.

Return to Summary Table.

Per-end of group (31 down to 0) internal signaling 'enabled' interrupt status vector, line #0. Enabled status isn't set unless event is enabled. Write 1 to clear the status after interrupt has been serviced (raw status gets cleared, i.e. even if not enabled).

Table 11-461. DISPC_IRQSTATUS Instances

Instance	Physical Address
DISPC_COMMON	0255 0028h

Figure 11-224. DISPC_IRQSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		WAKEUP_IRQ	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h		R/W-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
VID1_IRQ	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	VP1_IRQ
R/W-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-462. DISPC_IRQSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13	WAKEUP_IRQ	R/W	0h	Wake-up. **Note: Feature is not supported in this family of devices.** 0h (R/W) = No event pending 1h (R/W) = IRQ event pending
12	RESERVED	R	0h	Reserved
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	VID1_IRQ	R/W	0h	VID1 IRQ STATUS register indicates the video pipeline 1 interrupt events 0h (R/W) = No event pending 1h (R/W) = IRQ event pending
6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved

Table 11-462. DISPC_IRQSTATUS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	RESERVED	R	0h	Reserved
0	VP1_IRQ	R/W	0h	VP1 IRQ STATUS register indicates the Video Port 1 interrupt events 0h (R/W) = No event pending 1h (R/W) = IRQ event pending

Table 11-463. Register Call Summary for DISPC_IRQSTATUS

DSS Functional Description • DISPC Interrupt Requests: [0][1]
DISPC_COMMON Registers • DISPC_COMMON Registers: [0] • DISPC_IRQSTATUS Register (Offset = 28h) [reset = 0h]: [0]

11.3.5.2.7 DISPC_IRQENABLE_SET Register (Offset = 2Ch) [reset = 0h]

DISPC_IRQENABLE_SET is shown in Figure 11-225 and described in Table 11-465.

Return to Summary Table.

Per-end of group (31 down to 0) internal event interrupt enable bit vector, line #0. Write 1 to set (enable interrupt). Readout equal to corresponding _CLR register.

Table 11-464. DISPC_IRQENABLE_SET Instances

Instance	Physical Address
DISPC_COMMON	0255 002Ch

Figure 11-225. DISPC_IRQENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		SET_WAKEUP_IRQ	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h		R/W-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
SET_VID1_IRQ	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	SET_VP1_IRQ
R/W-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-465. DISPC_IRQENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13	SET_WAKEUP_IRQ	R/W	0h	Wake Up Mask. **Note: Feature is not supported in this family of devices.** 0h (R/W) = interrupt disabled 1h (R/W) = interrupt enabled
12	RESERVED	R	0h	Reserved
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	SET_VID1_IRQ	R/W	0h	VID1 IRQ 0h (R/W) = interrupt disabled 1h (R/W) = interrupt enabled
6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved

Table 11-465. DISPC_IRQENABLE_SET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	SET_VP1_IRQ	R/W	0h	VP1 IRQ 0h (R/W) = interrupt disabled 1h (R/W) = interrupt enabled

Table 11-466. Register Call Summary for DISPC_IRQENABLE_SET

DSS Functional Description • DISPC Interrupt Requests: [0][1]
DISPC_COMMON Registers • DISPC_COMMON Registers: [0] • DISPC_IRQENABLE_SET Register (Offset = 2Ch) [reset = 0h]: [0]

11.3.5.2.8 DISPC_IRQENABLE_CLR Register (Offset = 30h) [reset = 0h]

DISPC_IRQENABLE_CLR is shown in Figure 11-226 and described in Table 11-468.

Return to Summary Table.

Table 11-467. DISPC_IRQENABLE_CLR Instances

Instance	Physical Address
DISPC_COMMON	0255 0030h

Figure 11-226. DISPC_IRQENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		CLR_WAKEUP_IRQ	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h		R/W-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
CLR_VID1_IRQ	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	CLR_VP1_IRQ
R/W-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-468. DISPC_IRQENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13	CLR_WAKEUP_IRQ	R/W	0h	Wake Up Mask. **Note: Feature is not supported in this family of devices.** 0h (R/W) = interrupt disabled 1h (R/W) = interrupt enabled
12	RESERVED	R	0h	Reserved
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	CLR_VID1_IRQ	R/W	0h	VID1 IRQ 0h (R/W) = interrupt disabled 1h (R/W) = interrupt enabled
6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	CLR_VP1_IRQ	R/W	0h	VP1 IRQ 0h (R/W) = interrupt disabled 1h (R/W) = interrupt enabled

Table 11-469. Register Call Summary for DISPC_IRQENABLE_CLR

DSS Functional Description • DISPC Interrupt Requests: [0]
DISPC_COMMON Registers • DISPC_COMMON Registers: [0] • DISPC_IRQENABLE_CLR Register (Offset = 30h) [reset = 0h]: [0]

11.3.5.2.9 DISPC_IRQWAKEEN Register (Offset = 34h) [reset = 0h]

DISPC_IRQWAKEEN is shown in Figure 11-227 and described in Table 11-471.

Return to Summary Table.

IRQ wake up register.
Note: Feature is not supported in this family of devices.

Table 11-470. DISPC_IRQWAKEEN Instances

Instance	Physical Address
DISPC_COMMON	0255 0034h

Figure 11-227. DISPC_IRQWAKEEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				RESERVED	RESERVED	RESERVED	RESERVED
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
VID1_IRQWAKEEN	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	VP1_IRQWAKEEN
R/W-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-471. DISPC_IRQWAKEEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	VID1_IRQWAKEEN	R/W	0h	Wakeupen for VID1 first level interrupt. **Note: Feature is not supported in this family of devices.** 0h (R/W) = Swakeup is not generated when this interrupt is asserted in idle mode 1h (R/W) = Swakeup is generated when this interrupt is asserted in idle mode
6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved

Table 11-471. DISPC_IRQWAKEEN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	VP1_IRQWAKEEN	R/W	0h	Wakeupen for VP1 first level interrupt. **Note: Feature is not supported in this family of devices.** 0h (R/W) = Swakeup is not generated when this interrupt is asserted in idle mode 1h (R/W) = Swakeup is generated when this interrupt is asserted in idle mode

Table 11-472. Register Call Summary for DISPC_IRQWAKEEN

DISPC_COMMON Registers

- DISPC_COMMON Registers: [0]
- DISPC_IRQWAKEEN Register (Offset = 34h) [reset = 0h]: [0]

11.3.5.2.10 DISPC_GLOBAL_MFLAG_ATTRIBUTE Register (Offset = 40h) [reset = 0h]

DISPC_GLOBAL_MFLAG_ATTRIBUTE is shown in Figure 11-228 and described in Table 11-474.

Return to Summary Table.

MFLAG control register

Table 11-473. DISPC_GLOBAL_MFLAG_ATTRIBUTE Instances

Instance	Physical Address
DISPC_COMMON	0255 0040h

Figure 11-228. DISPC_GLOBAL_MFLAG_ATTRIBUTE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					MFLAG_START	MFLAG_CTRL	
R-0h					R/W-0h	R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-474. DISPC_GLOBAL_MFLAG_ATTRIBUTE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2	MFLAG_START	R/W	0h	0h (R/W) = reset value, when the DMA buffer is empty at the beginning of the frame, the MFLAG of each pipe is kept at 0 until PRELOAD is reached, then based on MFLAG_CTRL, MFLAG[1:0] are generated and internal logic is arbitrating between pipeline requests 1h (R/W) = Even at the beginning of the frame when the DMA buffer is empty, MFLAG_CTRL is used to determine how MFLAG dedicated to each pipe signal shall be driven
1-0	MFLAG_CTRL	R/W	0h	0h (R/W) = MFLAG mechanism is disabled: MFLAG[1:0] out band signals are set to 0 1h (R/W) = MFLAG mechanism is enabled: MFLAG[1:0] out band signals are always set to 1 2h (R/W) = MFLAG mechanism is enabled and MFLAG[1:0] out band signals are dynamically set to 0 or 1 depending on the MFLAG rules

Table 11-475. Register Call Summary for DISPC_GLOBAL_MFLAG_ATTRIBUTE

DSS Functional Description • DISPC DMA MFLAG Mechanism: [0][1][2][3][4]
DISPC_COMMON Registers • DISPC_COMMON Registers: [0] • DISPC_GLOBAL_MFLAG_ATTRIBUTE Register (Offset = 40h) [reset = 0h]: [0]

11.3.5.2.11 DISPC_GLOBAL_BUFFER Register (Offset = 44h) [reset = 800h]

DISPC_GLOBAL_BUFFER is shown in Figure 11-229 and described in Table 11-477.

Return to Summary Table.

The register configures the DMA buffers allocations to the pipeline.

Table 11-476. DISPC_GLOBAL_BUFFER Instances

Instance	Physical Address
DISPC_COMMON	0255 0044h

Figure 11-229. DISPC_GLOBAL_BUFFER Register

31	30	29	28	27	26	25	24
BUFFERFILLING	RESERVED						
R/W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED			RESERVED			RESERVED	
R-0h			R-0h			R-0h	
15	14	13	12	11	10	9	8
RESERVED	RESERVED			VID1_BUFFER			RESERVED
R-0h	R-0h			R/W-4h			R-0h
7	6	5	4	3	2	1	0
RESERVED		RESERVED			RESERVED		
R-0h		R-0h			R-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-477. DISPC_GLOBAL_BUFFER Register Field Descriptions

Bit	Field	Type	Reset	Description
31	BUFFERFILLING	R/W	0h	Controls if the DMA buffers are re-filled only when the LOW threshold is reached or if all DMA buffers are re-filled when at least one of them reaches the LOW threshold. wr: immediate 0h (R/W) = Each DMA buffer is re-filled when it reaches LOW threshold. 1h (R/W) = All DMA buffers are re-filled up to high threshold when at least one of them reaches the LOW threshold. (only active DMA buffers shall be considered and when reaching the end of the frame the DMA buffer goes to empty condition so no need to fill it again).
30-21	RESERVED	R	0h	Reserved
20-18	RESERVED	R	0h	Reserved
17-15	RESERVED	R	0h	Reserved
14-12	RESERVED	R	0h	Reserved
11-9	VID1_BUFFER	R/W	4h	Video1 DMA buffer allocation to one of the pipelines. By default to video 1 pipeline. 0h (R/W) = DMA buffer is not allocated to a pipeline. 4h (R/W) = DMA buffer allocated to the video1 pipeline. Other values (R/W) = Not supported
8-6	RESERVED	R	0h	Reserved
5-3	RESERVED	R	0h	Reserved
2-0	RESERVED	R	0h	Reserved

Table 11-478. Register Call Summary for DISPC_GLOBAL_BUFFER

DSS Functional Description • DISPC DMA Engine: [0]
DISPC_COMMON Registers • DISPC_COMMON Registers: [0] • DISPC_GLOBAL_BUFFER Register (Offset = 44h) [reset = 800h]: [0]

11.3.5.2.12 DISPC_BA0_FLIPIMMEDIATE_EN Register (Offset = 48h) [reset = 0h]

DISPC_BA0_FLIPIMMEDIATE_EN is shown in Figure 11-230 and described in Table 11-480.

Return to Summary Table.

Note: Feature is not supported in this family of devices.

Table 11-479. DISPC_BA0_FLIPIMMEDIATE_EN Instances

Instance	Physical Address
DISPC_COMMON	0255 0048h

Figure 11-230. DISPC_BA0_FLIPIMMEDIATE_EN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-480. DISPC_BA0_FLIPIMMEDIATE_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

Table 11-481. Register Call Summary for DISPC_BA0_FLIPIMMEDIATE_EN

DISPC_COMMON Registers

- DISPC_COMMON Registers: [0]
- DISPC_BA0_FLIPIMMEDIATE_EN Register (Offset = 48h) [reset = 0h]: [0]

11.3.5.2.13 DISPC_DBG_CONTROL Register (Offset = 4Ch) [reset = 0h]

DISPC_DBG_CONTROL is shown in Figure 11-231 and described in Table 11-483.

Return to Summary Table.

DISPC debug bus control register.

Table 11-482. DISPC_DBG_CONTROL Instances

Instance	Physical Address
DISPC_COMMON	0255 004Ch

Figure 11-231. DISPC_DBG_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							DBGMUXSEL
R-0h							R/W-0h
7	6	5	4	3	2	1	0
DBGMUXSEL							DBGEN
R/W-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-483. DISPC_DBG_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8-1	DBGMUXSEL	R/W	0h	0h (R/W) = Select first [31:0] bits of VID-1 debug bus 1h (R/W) = Select first [63:32] bits of VID-1 debug bus 2h (R/W) = Select first [95:64] bits of VID-1 debug bus 3h (R/W) = Select first [127:96] bits of VID-1 debug bus 4h (R/W) = Select first [159:128] bits of VID-1 debug bus 5h (R/W) = Select first [191:160] bits of VID-1 debug bus 6h (R/W) = Select first [223:192] bits of VID-1 debug bus 7h (R/W) = Select first [255:224] bits of VID-1 debug bus 8h to 1Fh (R/W) = Reserved 20h (R/W) = Select VP1 debug bus 21h to FFh(R/W) = Reserved
0	DBGEN	R/W	0h	Enable debug ports 0h (R/W) = DBGDIS 1h (R/W) = DBGEN

Table 11-484. Register Call Summary for DISPC_DBG_CONTROL

DISPC_COMMON Registers
- DISPC_COMMON Registers: [0]
- DISPC_DBG_CONTROL Register (Offset = 4Ch) [reset = 0h]: [0]
- DISPC_DBG_STATUS Register (Offset = 50h) [reset = 0h]: [0]

11.3.5.2.14 DISPC_DBG_STATUS Register (Offset = 50h) [reset = 0h]

DISPC_DBG_STATUS is shown in Figure 11-232 and described in Table 11-486.

Return to Summary Table.

DISPC debug status register.

Table 11-485. DISPC_DBG_STATUS Instances

Instance	Physical Address
DISPC_COMMON	0255 0050h

Figure 11-232. DISPC_DBG_STATUS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBGOUT																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-486. DISPC_DBG_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBGOUT	R	0h	Debug bus data selected in DISPC_DBG_CONTROL register

Table 11-487. Register Call Summary for DISPC_DBG_STATUS

DISPC_COMMON Registers
- DISPC_COMMON Registers: [0]
- DISPC_DBG_STATUS Register (Offset = 50h) [reset = 0h]: [0]

11.3.5.2.15 DISPC_CLKGATING_DISABLE Register (Offset = 54h) [reset = 0h]

DISPC_CLKGATING_DISABLE is shown in Figure 11-233 and described in Table 11-489.

Return to Summary Table.

Register to control clock gating at DISPC sub-module level .

Table 11-488. DISPC_CLKGATING_DISABLE Instances

Instance	Physical Address
DISPC_COMMON	0255 0054h

Figure 11-233. DISPC_CLKGATING_DISABLE Register

31	30	29	28	27	26	25	24
RESERVED					RESERVED	RESERVED	RESERVED
R-0h					R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
VP1	RESERVED	RESERVED	RESERVED	OVR1	RESERVED	RESERVED	RESERVED
R/W-0h	R-0h	R-0h	R-0h	R/W-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	VID1	DMA_CH8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DMA_CH7	DMA_CH6	DMA_CH5	DMA_CH4	DMA_CH3	DMA_CH2	DMA_CH1	DMA_COMMON
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-489. DISPC_CLKGATING_DISABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	Reserved
26	RESERVED	R	0h	Reserved
25	RESERVED	R	0h	Reserved
24	RESERVED	R	0h	Reserved
23	VP1	R/W	0h	Clock gating control for VP1 0h (R/W) = Clock-Gating is enabled 1h (R/W) = Clock-gating is disabled. Clocks are free running
22	RESERVED	R	0h	Reserved
21	RESERVED	R	0h	Reserved
20	RESERVED	R	0h	Reserved
19	OVR1	R/W	0h	Clock gating control for OVR1 0h (R/W) = Clock-Gating is enabled 1h (R/W) = Clock-gating is disabled. Clocks are free running
18	RESERVED	R	0h	Reserved
17	RESERVED	R	0h	Reserved
16	RESERVED	R	0h	Reserved
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13	RESERVED	R	0h	Reserved
12	RESERVED	R	0h	Reserved

Table 11-489. DISPC_CLKGATING_DISABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	VID1	R/W	0h	Clock gating control for VID1 0h (R/W) = Clock-Gating is enabled 1h (R/W) = Clock-gating is disabled. Clocks are free running
8	DMA_CH8	R/W	0h	Clock gating control for DMA Channel-8 0h (R/W) = Clock-Gating is enabled 1h (R/W) = Clock-gating is disabled. Clocks are free running
7	DMA_CH7	R/W	0h	Clock gating control for DMA Channel-7 0h (R/W) = Clock-Gating is enabled 1h (R/W) = Clock-gating is disabled. Clocks are free running
6	DMA_CH6	R/W	0h	Clock gating control for DMA Channel-6 0h (R/W) = Clock-Gating is enabled 1h (R/W) = Clock-gating is disabled. Clocks are free running
5	DMA_CH5	R/W	0h	Clock gating control for DMA Channel-5 0h (R/W) = Clock-Gating is enabled 1h (R/W) = Clock-gating is disabled. Clocks are free running
4	DMA_CH4	R/W	0h	Clock gating control for DMA Channel-4 0h (R/W) = Clock-Gating is enabled 1h (R/W) = Clock-gating is disabled. Clocks are free running
3	DMA_CH3	R/W	0h	Clock gating control for DMA Channel-3 0h (R/W) = Clock-Gating is enabled 1h (R/W) = Clock-gating is disabled. Clocks are free running
2	DMA_CH2	R/W	0h	Clock gating control for DMA Channel-2 0h (R/W) = Clock-Gating is enabled 1h (R/W) = Clock-gating is disabled. Clocks are free running
1	DMA_CH1	R/W	0h	Clock gating control for DMA Channel-1 0h (R/W) = Clock-Gating is enabled 1h (R/W) = Clock-gating is disabled. Clocks are free running
0	DMA_COMMON	R/W	0h	Clock gating control for DMA_COMMON module 0h (R/W) = Clock-Gating is enabled 1h (R/W) = Clock-gating is disabled. Clocks are free running

Table 11-490. Register Call Summary for DISPC_CLKGATING_DISABLE

DISPC_COMMON Registers

- DISPC_COMMON Registers: [0]
- DISPC_CLKGATING_DISABLE Register (Offset = 54h) [reset = 0h]: [0]

11.3.5.3 DISPC_VID1 Registers

Table 11-492 lists the memory-mapped registers for the DISPC_VID1. All register offset addresses not listed in Table 11-492 should be considered as reserved locations and the register contents should not be modified.

This section describes the DISPC_VID1 instances registers.

Table 11-491. DISPC_VID1 Instances

Instance	Base Address
DISPC_VID1	0255 7000h

Table 11-492. DISPC_VID1 Registers

Offset	Acronym	Register Name	DISPC_VID1 Physical Address	Section
0h to 4h	DISPC_VID1_ACCUH_0 to DISPC_VID1_ACCUH_1	The register configures the resize accumulator init values for horizontal up/down-sampling of the video window (DISPC_VIDx_ACCU__0 & DISPC_VIDx_ACCU__1 for ping-pong mechanism with external trigger, based on the field polarity) It is used for ARGB and Y setting. Shadow register.	0255 7000h to 0255 7004h	Section 11.3.5.3.1
8h to Ch	DISPC_VID1_ACCUH2_0 to DISPC_VID1_ACCUH2_1	The register configures the resize accumulator init value for horizontal up/down-sampling of the video window (DISPC_VID#n_ACCU2__0 & DISPC_VID#n_ACCU2__1 for ping-pong mechanism with external trigger, based on the field polarity) It is used for Cb and Cr setting. It is used only when the pixel format at the input of the filter is one of the YUV formats. If the pixel format at the input of the filter is ARGB (all ARGB, RGB, RGBA are converted to ARGB32-8888 by the color space conversion before going to the filter is the color space conversion is done before the filter). When the register is not used by the HW, any value can be used for the bit-fields. Shadow register.	0255 7008h to 0255 700Ch	Section 11.3.5.3.2
10h to 14h	DISPC_VID1_ACCUV_0 to DISPC_VID1_ACCUV_1	The register configures the resize accumulator init values for horizontal and vertical up/down-sampling of the video window (DISPC_VIDx_ACCU__0 & DISPC_VIDx_ACCU__1 for ping-pong mechanism with external trigger, based on the field polarity) It is used for ARGB and Y setting. Shadow register.	0255 7010h to 0255 7014h	Section 11.3.5.3.3

Table 11-492. DISPC_VID1 Registers (continued)

Offset	Acronym	Register Name	DISPC_VID1 Physical Address	Section
18h to 1Ch	DISPC_VID1_ACCUV2_0 to DISPC_VID1_ACCUV2_1	The register configures the resize accumulator init value for vertical up/down-sampling of the video window (DISPC_VID1_ACCU2__0 & DISPC_VID1_ACCU2__1 for ping-pong mechanism with external trigger, based on the field polarity) It is used for Cb and Cr setting. It is used only when the pixel format at the input of the filter is one of the YUV formats. If the pixel format at the input of the filter is ARGB (all ARGB, RGB, RGBA are converted to ARGB32-8888 by the color space conversion before going to the filter is the color space conversion is done before the filter). When the register is not used by the HW, any value can be used for the bit-fields. Shadow register.	0255 7018h to 0255 701Ch	Section 11.3.5.3.4
20h	DISPC_VID1_ATTRIBUTES	The register configures the attributes of the video window. Shadow register.	0255 7020h	Section 11.3.5.3.5
24h	DISPC_VID1_ATTRIBUTES2	The register configures the attributes of the video window. Shadow register.	0255 7024h	Section 11.3.5.3.6
28h to 2Ch	DISPC_VID1_BA_0 to DISPC_VID1_BA_1	The register configures the base address of the video buffer for the video window (DISPC_VID1_BA__0 & DISPC_VID1_BA__1 for ping-pong mechanism with external trigger, based on the field polarity otherwise only DISPC_VID1_BA__0 is used). Shadow register.	0255 7028h to 0255 702Ch	Section 11.3.5.3.7
30h to 34h	DISPC_VID1_BA_UV_0 to DISPC_VID1_BA_UV_1	The register configures the base address of the UV buffer for the video window. (DISPC_VID1_BA_UV__0 & DISPC_VID1_BA_UV__1 for ping-pong mechanism with external trigger, based on the field polarity otherwise only DISPC_VID1_BA_UV__0 is used)). The register is also used to configure the RGB plane BA for RGB565A8 format Shadow register.	0255 7030h to 0255 7034h	Section 11.3.5.3.8
38h	DISPC_VID1_BUF_SIZE_STATUS	The register defines the Video buffer size for the video pipeline.	0255 7038h	Section 11.3.5.3.9
3Ch	DISPC_VID1_BUF_THRESHOLD	The register configures the video buffer associated with the video pipeline. Shadow register.	0255 703Ch	Section 11.3.5.3.10
40h	DISPC_VID1_CONV_COEF0	The register configures the color space conversion matrix coefficients for the video pipeline. Shadow register.	0255 7040h	Section 11.3.5.3.11
44h	DISPC_VID1_CONV_COEF1	The register configures the color space conversion matrix coefficients for the video pipeline. Shadow register.	0255 7044h	Section 11.3.5.3.12
48h	DISPC_VID1_CONV_COEF2	The register configures the color space conversion matrix coefficients for the video pipeline. Shadow register.	0255 7048h	Section 11.3.5.3.13

Table 11-492. DISPC_VID1 Registers (continued)

Offset	Acronym	Register Name	DISPC_VID1 Physical Address	Section
4Ch	DISPC_VID1_CONV_COEF3	The register configures the color space conversion matrix coefficients for the video pipeline. Shadow register.	0255 704Ch	Section 11.3.5.3.14
50h	DISPC_VID1_CONV_COEF4	The register configures the color space conversion matrix coefficients for the video pipeline. Shadow register.	0255 7050h	Section 11.3.5.3.15
54h	DISPC_VID1_CONV_COEF5	The register configures the color space conversion matrix coefficients for the video pipeline. Shadow register.	0255 7054h	Section 11.3.5.3.16
58h	DISPC_VID1_CONV_COEF6	The register configures the color space conversion matrix coefficients for the video pipeline. Shadow register.	0255 7058h	Section 11.3.5.3.17
5Ch	DISPC_VID1_FIRH	The register configures the resize factor for horizontal up/down-sampling of the video window. It is used for ARGB and Y setting. Shadow register.	0255 705Ch	Section 11.3.5.3.18
60h	DISPC_VID1_FIRH2	The register configures the resize factor for horizontal up/down-sampling of the video window. It is used for Crb and Cr setting. It is used only when the pixel format at the input of the filter is one of the YUV formats. If the pixel format at the input of the filter is ARGB (all ARGB, RGB, RGBA are converted to ARGB32-8888 by the color space conversion before going to the filter is the color space conversion is done before the filter). When the register is not used by the HW, any value can be used for the bit-fields. Shadow register.	0255 7060h	Section 11.3.5.3.19
64h	DISPC_VID1_FIRV	The register configures the resize factor for vertical up/down-sampling of the video window. It is used for ARGB and Y setting. Shadow register.	0255 7064h	Section 11.3.5.3.20
68h	DISPC_VID1_FIRV2	The register configures the resize factor for vertical up/down-sampling of the video window. It is used for Crb and Cr setting. It is used only when the pixel format at the input of the filter is one of the YUV formats. If the pixel format at the input of the filter is ARGB (all ARGB, RGB, RGBA are converted to ARGB32-8888 by the color space conversion before going to the filter is the color space conversion is done before the filter). When the register is not used by the HW, any value can be used for the bit-fields. Shadow register.	0255 7068h	Section 11.3.5.3.21
6Ch to 8Ch	DISPC_VID1_FIR_COEF_H0_0 to DISPC_VID1_FIR_COEF_H0_8	The bank of registers configure the up/down-scaling coefficients for the horizontal resize of the video picture associated with the video window for the 16 phases It is used for ARGB and Y setting. Shadow register.	0255 706Ch to 0255 708Ch	Section 11.3.5.3.22

Table 11-492. DISPC_VID1 Registers (continued)

Offset	Acronym	Register Name	DISPC_VID1 Physical Address	Section
90h to B0h	DISPC_VID1_FIR_COEF_H0_C_0 to DISPC_VID1_FIR_COEF_H0_C_8	The bank of registers configure the up/down-scaling coefficients for the horizontal resize of the video picture associated with the video window for the phases from 0 to 15. It is used for Crb and Cr setting. It is used only when the pixel format at the input of the filter is one of the YUV formats. If the pixel format at the input of the filter is ARGB (all ARGB, RGB, RGBA are converted to ARGB32-8888 by the color space conversion before going to the filter is the color space conversion is done before the filter). When the register is not used by the HW, any value can be used for the bit-fields. Shadow register.	0255 7090h to 0255 70B0h	Section 11.3.5.3.23
B4h to F0h	DISPC_VID1_FIR_COEF_H12_0 to DISPC_VID1_FIR_COEF_H12_15	The bank of registers configure the up/down-scaling coefficients for the horizontal resize of the video picture associated with the video window for the 16 phases It is used for ARGB and Y setting. Shadow register.	0255 70B4h to 0255 70F0h	Section 11.3.5.3.24
F4h to 130h	DISPC_VID1_FIR_COEF_H12_C_0 to DISPC_VID1_FIR_COEF_H12_C_15	The bank of registers configure the up/down-scaling coefficients for the horizontal resize of the video picture associated with the video window for the phases from 0 to 15. It is used for Crb and Cr setting. It is used only when the pixel format at the input of the filter is one of the YUV formats. If the pixel format at the input of the filter is ARGB (all ARGB, RGB, RGBA are converted to ARGB32-8888 by the color space conversion before going to the filter is the color space conversion is done before the filter). When the register is not used by the HW, any value can be used for the bit-fields. Shadow register.	0255 70F4h to 0255 7130h	Section 11.3.5.3.25
134h to 154h	DISPC_VID1_FIR_COEF_V0_0 to DISPC_VID1_FIR_COEF_V0_8	The bank of registers configure the up/down-scaling coefficients for the vertical resize of the video picture associated with the video window for the 16 phases It is used for ARGB and Y setting. Shadow register.	0255 7134h to 0255 7154h	Section 11.3.5.3.26
158h to 178h	DISPC_VID1_FIR_COEF_V0_C_0 to DISPC_VID1_FIR_COEF_V0_C_8	The bank of registers configure the up/down-scaling coefficients for the vertical resize of the video picture associated with the video window for the phases from 0 to 15. It is used for Crb and Cr setting. It is used only when the pixel format at the input of the filter is one of the YUV formats. If the pixel format at the input of the filter is ARGB (all ARGB, RGB, RGBA are converted to ARGB32-8888 by the color space conversion before going to the filter is the color space conversion is done before the filter). When the register is not used by the HW, any value can be used for the bit-fields. Shadow register.	0255 7158h to 0255 7178h	Section 11.3.5.3.27

Table 11-492. DISPC_VID1 Registers (continued)

Offset	Acronym	Register Name	DISPC_VID1 Physical Address	Section
17Ch to 1B8h	DISPC_VID1_FIR_COEF_V12_0 to DISPC_VID1_FIR_COEF_V12_15	The bank of registers configure the up/down-scaling coefficients for the vertical resize of the video picture associated with the video window for the 16 phases It is used for ARGB and Y setting. Shadow register.	0255 717Ch to 0255 71B8h	Section 11.3.5.3.28
1BCh to 1F8h	DISPC_VID1_FIR_COEF_V12_C_0 to DISPC_VID1_FIR_COEF_V12_C_15	The bank of registers configure the up/down-scaling coefficients for the vertical resize of the video picture associated with the video window for the phases from 0 to 15. It is used for Crb and Cr setting. It is used only when the pixel format at the input of the filter is one of the YUV formats. If the pixel format at the input of the filter is ARGB (all ARGB, RGB, RGBA are converted to ARGB32-8888 by the color space conversion before going to the filter is the color space conversion is done before the filter). When the register is not used by the HW, any value can be used for the bit-fields. Shadow register.	0255 71BCh to 0255 71F8h	Section 11.3.5.3.29
1FCh	DISPC_VID1_GLOBAL_ALPHA	The register defines the global alpha value for the video pipeline. Shadow register.	0255 71FCh	Section 11.3.5.3.30
200h	DISPC_VID1_IRQENABLE	This register allows to mask/unmask the module internal sources of interrupt, on an event-by-event basis	0255 7200h	Section 11.3.5.3.31
204h	DISPC_VID1_IRQSTATUS	This register regroups all the status of the module internal events that generate an interrupt. Write 1 to a given bit resets this bit	0255 7204h	Section 11.3.5.3.32
208h	DISPC_VID1_MFLAG_THRESHOLD	MFLAG thresholds for video pipelines. Shadow register.	0255 7208h	Section 11.3.5.3.33
20Ch	DISPC_VID1_PICTURE_SIZE	The register configures the size of the video picture associated with the video layer before up/down-scaling. Shadow register.	0255 720Ch	Section 11.3.5.3.34
210h	DISPC_VID1_PIXEL_INC	The register configures the number of bytes to increment between two pixels for the buffer associated with the video window. The register is used in order to perform low performance rotation. Shadow register.	0255 7210h	Section 11.3.5.3.35
214h	DISPC_VID1_POSITION	The register configures the position of the video window. Shadow register.	0255 7214h	Section 11.3.5.3.36
218h	DISPC_VID1_PRELOAD	The register configures the DMA buffer of the video pipeline. Shadow register.	0255 7218h	Section 11.3.5.3.37
21Ch	DISPC_VID1_ROW_INC	The register configures the number of bytes to increment at the end of the row for the buffer associated with the video window. Shadow register.	0255 721Ch	Section 11.3.5.3.38
220h	DISPC_VID1_SIZE	The register configures the size of the video window. Shadow register.	0255 7220h	Section 11.3.5.3.39

Table 11-492. DISPC_VID1 Registers (continued)

Offset	Acronym	Register Name	DISPC_VID1 Physical Address	Section
224h	DISPC_VID1_CLUT	The register configures the Color Look Up Table (CLUT) for VID pipeline. CLUT is used in conjunction with bitmap formats	0255 7224h	Section 11.3.5.3.40

11.3.5.3.1 DISPC_VID1_ACCUH_0 to DISPC_VID1_ACCUH_1 Register (Offset = 0h to 4h) [reset = 0h]

DISPC_VID1_ACCUH_0 to DISPC_VID1_ACCUH_1 is shown in Figure 11-234 and described in Table 11-494.

Return to Summary Table.

The register configures the resize accumulator init values for horizontal up/down-sampling of the video window (DISPC_VIDx_ACCU__0 DISPC_VIDx_ACCU__1 for ping-pong mechanism with external trigger, based on the field polarity) It is used for ARGB and Y setting. Shadow register.

Table 11-493. DISPC_VID1_ACCUH_0 to DISPC_VID1_ACCUH_1 Instances

Instance	Physical Address
DISPC_VID1	0255 7000h to 0255 7004h

Figure 11-234. DISPC_VID1_ACCUH_0 to DISPC_VID1_ACCUH_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								HORIZONTALACCU																							
R-0h								R/W-0h																							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-494. DISPC_VID1_ACCUH_0 to DISPC_VID1_ACCUH_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-0	HORIZONTALACCU	R/W	0h	Horizontal initialization accu signed value.

Table 11-495. Register Call Summary for DISPC_VID1_ACCUH_0

DSS Functional Description • DISPC VID1 Scaler Unit: [0][1][2]
DISPC_VID1 Registers • DISPC_VID1 Registers: [0] • DISPC_VID1_ACCUH_0 to DISPC_VID1_ACCUH_1 Register (Offset = 0h to 4h) [reset = 0h]: [0]

11.3.5.3.2 DISPC_VID1_ACCUH2_0 to DISPC_VID1_ACCUH2_1 Register (Offset = 8h to Ch) [reset = 0h]

DISPC_VID1_ACCUH2_0 to DISPC_VID1_ACCUH2_1 is shown in Figure 11-235 and described in Table 11-497.

Return to Summary Table.

The register configures the resize accumulator init value for horizontal up/down-sampling of the video window (DISPC_VID#n_ACCU2__0 DISPC_VID#n_ACCU2__1 for ping-pong mechanism with external trigger, based on the field polarity) It is used for Cb and Cr setting. It is used only when the pixel format at the input of the filter is one of the YUV formats. If the pixel format at the input of the filter is ARGB (all ARGB, RGB, RGBA are converted to ARGB32-8888 by the color space conversion before going to the filter is the color space conversion is done before the filter). When the register is not used by the HW, any value can be used for the bit-fields. Shadow register.

Table 11-496. DISPC_VID1_ACCUH2_0 to DISPC_VID1_ACCUH2_1 Instances

Instance	Physical Address
DISPC_VID1	0255 7008h to 0255 700Ch

Figure 11-235. DISPC_VID1_ACCUH2_0 to DISPC_VID1_ACCUH2_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								HORIZONTALACCU																							
R-0h								R/W-0h																							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-497. DISPC_VID1_ACCUH2_0 to DISPC_VID1_ACCUH2_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-0	HORIZONTALACCU	R/W	0h	Horizontal initialization accu signed value

Table 11-498. Register Call Summary for DISPC_VID1_ACCUH2_0

DSS Functional Description • DISPC VID1 Scaler Unit: [0][1]
DISPC_VID1 Registers • DISPC_VID1_ACCUH2_0 to DISPC_VID1_ACCUH2_1 Register (Offset = 8h to Ch) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.3 DISPC_VID1_ACCUV_0 to DISPC_VID1_ACCUV_1 Register (Offset = 10h to 14h) [reset = 0h]

DISPC_VID1_ACCUV_0 to DISPC_VID1_ACCUV_1 is shown in Figure 11-236 and described in Table 11-500.

Return to Summary Table.

The register configures the resize accumulator init values for horizontal and vertical up/down-sampling of the video window (DISPC_VIDx_ACCU__0 DISPC_VIDx_ACCU__1 for ping-pong mechanism with external trigger, based on the field polarity) It is used for ARGB and Y setting. Shadow register.

Table 11-499. DISPC_VID1_ACCUV_0 to DISPC_VID1_ACCUV_1 Instances

Instance	Physical Address
DISPC_VID1	0255 7010h to 0255 7014h

Figure 11-236. DISPC_VID1_ACCUV_0 to DISPC_VID1_ACCUV_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								VERTICALACCU																							
R-0h								R/W-0h																							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-500. DISPC_VID1_ACCUV_0 to DISPC_VID1_ACCUV_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-0	VERTICALACCU	R/W	0h	Vertical initialization accu signed value.

Table 11-501. Register Call Summary for DISPC_VID1_ACCUV_0

DSS Functional Description • DISPC VID1 Scaler Unit: [0][1][2]
DISPC_VID1 Registers • DISPC_VID1_ACCUV_0 to DISPC_VID1_ACCUV_1 Register (Offset = 10h to 14h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.4 DISPC_VID1_ACCUV2_0 to DISPC_VID1_ACCUV2_1 Register (Offset = 18h to 1Ch) [reset = 0h]

DISPC_VID1_ACCUV2_0 to DISPC_VID1_ACCUV2_1 is shown in Figure 11-237 and described in Table 11-503.

Return to Summary Table.

The register configures the resize accumulator init value for vertical up/down-sampling of the video window (DISPC_VID1_ACCU2__0 DISPC_VID1_ACCU2__1 for ping-pong mechanism with external trigger, based on the field polarity) It is used for Cb and Cr setting. It is used only when the pixel format at the input of the filter is one of the YUV formats. If the pixel format at the input of the filter is ARGB (all ARGB, RGB, RGBA are converted to ARGB32-8888 by the color space conversion before going to the filter is the color space conversion is done before the filter). When the register is not used by the HW, any value can be used for the bit-fields. Shadow register.

Table 11-502. DISPC_VID1_ACCUV2_0 to DISPC_VID1_ACCUV2_1 Instances

Instance	Physical Address
DISPC_VID1	0255 7018h to 0255 701Ch

Figure 11-237. DISPC_VID1_ACCUV2_0 to DISPC_VID1_ACCUV2_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								VERTICALACCU																							
R-0h								R/W-0h																							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-503. DISPC_VID1_ACCUV2_0 to DISPC_VID1_ACCUV2_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-0	VERTICALACCU	R/W	0h	Vertical initialization accu signed value.

Table 11-504. Register Call Summary for DISPC_VID1_ACCUV2_0

DSS Functional Description • DISPC VID1 Scaler Unit: [0][1]
DISPC_VID1 Registers • DISPC_VID1_ACCUV2_0 to DISPC_VID1_ACCUV2_1 Register (Offset = 18h to 1Ch) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.5 DISPC_VID1_ATTRIBUTES Register (Offset = 20h) [reset = 0h]

DISPC_VID1_ATTRIBUTES is shown in Figure 11-238 and described in Table 11-506.

Return to Summary Table.

The register configures the attributes of the video window. Shadow register.

Table 11-505. DISPC_VID1_ATTRIBUTES Instances

Instance	Physical Address
DISPC_VID1	0255 7020h

Figure 11-238. DISPC_VID1_ATTRIBUTES Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	PREMULTIPLYALPHA	ZORDER			SELFREFRESH
R-0h	R-0h	R-0h	R/W-0h	R/W-0h			R/W-0h
23	22	21	20	19	18	17	16
ARBITRATION	DOUBLESTRIDE	VERTICALTAPS	RESERVED	BUFPRELOAD	RESERVED	SELFREFRESHAUTO	CHANNELOUT
R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
CHANNELOUT		RESERVED		FULLRANGE	NIBBLEMODE	COLORCONVENABLE	RESIZEENABLE
R/W-0h		R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESIZEENABLE	FORMAT						ENABLE
R/W-0h	R/W-0h						R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-506. DISPC_VID1_ATTRIBUTES Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30	RESERVED	R	0h	Reserved
29	RESERVED	R	0h	Reserved
28	PREMULTIPLYALPHA	R/W	0h	The field configures the DISPC VID1 to process incoming data as premultiplied alpha data or non premultiplied alpha data. Default setting is non premultiplied alpha data. **Note: Feature is not supported in this family of devices.** 0h (R/W) = Non premultiplyalpha data color component 1h (R/W) = Premultiplyalpha data color component

Table 11-506. DISPC_VID1_ATTRIBUTES Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27-25	ZORDER	R/W	0h	Z-Order defining the priority of the layer compared to others when overlaying. It is SW responsibility to ensure that each layer connected to the same overlay manager has a different z-order value. **Note: Feature is not supported in this family of devices.** 0h (R/W) = Z-order 0: layer above solid background color and below layer with higher Z-order values. 1h (R/W) = Z-order 1: layer above layer with z-order value of 0 and below layers with z-order values of 2 and 3 2h (R/W) = Z-order 2: layer above layers with z-order value of 0 and 1 and below layer with z-order value of 3 3h (R/W) = Z-order 3: layer above layers with z-order value of 0, 1 and 2 and below layer with z-order value of 4 4h (R/W) = Z-order 4: layer above layers with z-order value of 0, 1, 2 and 3 and below layer with z-order value of 5 5h (R/W) = Z-order 5: layer above all the other layers except cursor
24	SELFREFRESH	R/W	0h	Enables the self refresh of the video window from its own DMA buffer only. 0h (R/W) = The video pipeline accesses the interconnect to fetch data from the system memory. 1h (R/W) = The video pipeline does not need anymore to fetch data from memory. Only the DMA buffer associated with the video1 is used. It takes effect after the frame has been loaded in the DMA buffer.
23	ARBITRATION	R/W	0h	Determines the priority of the video pipeline. The video pipeline is one of the high priority pipeline. The arbitration gives always the priority first to the high priority pipelines using round-robin between them. When there is only normal priority pipelines sending requests, the round-robin applies between them. **Note: Feature is not supported in this family of devices.** 0h (R/W) = The video pipeline is one of the normal priority pipeline. 1h (R/W) = The video pipeline is one of the high priority pipeline.
22	DOUBLESTRIDE	R/W	0h	Determines if the stride for CbCr buffer is the 1x or 2x of the Y buffer stride. It is only used in case of YUV420 and 2D access 0h (R/W) = The CbCr stride value is equal to the Y stride. 1h (R/W) = The CbCr stride value is double to the Y stride.
21	VERTICALTAPS	R/W	0h	Video Vertical Resize Tap Number. The vertial poly-phase filter can be configured in 3-tap or 5-tap configuration. According to the number of taps, the maximum input picture width is double while using 3-tap compared to 5-tap. 0h (R/W) = 3 taps are used for the vertical filtering logic. The 2 other taps are not used. The associated bit-fields for the 2 other taps coefficients do not need to be initialized. 1h (R/W) = 5 taps are used for the vertical filtering logic.
20	RESERVED	R	0h	Reserved
19	BUFPRELOAD	R/W	0h	Video Preload Value 0h (R/W) = H/W prefetches pixels up to the preload value defined in the preload register 1h (R/W) = H/W prefetches pixels up to high threshold value
18	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
17	SELFREFRESHAUTO	R/W	0h	Automatic self refresh mode 0h (R/W) = The transition from SELFREFRESH <disabled> to <enabled> is controlled by SW. 1h (R/W) = The transition from SELFREFRESH <disabled> to <enabled> is controlled only by HW.

Table 11-506. DISPC_VID1_ATTRIBUTES Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16-14	CHANNELOUT	R/W	0h	Video Channel Out configuration wr: immediate 0h (R/W) = OVR1 (VP1) Others = Reserved
13-12	RESERVED	R	0h	Reserved
11	FULLRANGE	R/W	0h	Color Space Conversion full range setting. 0h (R/W) = Limited range selected: 16 subtracted from Y before color space conversion 1h (R/W) = Full range selected: Y is not modified before the color space conversion
10	NIBBLEMODE	R/W	0h	Video Nibble mode (only for 1-, 2- and 4-bpp) 0h (R/W) = Nibble mode is disabled 1h (R/W) = Nibble mode is enabled
9	COLORCONVENABLE	R/W	0h	Enable the color space conversion. The HW does not enable/disable the conversion based on the pixel format. The bit-field shall be reset when the format is not YUV. 0h (R/W) = Disable Color Space Conversion YUV to RGB 1h (R/W) = Enable Color Space Conversion YUV to RGB
8-7	RESIZEENABLE	R/W	0h	Video Resize Enable 0h (R/W) = Disable both horizontal and vertical resize processing 1h (R/W) = Enable the horizontal resize processing 2h (R/W) = Enable the vertical resize processing 3h (R/W) = Enable both horizontal and vertical resize processing

Table 11-506. DISPC_VID1_ATTRIBUTES Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-1	FORMAT	R/W	0h	Video Format. It defines the pixel format when fetching the video frame buffer. 0h (R/W) = ARGB16-4444 1h (R/W) = ABGR16-4444 2h (R/W) = RGBA16-4444 3h (R/W) = RGB16-565 4h (R/W) = BGR16-565 5h (R/W) = ARGB16-1555 6h (R/W) = ABGR16-1555 7h (R/W) = ARGB32-8888 8h (R/W) = ABGR32-8888 9h (R/W) = RGBA32-8888 Ah (R/W) = BGRA32-8888 Bh (R/W) = RGB24-888 (24-bit container) Ch - Dh (R/W) = RESERVED Eh (R/W) = ARGB32-2101010 Fh (R/W) = ABGR32-2101010 10h (R/W) = ARGB64-16161616 11h (R/W) = RGBA64_16161616 12h (R/W) = BITMAP1 (CLUT is required) 13h (R/W) = BITMAP2 (CLUT is required) 14h (R/W) = BITMAP4 (CLUT is required) 15h (R/W) = BITMAP8 (CLUT is required) 16h (R/W) = RGB565A8 17h (R/W) = BGR565A8 18h - 1Fh (R/W) = RESERVED1 20h (R/W) = xRGB12-4444 21h (R/W) = xBGR16-4444 22h (R/W) = RGBx16-4444 23h (R/W) = RESERVED2 25h (R/W) = xRGB16-1555 26h (R/W) = xBGR16-1555 27h (R/W) = xRGB32-8888 (32-bit container) 28h (R/W) = xBGR32_8888 29h (R/W) = RGBx32-8888 (24-bit RGB aligned on MSB of the 32-bit container) 2Ah (R/W) = BGRX32_8888 2Bh - 2Dh (R/W) = RESERVED3 2Eh (R/W) = xRGB32-2101010 2Fh (R/W) = xBGR32-2101010 30h (R/W) = xRGB64-16161616 31h (R/W) = RGBX64_16161616 32h - 3Ch (R/W) = RESERVED4 3Dh (R/W) = NV12/N21 4:2:0 2 buffers (Y + UV) 3Eh (R/W) = YUV2 4:2:2 co-sited 3Fh (R/W) = UYVY 4:2:2 co-sited
0	ENABLE	R/W	0h	Video pipeline Enable 0h (R/W) = Video disabled (video pipeline inactive and window not present) 1h (R/W) = Video enabled (video pipeline active and window present on the screen)

Table 11-507. Register Call Summary for DISPC_VID1_ATTRIBUTES

DSS Functional Description • DISPC VID1 Color Look-Up Table (CLUT): [0] • DISPC Video Pipeline: [0] • DISPC DMA Addressing and Bursts: [0] • DISPC VID1 Scaler Unit: [0][2][3][4] • DISPC DMA Power Modes: [0][1] • DISPC Memory Formats: [0] • DISPC VID1 CSC Unit - YUV to RGB: [0][1][2]
DISPC_VID1 Registers • DISPC_VID1_ATTRIBUTES Register (Offset = 20h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.6 DISPC_VID1_ATTRIBUTES2 Register (Offset = 24h) [reset = 7C000000h]

DISPC_VID1_ATTRIBUTES2 is shown in Figure 11-239 and described in Table 11-509.

Return to Summary Table.

The register configures the attributes of the video window. Shadow register.

Table 11-508. DISPC_VID1_ATTRIBUTES2 Instances

Instance	Physical Address
DISPC_VID1	0255 7024h

Figure 11-239. DISPC_VID1_ATTRIBUTES2 Register

31	30	29	28	27	26	25	24
RESERVED	TAGS					RESERVED	REGION_BASED
R-0h	R/W-1Fh					R-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED							SECURE
R-0h							R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	VC1_RANGE_CBCR			VC1_RANGE_Y			VC1ENABLE
R-0h	R/W-0h			R/W-0h			R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-509. DISPC_VID1_ATTRIBUTES2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30-26	TAGS	R/W	1Fh	Number of OCP TAGS to be used for the pipeline (from 1 to 32)
25	RESERVED	R	0h	Reserved
24	REGION_BASED	R/W	0h	Enable region-based mechanism 0h (R/W) = DISABLE 1h (R/W) = ENABLE
23-17	RESERVED	R	0h	Reserved
16	SECURE	R/W	0h	OCP requests corresponds to pipeline data are secure/unsecure. The bit-field can be modified only by secure transaction using MReqSecure qualifier.
15-7	RESERVED	R	0h	Reserved
6-4	VC1_RANGE_CBCR	R/W	0h	Defines the VC1 range value for the CbCr component from 0 to 7.
3-1	VC1_RANGE_Y	R/W	0h	Defines the VC1 range value for the Y component from 0 to 7.
0	VC1ENABLE	R/W	0h	Enable/disable the VC1 range mapping processing. The bit-field is ignored if the format is not one of the supported YUV formats. 0h (R/W) = VC1 range mapping disabled 1h (R/W) = VC1 range mapping enabled

Table 11-510. Register Call Summary for DISPC_VID1_ATTRIBUTES2

DSS Functional Description
• DISPC VID1 VC-1 Range Mapping Unit: [0][1][2] • DISPC Region-Based Mechanism: [0]

Table 11-510. Register Call Summary for DISPC_VID1_ATTRIBUTES2 (continued)

DISPC_VID1 Registers
• DISPC_VID1_ATTRIBUTES2 Register (Offset = 24h) [reset = 7C000000h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.7 DISPC_VID1_BA_0 to DISPC_VID1_BA_1 Register (Offset = 28h to 2Ch) [reset = 0h]

DISPC_VID1_BA_0 to DISPC_VID1_BA_1 is shown in Figure 11-240 and described in Table 11-512.

Return to Summary Table.

The register configures the base address of the video buffer for the video window (DISPC_VID1_BA__0 and DISPC_VID1_BA__1 for ping-pong mechanism with external trigger, based on the field polarity. Otherwise only DISPC_VID1_BA__0 is used). Shadow register.

Table 11-511. DISPC_VID1_BA_0 to DISPC_VID1_BA_1 Instances

Instance	Physical Address
DISPC_VID1	0255 7028h to 0255 702Ch

Figure 11-240. DISPC_VID1_BA_0 to DISPC_VID1_BA_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BA																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-512. DISPC_VID1_BA_0 to DISPC_VID1_BA_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BA	R/W	0h	Video base address. When decompression is enabled bit[5-0] shall be set to 0. Base address of the video buffer (aligned on pixel size boundary except in case of RGB24 packed format, 4-pixel alignment is required; in case of YUV422, 2-pixel alignment is required, and YUV420, byte alignment is supported). In case of YUV 4:2:0 format, it indicates the base address of the Y buffer. Otherwise the bits indicate the corresponding bit address to access the SDRAM.

Table 11-513. Register Call Summary for DISPC_VID1_BA_0

DSS Functional Description • DISPC DMA Addressing and Bursts: [0][1][2][3]
DISPC_VID1 Registers • DISPC_VID1_BA_0 to DISPC_VID1_BA_1 Register (Offset = 28h to 2Ch) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.8 DISPC_VID1_BA_UV_0 to DISPC_VID1_BA_UV_1 Register (Offset = 30h to 34h) [reset = 0h]

DISPC_VID1_BA_UV_0 to DISPC_VID1_BA_UV_1 is shown in Figure 11-241 and described in Table 11-515.

Return to Summary Table.

The register configures the base address of the UV buffer for the video window. DISPC_VID1_BA_UV__0 and DISPC_VID1_BA_UV__1 for ping-pong mechanism with external trigger, based on the field polarity. Otherwise only DISPC_VID1_BA_UV__0 is used. The register is also used to configure the RGB plane BA for RGB565A8 format. Shadow register.

Table 11-514. DISPC_VID1_BA_UV_0 to DISPC_VID1_BA_UV_1 Instances

Instance	Physical Address
DISPC_VID1	0255 7030h to 0255 7034h

Figure 11-241. DISPC_VID1_BA_UV_0 to DISPC_VID1_BA_UV_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BA																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-515. DISPC_VID1_BA_UV_0 to DISPC_VID1_BA_UV_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BA	R/W	0h	Video base address alinged on 16-bit boundary. Base address of the UV video buffer used only in case of YUV420-NV12. Otherwise the bits indicated the corresponding bit address to access the SDRAM.

Table 11-516. Register Call Summary for DISPC_VID1_BA_UV_0

DSS Functional Description • DISPC DMA Addressing and Bursts: [0][1][2][3]
DISPC_VID1 Registers • DISPC_VID1_BA_UV_0 to DISPC_VID1_BA_UV_1 Register (Offset = 30h to 34h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.9 DISPC_VID1_BUF_SIZE_STATUS Register (Offset = 38h) [reset = A00h]

DISPC_VID1_BUF_SIZE_STATUS is shown in Figure 11-242 and described in Table 11-518.

Return to Summary Table.

The register defines the Video buffer size for the video pipeline.

Table 11-517. DISPC_VID1_BUF_SIZE_STATUS Instances

Instance	Physical Address
DISPC_VID1	0255 7038h

Figure 11-242. DISPC_VID1_BUF_SIZE_STATUS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																BUFSIZE															
R-0h																R-A00h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-518. DISPC_VID1_BUF_SIZE_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
15-0	BUFSIZE	R	A00h	Video DMA buffer Size in number of 128-bits

Table 11-519. Register Call Summary for DISPC_VID1_BUF_SIZE_STATUS

DISPC_VID1 Registers
- DISPC_VID1_BUF_SIZE_STATUS Register (Offset = 38h) [reset = A00h]: [0]
- DISPC_VID1 Registers: [0]

11.3.5.3.10 DISPC_VID1_BUF_THRESHOLD Register (Offset = 3Ch) [reset = 09FF09F8h]

DISPC_VID1_BUF_THRESHOLD is shown in Figure 11-243 and described in Table 11-521.

Return to Summary Table.

The register configures the video buffer associated with the video pipeline. Shadow register.

Table 11-520. DISPC_VID1_BUF_THRESHOLD Instances

Instance	Physical Address
DISPC_VID1	0255 703Ch

Figure 11-243. DISPC_VID1_BUF_THRESHOLD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUFHIGHTHRESHOLD																BUFLOWTHRESHOLD															
R/W-9FFh																R/W-9F8h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-521. DISPC_VID1_BUF_THRESHOLD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	BUFHIGHTHRESHOLD	R/W	9FFh	Video DMA buffer High Threshold. Number of 128-bits defining the threshold value.
15-0	BUFLOWTHRESHOLD	R/W	9F8h	DMA buffer High Threshold. Number of 128-bits defining the threshold value.

Table 11-522. Register Call Summary for DISPC_VID1_BUF_THRESHOLD

DSS Functional Description • DISPC Read DMA Buffer: [0][1][2][3] • DISPC DMA Power Modes: [0]
DISPC_VID1 Registers • DISPC_VID1_BUF_THRESHOLD Register (Offset = 3Ch) [reset = 09FF09F8h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.11 DISPC_VID1_CONV_COEF0 Register (Offset = 40h) [reset = 0h]

DISPC_VID1_CONV_COEF0 is shown in Figure 11-244 and described in Table 11-524.

Return to Summary Table.

The register configures the color space conversion matrix coefficients for the video pipeline. Shadow register.

Table 11-523. DISPC_VID1_CONV_COEF0 Instances

Instance	Physical Address
DISPC_VID1	0255 7040h

Figure 11-244. DISPC_VID1_CONV_COEF0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED					RCR										
R-0h					R/W-0h										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					RY										
R-0h					R/W-0h										

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-524. DISPC_VID1_CONV_COEF0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
26-16	RCR	R/W	0h	RCr Coefficient Encoded signed value (from -1024 to 1023).
15-11	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
10-0	RY	R/W	0h	RY Coefficient Encoded signed value (from -1024 to 1023).

Table 11-525. Register Call Summary for DISPC_VID1_CONV_COEF0

DSS Functional Description • DISPC VID1 CSC Unit - YUV to RGB: [0][1]
DISPC_VID1 Registers • DISPC_VID1_CONV_COEF0 Register (Offset = 40h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.12 DISPC_VID1_CONV_COEF1 Register (Offset = 44h) [reset = 0h]

DISPC_VID1_CONV_COEF1 is shown in Figure 11-245 and described in Table 11-527.

Return to Summary Table.

The register configures the color space conversion matrix coefficients for the video pipeline. Shadow register.

Table 11-526. DISPC_VID1_CONV_COEF1 Instances

Instance	Physical Address
DISPC_VID1	0255 7044h

Figure 11-245. DISPC_VID1_CONV_COEF1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED					GY										
R-0h					R/W-0h										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					RCB										
R-0h					R/W-0h										

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-527. DISPC_VID1_CONV_COEF1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
26-16	GY	R/W	0h	GY Coefficient Encoded signed value (from -1024 to 1023).
15-11	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
10-0	RCB	R/W	0h	RCb Coefficient Encoded signed value (from -1024 to 1023).

Table 11-528. Register Call Summary for DISPC_VID1_CONV_COEF1

DSS Functional Description • DISPC VID1 CSC Unit - YUV to RGB: [0][1]
DISPC_VID1 Registers • DISPC_VID1_CONV_COEF1 Register (Offset = 44h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.13 DISPC_VID1_CONV_COEF2 Register (Offset = 48h) [reset = 0h]

DISPC_VID1_CONV_COEF2 is shown in Figure 11-246 and described in Table 11-530.

Return to Summary Table.

The register configures the color space conversion matrix coefficients for the video pipeline. Shadow register.

Table 11-529. DISPC_VID1_CONV_COEF2 Instances

Instance	Physical Address
DISPC_VID1	0255 7048h

Figure 11-246. DISPC_VID1_CONV_COEF2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED					GCB										
R-0h					R/W-0h										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					GCR										
R-0h					R/W-0h										

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-530. DISPC_VID1_CONV_COEF2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
26-16	GCB	R/W	0h	GCb Coefficient Encoded signed value (from -1024 to 1023).
15-11	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
10-0	GCR	R/W	0h	GCr Coefficient Encoded signed value (from -1024 to 1023).

Table 11-531. Register Call Summary for DISPC_VID1_CONV_COEF2

DSS Functional Description • DISPC VID1 CSC Unit - YUV to RGB: [0][1]
DISPC_VID1 Registers • DISPC_VID1_CONV_COEF2 Register (Offset = 48h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.14 DISPC_VID1_CONV_COEF3 Register (Offset = 4Ch) [reset = 0h]

DISPC_VID1_CONV_COEF3 is shown in Figure 11-247 and described in Table 11-533.

Return to Summary Table.

The register configures the color space conversion matrix coefficients for the video pipeline. Shadow register.

Table 11-532. DISPC_VID1_CONV_COEF3 Instances

Instance	Physical Address
DISPC_VID1	0255 704Ch

Figure 11-247. DISPC_VID1_CONV_COEF3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED					BCR										
R-0h					R/W-0h										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					BY										
R-0h					R/W-0h										

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-533. DISPC_VID1_CONV_COEF3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
26-16	BCR	R/W	0h	BCr coefficient Encoded signed value (from -1024 to 1023).
15-11	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
10-0	BY	R/W	0h	BY coefficient Encoded signed value (from -1024 to 1023).

Table 11-534. Register Call Summary for DISPC_VID1_CONV_COEF3

DSS Functional Description • DISPC VID1 CSC Unit - YUV to RGB: [0][1]
DISPC_VID1 Registers • DISPC_VID1_CONV_COEF3 Register (Offset = 4Ch) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.15 DISPC_VID1_CONV_COEF4 Register (Offset = 50h) [reset = 0h]

DISPC_VID1_CONV_COEF4 is shown in Figure 11-248 and described in Table 11-536.

Return to Summary Table.

The register configures the color space conversion matrix coefficients for the video pipeline. Shadow register.

Table 11-535. DISPC_VID1_CONV_COEF4 Instances

Instance	Physical Address
DISPC_VID1	0255 7050h

Figure 11-248. DISPC_VID1_CONV_COEF4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					BCB										
R-0h																					R/W-0h										

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-536. DISPC_VID1_CONV_COEF4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
10-0	BCB	R/W	0h	BCb Coefficient Encoded signed value (from -1024 to 1023).

Table 11-537. Register Call Summary for DISPC_VID1_CONV_COEF4

DSS Functional Description • DISPC VID1 CSC Unit - YUV to RGB: [0]
DISPC_VID1 Registers • DISPC_VID1_CONV_COEF4 Register (Offset = 50h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.16 DISPC_VID1_CONV_COEF5 Register (Offset = 54h) [reset = 0h]

DISPC_VID1_CONV_COEF5 is shown in Figure 11-249 and described in Table 11-539.

Return to Summary Table.

The register configures the color space conversion matrix coefficients for the video pipeline. Shadow register.

Table 11-538. DISPC_VID1_CONV_COEF5 Instances

Instance	Physical Address
DISPC_VID1	0255 7054h

Figure 11-249. DISPC_VID1_CONV_COEF5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GOFFSET													RESERVED		
R/W-0h													R-0h		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROFFSET													RESERVED		
R/W-0h													R-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-539. DISPC_VID1_CONV_COEF5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	GOFFSET	R/W	0h	G offset Encoded signed value (from -4096 to 4095).
18-16	RESERVED	R	0h	Reserved
15-3	ROFFSET	R/W	0h	R offset Encoded signed value (from -4096 to 4095).
2-0	RESERVED	R	0h	Reserved

Table 11-540. Register Call Summary for DISPC_VID1_CONV_COEF5

DSS Functional Description • DISPC VID1 CSC Unit - YUV to RGB: [0][1][2][3]
DISPC_VID1 Registers • DISPC_VID1_CONV_COEF5 Register (Offset = 54h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.17 DISPC_VID1_CONV_COEF6 Register (Offset = 58h) [reset = 0h]

DISPC_VID1_CONV_COEF6 is shown in Figure 11-250 and described in Table 11-542.

Return to Summary Table.

The register configures the color space conversion matrix coefficients for the video pipeline. Shadow register.

Table 11-541. DISPC_VID1_CONV_COEF6 Instances

Instance	Physical Address
DISPC_VID1	0255 7058h

Figure 11-250. DISPC_VID1_CONV_COEF6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOFFSET													RESERVED		
R/W-0h													R-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-542. DISPC_VID1_CONV_COEF6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-3	BOFFSET	R/W	0h	B offset Encoded signed value (from -4096 to 4095).
2-0	RESERVED	R	0h	Reserved

Table 11-543. Register Call Summary for DISPC_VID1_CONV_COEF6

DSS Functional Description • DISPC VID1 CSC Unit - YUV to RGB: [0][1][2]
DISPC_VID1 Registers • DISPC_VID1_CONV_COEF6 Register (Offset = 58h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.18 DISPC_VID1_FIRH Register (Offset = 5Ch) [reset = 00200000h]

DISPC_VID1_FIRH is shown in Figure 11-251 and described in Table 11-545.

Return to Summary Table.

The register configures the resize factor for horizontal up/down-sampling of the video window. It is used for ARGB and Y setting. Shadow register.

Table 11-544. DISPC_VID1_FIRH Instances

Instance	Physical Address
DISPC_VID1	0255 705Ch

Figure 11-251. DISPC_VID1_FIRH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FIRHINC																							
R-0h								R/W-00200000h																							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-545. DISPC_VID1_FIRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-0	FIRHINC	R/W	00200000h	Horizontal increment of the up/down-sampling filter. The value 0 is invalid.

Table 11-546. Register Call Summary for DISPC_VID1_FIRH

DSS Functional Description • DISPC VID1 Scaler Unit: [0]
DISPC_VID1 Registers • DISPC_VID1_FIRH Register (Offset = 5Ch) [reset = 00200000h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.19 DISPC_VID1_FIRH2 Register (Offset = 60h) [reset = 00200000h]

DISPC_VID1_FIRH2 is shown in Figure 11-252 and described in Table 11-548.

Return to Summary Table.

The register configures the resize factor for horizontal up/down-sampling of the video window. It is used for Crb and Cr setting. It is used only when the pixel format at the input of the filter is one of the YUV formats. If the pixel format at the input of the filter is ARGB (all ARGB, RGB, RGBA are converted to ARGB32-8888 by the color space conversion before going to the filter is the color space conversion is done before the filter). When the register is not used by the HW, any value can be used for the bit-fields. Shadow register.

Table 11-547. DISPC_VID1_FIRH2 Instances

Instance	Physical Address
DISPC_VID1	0255 7060h

Figure 11-252. DISPC_VID1_FIRH2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FIRHINC																							
R-0h								R/W-00200000h																							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-548. DISPC_VID1_FIRH2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-0	FIRHINC	R/W	00200000h	Horizontal increment of the up/down-sampling filter for Cb and Cr. The value 0 is invalid.

Table 11-549. Register Call Summary for DISPC_VID1_FIRH2

DSS Functional Description • DISPC VID1 Scaler Unit: [0]
DISPC_VID1 Registers • DISPC_VID1_FIRH2 Register (Offset = 60h) [reset = 00200000h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.20 DISPC_VID1_FIRV Register (Offset = 64h) [reset = 00200000h]

DISPC_VID1_FIRV is shown in Figure 11-253 and described in Table 11-551.

Return to Summary Table.

The register configures the resize factor for vertical up/down-sampling of the video window. It is used for ARGB and Y setting. Shadow register.

Table 11-550. DISPC_VID1_FIRV Instances

Instance	Physical Address
DISPC_VID1	0255 7064h

Figure 11-253. DISPC_VID1_FIRV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FIRVINC																							
R-0h								R/W-00200000h																							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-551. DISPC_VID1_FIRV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-0	FIRVINC	R/W	00200000h	Vertical increment of the up/down-sampling filter. The value 0 is invalid.

Table 11-552. Register Call Summary for DISPC_VID1_FIRV

DSS Functional Description • DISPC VID1 Scaler Unit: [0]
DISPC_VID1 Registers • DISPC_VID1_FIRV Register (Offset = 64h) [reset = 00200000h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.21 DISPC_VID1_FIRV2 Register (Offset = 68h) [reset = 00200000h]

DISPC_VID1_FIRV2 is shown in Figure 11-254 and described in Table 11-554.

Return to Summary Table.

The register configures the resize factor for vertical up/down-sampling of the video window. It is used for Crb and Cr setting. It is used only when the pixel format at the input of the filter is one of the YUV formats. If the pixel format at the input of the filter is ARGB (all ARGB, RGB, RGBA are converted to ARGB32-8888 by the color space conversion before going to the filter is the color space conversion is done before the filter). When the register is not used by the HW, any value can be used for the bit-fields. Shadow register.

Table 11-553. DISPC_VID1_FIRV2 Instances

Instance	Physical Address
DISPC_VID1	0255 7068h

Figure 11-254. DISPC_VID1_FIRV2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FIRVINC																							
R-0h								R/W-00200000h																							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-554. DISPC_VID1_FIRV2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-0	FIRVINC	R/W	00200000h	Vertical increment of the up/down-sampling filter for Cb and Cr. The value 0 is invalid.

Table 11-555. Register Call Summary for DISPC_VID1_FIRV2

DSS Functional Description • DISPC VID1 Scaler Unit: [0]
DISPC_VID1 Registers • DISPC_VID1_FIRV2 Register (Offset = 68h) [reset = 00200000h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.22 DISPC_VID1_FIR_COEF_H0_0 to DISPC_VID1_FIR_COEF_H0_8 Register (Offset = 6Ch to 8Ch) [reset = 0h]

DISPC_VID1_FIR_COEF_H0_0 to DISPC_VID1_FIR_COEF_H0_8 is shown in Figure 11-255 and described in Table 11-557.

Return to Summary Table.

The bank of registers configure the up/down-scaling coefficients for the horizontal resize of the video picture associated with the video window for the 16 phases It is used for ARGB and Y setting. Shadow register.

Table 11-556. DISPC_VID1_FIR_COEF_H0_0 to DISPC_VID1_FIR_COEF_H0_8 Instances

Instance	Physical Address
DISPC_VID1	0255 706Ch to 0255 708Ch

Figure 11-255. DISPC_VID1_FIR_COEF_H0_0 to DISPC_VID1_FIR_COEF_H0_8 Register

31	30	29	28	27	26	25	24
RESERVED		RESERVED					
R-0h		R-0h					
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						FIRHC0	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
FIRHC0							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-557. DISPC_VID1_FIR_COEF_H0_0 to DISPC_VID1_FIR_COEF_H0_8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29-10	RESERVED	R	0h	Reserved
9-0	FIRHC0	R/W	0h	Unsigned coefficient C0 for the horizontal up/down-scaling with the phase n

Table 11-558. Register Call Summary for DISPC_VID1_FIR_COEF_H0_0

DSS Functional Description • DISPC VID1 Scaler Unit: [0][1][2]
DISPC_VID1 Registers • DISPC_VID1_FIR_COEF_H0_0 to DISPC_VID1_FIR_COEF_H0_8 Register (Offset = 6Ch to 8Ch) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.23 DISPC_VID1_FIR_COEF_H0_C_0 to DISPC_VID1_FIR_COEF_H0_C_8 Register (Offset = 90h to B0h) [reset = 0h]

DISPC_VID1_FIR_COEF_H0_C_0 to DISPC_VID1_FIR_COEF_H0_C_8 is shown in Figure 11-256 and described in Table 11-560.

Return to Summary Table.

The bank of registers configure the up/down-scaling coefficients for the horizontal resize of the video picture associated with the video window for the phases from 0 to 15. It is used for Crb and Cr setting. It is used only when the pixel format at the input of the filter is one of the YUV formats. If the pixel format at the input of the filter is ARGB (all ARGB, RGB, RGBA are converted to ARGB32-8888 by the color space conversion before going to the filter is the color space conversion is done before the filter). When the register is not used by the HW, any value can be used for the bit-fields. Shadow register.

Table 11-559. DISPC_VID1_FIR_COEF_H0_C_0 to DISPC_VID1_FIR_COEF_H0_C_8 Instances

Instance	Physical Address
DISPC_VID1	0255 7090h to 0255 70B0h

Figure 11-256. DISPC_VID1_FIR_COEF_H0_C_0 to DISPC_VID1_FIR_COEF_H0_C_8 Register

31	30	29	28	27	26	25	24
RESERVED		RESERVED					
R-0h		R-0h					
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						FIRHC0	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
FIRHC0							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-560. DISPC_VID1_FIR_COEF_H0_C_0 to DISPC_VID1_FIR_COEF_H0_C_8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29-10	RESERVED	R	0h	Reserved
9-0	FIRHC0	R/W	0h	Unsigned coefficient C0 for the horizontal up/down-scaling with the phase n

Table 11-561. Register Call Summary for DISPC_VID1_FIR_COEF_H0_C_0

DSS Functional Description • DISPC VID1 Scaler Unit: [0][1]
DISPC_VID1 Registers • DISPC_VID1_FIR_COEF_H0_C_0 to DISPC_VID1_FIR_COEF_H0_C_8 Register (Offset = 90h to B0h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.24 DISPC_VID1_FIR_COEF_H12_0 to DISPC_VID1_FIR_COEF_H12_15 Register (Offset = B4h to F0h) [reset = 0h]

DISPC_VID1_FIR_COEF_H12_0 to DISPC_VID1_FIR_COEF_H12_15 is shown in Figure 11-257 and described in Table 11-563.

Return to Summary Table.

The bank of registers configure the up/down-scaling coefficients for the horizontal resize of the video picture associated with the video window for the 16 phases It is used for ARGB and Y setting. Shadow register.

Table 11-562. DISPC_VID1_FIR_COEF_H12_0 to DISPC_VID1_FIR_COEF_H12_15 Instances

Instance	Physical Address
DISPC_VID1	0255 70B4h to 0255 70F0h

Figure 11-257. DISPC_VID1_FIR_COEF_H12_0 to DISPC_VID1_FIR_COEF_H12_15 Register

31	30	29	28	27	26	25	24
RESERVED		FIRHC2					
R-0h		R/W-0h					
23	22	21	20	19	18	17	16
FIRHC2				FIRHC1			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
FIRHC1						RESERVED	
R/W-0h						R-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-563. DISPC_VID1_FIR_COEF_H12_0 to DISPC_VID1_FIR_COEF_H12_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29-20	FIRHC2	R/W	0h	Signed coefficient C2 for the horizontal up/down-scaling with the phase n
19-10	FIRHC1	R/W	0h	Signed coefficient C1 for the horizontal up/down-scaling with the phase n
9-0	RESERVED	R	0h	Reserved

Table 11-564. Register Call Summary for DISPC_VID1_FIR_COEF_H12_0

DSS Functional Description • DISPC VID1 Scaler Unit: [0][1][2][3][4][5]
DISPC_VID1 Registers • DISPC_VID1_FIR_COEF_H12_0 to DISPC_VID1_FIR_COEF_H12_15 Register (Offset = B4h to F0h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.25 DISPC_VID1_FIR_COEF_H12_C_0 to DISPC_VID1_FIR_COEF_H12_C_15 Register (Offset = F4h to 130h) [reset = 0h]

DISPC_VID1_FIR_COEF_H12_C_0 to DISPC_VID1_FIR_COEF_H12_C_15 is shown in Figure 11-258 and described in Table 11-566.

Return to Summary Table.

The bank of registers configure the up/down-scaling coefficients for the horizontal resize of the video picture associated with the video window for the phases from 0 to 15. It is used for Crb and Cr setting. It is used only when the pixel format at the input of the filter is one of the YUV formats. If the pixel format at the input of the filter is ARGB (all ARGB, RGB, RGBA are converted to ARGB32-8888 by the color space conversion before going to the filter is the color space conversion is done before the filter). When the register is not used by the HW, any value can be used for the bit-fields. Shadow register.

Table 11-565. DISPC_VID1_FIR_COEF_H12_C_0 to DISPC_VID1_FIR_COEF_H12_C_15 Instances

Instance	Physical Address
DISPC_VID1	0255 70F4h to 0255 7130h

Figure 11-258. DISPC_VID1_FIR_COEF_H12_C_0 to DISPC_VID1_FIR_COEF_H12_C_15 Register

31	30	29	28	27	26	25	24
RESERVED		FIRHC2					
R-0h		R/W-0h					
23	22	21	20	19	18	17	16
FIRHC2				FIRHC1			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
FIRHC1						RESERVED	
R/W-0h						R-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-566. DISPC_VID1_FIR_COEF_H12_C_0 to DISPC_VID1_FIR_COEF_H12_C_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29-20	FIRHC2	R/W	0h	Signed coefficient C2 for the horizontal up/down-scaling with the phase n
19-10	FIRHC1	R/W	0h	Signed coefficient C1 for the horizontal up/down-scaling with the phase n
9-0	RESERVED	R	0h	Reserved

Table 11-567. Register Call Summary for DISPC_VID1_FIR_COEF_H12_C_0

DSS Functional Description • DISPC VID1 Scaler Unit: [0][1]
DISPC_VID1 Registers • DISPC_VID1_FIR_COEF_H12_C_0 to DISPC_VID1_FIR_COEF_H12_C_15 Register (Offset = F4h to 130h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.26 DISPC_VID1_FIR_COEF_V0_0 to DISPC_VID1_FIR_COEF_V0_8 Register (Offset = 134h to 154h) [reset = 0h]

DISPC_VID1_FIR_COEF_V0_0 to DISPC_VID1_FIR_COEF_V0_8 is shown in Figure 11-259 and described in Table 11-569.

Return to Summary Table.

The bank of registers configure the up/down-scaling coefficients for the vertical resize of the video picture associated with the video window for the 16 phases It is used for ARGB and Y setting. Shadow register.

Table 11-568. DISPC_VID1_FIR_COEF_V0_0 to DISPC_VID1_FIR_COEF_V0_8 Instances

Instance	Physical Address
DISPC_VID1	0255 7134h to 0255 7154h

Figure 11-259. DISPC_VID1_FIR_COEF_V0_0 to DISPC_VID1_FIR_COEF_V0_8 Register

31	30	29	28	27	26	25	24
RESERVED		RESERVED					
R-0h		R-0h					
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						FIRVC0	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
FIRVC0							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-569. DISPC_VID1_FIR_COEF_V0_0 to DISPC_VID1_FIR_COEF_V0_8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29-10	RESERVED	R	0h	Reserved
9-0	FIRVC0	R/W	0h	Unsigned coefficient C0 for the vertical up/down-scaling with the phase n

Table 11-570. Register Call Summary for DISPC_VID1_FIR_COEF_V0_0

DSS Functional Description • DISPC VID1 Scaler Unit: [0][1]
DISPC_VID1 Registers • DISPC_VID1_FIR_COEF_V0_0 to DISPC_VID1_FIR_COEF_V0_8 Register (Offset = 134h to 154h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.27 DISPC_VID1_FIR_COEF_V0_C_0 to DISPC_VID1_FIR_COEF_V0_C_8 Register (Offset = 158h to 178h) [reset = 0h]

DISPC_VID1_FIR_COEF_V0_C_0 to DISPC_VID1_FIR_COEF_V0_C_8 is shown in Figure 11-260 and described in Table 11-572.

Return to Summary Table.

The bank of registers configure the up/down-scaling coefficients for the vertical resize of the video picture associated with the video window for the phases from 0 to 15. It is used for Crb and Cr setting. It is used only when the pixel format at the input of the filter is one of the YUV formats. If the pixel format at the input of the filter is ARGB (all ARGB, RGB, RGBA are converted to ARGB32-8888 by the color space conversion before going to the filter is the color space conversion is done before the filter). When the register is not used by the HW, any value can be used for the bit-fields. Shadow register.

Table 11-571. DISPC_VID1_FIR_COEF_V0_C_0 to DISPC_VID1_FIR_COEF_V0_C_8 Instances

Instance	Physical Address
DISPC_VID1	0255 7158h to 0255 7178h

Figure 11-260. DISPC_VID1_FIR_COEF_V0_C_0 to DISPC_VID1_FIR_COEF_V0_C_8 Register

31	30	29	28	27	26	25	24
RESERVED		RESERVED					
R-0h		R-0h					
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						FIRVC0	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
FIRVC0							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-572. DISPC_VID1_FIR_COEF_V0_C_0 to DISPC_VID1_FIR_COEF_V0_C_8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29-10	RESERVED	R	0h	Reserved
9-0	FIRVC0	R/W	0h	Unsigned coefficient C0 for the vertical up/down-scaling with the phase n

Table 11-573. Register Call Summary for DISPC_VID1_FIR_COEF_V0_C_0

DSS Functional Description • DISPC VID1 Scaler Unit: [0][1]
DISPC_VID1 Registers • DISPC_VID1_FIR_COEF_V0_C_0 to DISPC_VID1_FIR_COEF_V0_C_8 Register (Offset = 158h to 178h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.28 DISPC_VID1_FIR_COEF_V12_0 to DISPC_VID1_FIR_COEF_V12_15 Register (Offset = 17Ch to 1B8h) [reset = 0h]

DISPC_VID1_FIR_COEF_V12_0 to DISPC_VID1_FIR_COEF_V12_15 is shown in Figure 11-261 and described in Table 11-575.

Return to Summary Table.

The bank of registers configure the up/down-scaling coefficients for the vertical resize of the video picture associated with the video window for the 16 phases It is used for ARGB and Y setting. Shadow register.

Table 11-574. DISPC_VID1_FIR_COEF_V12_0 to DISPC_VID1_FIR_COEF_V12_15 Instances

Instance	Physical Address
DISPC_VID1	0255 717Ch to 0255 71B8h

Figure 11-261. DISPC_VID1_FIR_COEF_V12_0 to DISPC_VID1_FIR_COEF_V12_15 Register

31	30	29	28	27	26	25	24
RESERVED		FIRVC2					
R-0h		R/W-0h					
23	22	21	20	19	18	17	16
FIRVC2				FIRVC1			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
FIRVC1						RESERVED	
R/W-0h						R-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-575. DISPC_VID1_FIR_COEF_V12_0 to DISPC_VID1_FIR_COEF_V12_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29-20	FIRVC2	R/W	0h	Signed coefficient C2 for the vertical up/down-scaling with the phase n
19-10	FIRVC1	R/W	0h	Signed coefficient C1 for the vertical up/down-scaling with the phase n
9-0	RESERVED	R	0h	Reserved

Table 11-576. Register Call Summary for DISPC_VID1_FIR_COEF_V12_0

DSS Functional Description • DISPC VID1 Scaler Unit: [0][1]
DISPC_VID1 Registers • DISPC_VID1_FIR_COEF_V12_0 to DISPC_VID1_FIR_COEF_V12_15 Register (Offset = 17Ch to 1B8h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.29 DISPC_VID1_FIR_COEF_V12_C_0 to DISPC_VID1_FIR_COEF_V12_C_15 Register (Offset = 1BCh to 1F8h) [reset = 0h]

DISPC_VID1_FIR_COEF_V12_C_0 to DISPC_VID1_FIR_COEF_V12_C_15 is shown in Figure 11-262 and described in Table 11-578.

Return to Summary Table.

The bank of registers configure the up/down-scaling coefficients for the vertical resize of the video picture associated with the video window for the phases from 0 to 15. It is used for Crb and Cr setting. It is used only when the pixel format at the input of the filter is one of the YUV formats. If the pixel format at the input of the filter is ARGB (all ARGB, RGB, RGBA are converted to ARGB32-8888 by the color space conversion before going to the filter is the color space conversion is done before the filter). When the register is not used by the HW, any value can be used for the bit-fields. Shadow register.

Table 11-577. DISPC_VID1_FIR_COEF_V12_C_0 to DISPC_VID1_FIR_COEF_V12_C_15 Instances

Instance	Physical Address
DISPC_VID1	0255 71BCh to 0255 71F8h

Figure 11-262. DISPC_VID1_FIR_COEF_V12_C_0 to DISPC_VID1_FIR_COEF_V12_C_15 Register

31	30	29	28	27	26	25	24
RESERVED		FIRVC2					
R-0h		R/W-0h					
23	22	21	20	19	18	17	16
FIRVC2				FIRVC1			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
FIRVC1						RESERVED	
R/W-0h						R-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-578. DISPC_VID1_FIR_COEF_V12_C_0 to DISPC_VID1_FIR_COEF_V12_C_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29-20	FIRVC2	R/W	0h	Signed coefficient C2 for the vertical up/down-scaling with the phase n
19-10	FIRVC1	R/W	0h	Signed coefficient C1 for the vertical up/down-scaling with the phase n
9-0	RESERVED	R	0h	Reserved

Table 11-579. Register Call Summary for DISPC_VID1_FIR_COEF_V12_C_0

DSS Functional Description • DISPC VID1 Scaler Unit: [0][1]
DISPC_VID1 Registers • DISPC_VID1_FIR_COEF_V12_C_0 to DISPC_VID1_FIR_COEF_V12_C_15 Register (Offset = 1BCh to 1F8h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.30 DISPC_VID1_GLOBAL_ALPHA Register (Offset = 1FCh) [reset = FFh]

DISPC_VID1_GLOBAL_ALPHA is shown in Figure 11-263 and described in Table 11-581.

Return to Summary Table.

The register defines the global alpha value for the video pipeline. Shadow register.

Table 11-580. DISPC_VID1_GLOBAL_ALPHA Instances

Instance	Physical Address
DISPC_VID1	0255 71FCh

Figure 11-263. DISPC_VID1_GLOBAL_ALPHA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								GLOBALALPHA							
R-0h								R/W-FFh							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-581. DISPC_VID1_GLOBAL_ALPHA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	GLOBALALPHA	R/W	FFh	Global alpha value from 0 to 255. 0 corresponds to fully transparent and 255 to fully opaque.

Table 11-582. Register Call Summary for DISPC_VID1_GLOBAL_ALPHA

DISPC_VID1 Registers
- DISPC_VID1_GLOBAL_ALPHA Register (Offset = 1FCh) [reset = FFh]: [0]
- DISPC_VID1 Registers: [0]

11.3.5.3.31 DISPC_VID1_IRQENABLE Register (Offset = 200h) [reset = 0h]

DISPC_VID1_IRQENABLE is shown in Figure 11-264 and described in Table 11-584.

Return to Summary Table.

This register allows to mask/unmask the module internal sources of interrupt, on an event-by-event basis

Table 11-583. DISPC_VID1_IRQENABLE Instances

Instance	Physical Address
DISPC_VID1	0255 7200h

Figure 11-264. DISPC_VID1_IRQENABLE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				VIDREGIONBASEDPIPEEND_EN	VIDREGIONBASEDPIPESTART_EN	VIDENDWINDOW_EN	VIDBUFFERUNDERFLOW_EN
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-584. DISPC_VID1_IRQENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	VIDREGIONBASEDPIPEEND_EN	R/W	0h	PIPE end window IRQ for region-based feature 0h (R/W) = VIDREGIONBASEDPIPEEND is masked 1h (R/W) = VIDREGIONBASEDPIPEEND generates an interrupt when it occurs
2	VIDREGIONBASEDPIPESTART_EN	R/W	0h	PIPE start window IRQ for region-based feature 0h (R/W) = VIDREGIONBASEDPIPESTART is masked 1h (R/W) = VIDREGIONBASEDPIPESTART generates an interrupt when it occurs
1	VIDENDWINDOW_EN	R/W	0h	The end of the video Window has been reached. It is detected by the overlay manager when the full video has been displayed. 0h (R/W) = EndVid1Window is masked 1h (R/W) = EndVid1Window generates an interrupt when it occurs
0	VIDBUFFERUNDERFLOW_EN	R/W	0h	Video DMA Buffer Underflow. The DMA buffer is not necessary empty but required data are not present in the DMA buffer (due to out of order responses) 0h (R/W) = Vid1BufferUnderflow is masked 1h (R/W) = Vid1BufferUnderflow generates an interrupt when it occurs

Table 11-585. Register Call Summary for DISPC_VID1_IRQENABLE

DSS Functional Description • DISPC Interrupt Requests: [0]
DISPC_VID1 Registers • DISPC_VID1_IRQENABLE Register (Offset = 200h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.32 DISPC_VID1_IRQSTATUS Register (Offset = 204h) [reset = 0h]

DISPC_VID1_IRQSTATUS is shown in Figure 11-265 and described in Table 11-587.

Return to Summary Table.

This register regroups all the status of the module internal events that generate an interrupt. Write 1 to a given bit resets this bit

Table 11-586. DISPC_VID1_IRQSTATUS Instances

Instance	Physical Address
DISPC_VID1	0255 7204h

Figure 11-265. DISPC_VID1_IRQSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				VIDREGIONBASEDPIPEEND_IRQ	VIDREGIONBASEDPIPESTART_IRQ	VIDENDWINDOW_IRQ	VIDBUFFERUNDERFLOW_IRQ
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-587. DISPC_VID1_IRQSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	VIDREGIONBASEDPIPEEND_IRQ	R/W	0h	PIPE end window IRQ for region-based feature 0h (R/W) = READS: Event is false. WRITES: Status bit unchanged. 1h (R/W) = READS: Event is true (pending). WRITES: Status bit is reset.
2	VIDREGIONBASEDPIPESTART_IRQ	R/W	0h	PIPE start window IRQ for region-based feature 0h (R/W) = READS: Event is false. WRITES: Status bit unchanged. 1h (R/W) = READS: Event is true (pending). WRITES: Status bit is reset.
1	VIDENDWINDOW_IRQ	R/W	0h	The end of the video Window has been reached. It is detected by the overlay manager when the full video has been displayed. 0h (R/W) = READS: Event is false. WRITES: Status bit unchanged. 1h (R/W) = READS: Event is true (pending). WRITES: Status bit is reset.
0	VIDBUFFERUNDERFLOW_IRQ	R/W	0h	Video DMA Buffer Underflow. The DMA buffer is not necessarly empty but required data are not present in the DMA buffer (due to out of order responses) 0h (R/W) = READS: Event is false. WRITES: Status bit unchanged. 1h (R/W) = READS: Event is true (pending). WRITES: Status bit is reset.

Table 11-588. Register Call Summary for DISPC_VID1_IRQSTATUS

DSS Functional Description • DISPC Interrupt Requests: [0]
DISPC_VID1 Registers • DISPC_VID1_IRQSTATUS Register (Offset = 204h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.33 DISPC_VID1_MFLAG_THRESHOLD Register (Offset = 208h) [reset = 0h]

DISPC_VID1_MFLAG_THRESHOLD is shown in Figure 11-266 and described in Table 11-590.

Return to Summary Table.

MFLAG thresholds for video pipelines. Shadow register.

Table 11-589. DISPC_VID1_MFLAG_THRESHOLD Instances

Instance	Physical Address
DISPC_VID1	0255 7208h

Figure 11-266. DISPC_VID1_MFLAG_THRESHOLD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HT_MFLAG																LT_MFLAG															
R/W-0h																R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-590. DISPC_VID1_MFLAG_THRESHOLD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	HT_MFLAG	R/W	0h	High Thresholds (in 128bits) for MFLAG generation: when FIFO fullness reaches HT_MFLAG level, MFLAG is reset to 0
15-0	LT_MFLAG	R/W	0h	Low Thresholds (in 128bits) for MFLAG generation: when FIFO fullness reaches LT_MFLAG level, MFLAG is set to 1

Table 11-591. Register Call Summary for DISPC_VID1_MFLAG_THRESHOLD

DSS Functional Description • DISPC DMA MFLAG Mechanism: [0][1][2][3][4][5]
DISPC_VID1 Registers • DISPC_VID1_MFLAG_THRESHOLD Register (Offset = 208h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.34 DISPC_VID1_PICTURE_SIZE Register (Offset = 20Ch) [reset = 0h]

DISPC_VID1_PICTURE_SIZE is shown in Figure 11-267 and described in Table 11-593.

Return to Summary Table.

The register configures the size of the video picture associated with the video layer before up/down-scaling. Shadow register.

Table 11-592. DISPC_VID1_PICTURE_SIZE Instances

Instance	Physical Address
DISPC_VID1	0255 720Ch

Figure 11-267. DISPC_VID1_PICTURE_SIZE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				MEMSIZEY											
R-0h				R/W-0h											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				MEMSIZEX											
R-0h				R/W-0h											

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-593. DISPC_VID1_PICTURE_SIZE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-16	MEMSIZEY	R/W	0h	Number of lines of the video picture Encoded value (from 1 to 4096) to specify the number of lines of the video picture in memory (program to value minus one). When predecimation is set, the value represents the size of the image after predecimation but the max size of the unpredecimated image size in memory is still bounded to 2exp(11).
15-12	RESERVED	R	0h	Reserved
11-0	MEMSIZEX	R/W	0h	Number of pixels of the video picture Encoded value (from 1 to 4096) to specify the number of pixels of the video picture in memory (program to value minus one). The size is limited to the size of the line buffer of the vertical sampling block in case the video picture is processed by the vertical filtering unit. (program to value minus one). When predecimation is set, the value represents the size of the image after predecimation but the max size of the unpredecimated image size in memory is still bounded to 2exp(11).

Table 11-594. Register Call Summary for DISPC_VID1_PICTURE_SIZE

DSS Functional Description
DISPC_VID1 Registers • DISPC_VID1_PICTURE_SIZE Register (Offset = 20Ch) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.35 DISPC_VID1_PIXEL_INC Register (Offset = 210h) [reset = 1h]

DISPC_VID1_PIXEL_INC is shown in Figure 11-268 and described in Table 11-596.

Return to Summary Table.

The register configures the number of bytes to increment between two pixels for the buffer associated with the video window. The register is used in order to perform low performance rotation. Shadow register.

Table 11-595. DISPC_VID1_PIXEL_INC Instances

Instance	Physical Address
DISPC_VID1	0255 7210h

Figure 11-268. DISPC_VID1_PIXEL_INC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PIXELINC							
R-0h																								R/W-1h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-596. DISPC_VID1_PIXEL_INC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
7-0	PIXELINC	R/W	1h	Number of bytes to increment between two pixels. Encoded unsigned value (from 1 to 255) to specify the number of bytes between two pixels in the video buffer. The value 0 is invalid. The value 1 means next pixel. The value 1+n*bpp means increment of n pixels. For YUV420, Max supported value is 128.

Table 11-597. Register Call Summary for DISPC_VID1_PIXEL_INC

DSS Functional Description • DISPC DMA Predecimation: [0] • DISPC DMA Addressing and Bursts: [0][1]
DISPC_VID1 Registers • DISPC_VID1_PIXEL_INC Register (Offset = 210h) [reset = 1h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.36 DISPC_VID1_POSITION Register (Offset = 214h) [reset = 0h]

DISPC_VID1_POSITION is shown in Figure 11-269 and described in Table 11-599.

Return to Summary Table.

The register configures the position of the video window. Shadow register.

Table 11-598. DISPC_VID1_POSITION Instances

Instance	Physical Address
DISPC_VID1	0255 7214h

Figure 11-269. DISPC_VID1_POSITION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				POSY											
R-0h				R/W-0h											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				POSX											
R-0h				R/W-0h											

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-599. DISPC_VID1_POSITION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-16	POSY	R/W	0h	Y position of the video window Encoded value (from 0 to 4095) to specify the Y position of the video window #1 .The line at the top has the Y-position 0.
15-12	RESERVED	R	0h	Reserved
11-0	POSX	R/W	0h	X position of the video window Encoded value (from 0 to 4095) to specify the X position of the video window #1. The first pixel on the left of the display screen has the X-position 0.

Table 11-600. Register Call Summary for DISPC_VID1_POSITION

DSS Functional Description • DISPC VID1 Scaler Unit: [0][1]
DISPC_VID1 Registers • DISPC_VID1_POSITION Register (Offset = 214h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.37 DISPC_VID1_PRELOAD Register (Offset = 218h) [reset = 100h]

DISPC_VID1_PRELOAD is shown in Figure 11-270 and described in Table 11-602.

Return to Summary Table.

The register configures the DMA buffer of the video pipeline. Shadow register.

Table 11-601. DISPC_VID1_PRELOAD Instances

Instance	Physical Address
DISPC_VID1	0255 7218h

Figure 11-270. DISPC_VID1_PRELOAD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				PRELOAD											
R-0h																				R/W-100h											

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-602. DISPC_VID1_PRELOAD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
11-0	PRELOAD	R/W	100h	DMA buffer preload value Number of 128-bit words defining the preload value.

Table 11-603. Register Call Summary for DISPC_VID1_PRELOAD

DSS Functional Description • DISPC Read DMA Buffer: [0]
DISPC_VID1 Registers • DISPC_VID1_PRELOAD Register (Offset = 218h) [reset = 100h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.38 DISPC_VID1_ROW_INC Register (Offset = 21Ch) [reset = 1h]

DISPC_VID1_ROW_INC is shown in Figure 11-271 and described in Table 11-605.

Return to Summary Table.

The register configures the number of bytes to increment at the end of the row for the buffer associated with the video window. Shadow register.

Table 11-604. DISPC_VID1_ROW_INC Instances

Instance	Physical Address
DISPC_VID1	0255 721Ch

Figure 11-271. DISPC_VID1_ROW_INC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROWINC																															
R/W-1h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-605. DISPC_VID1_ROW_INC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ROWINC	R/W	1h	Number of bytes to increment at the end of the row Encoded signed value (from -2^{31}-1 to 2^{31}) to specify the number of bytes to increment at the end of the row in the video buffer. The value 0 is invalid. The value 1 means next pixel. The value 1+n*bpp means increment of n pixels. The value 1- (n+1)*bpp means decrement of n pixels.

Table 11-606. Register Call Summary for DISPC_VID1_ROW_INC

DSS Functional Description • DISPC DMA Predecimation: [0]
DISPC_VID1 Registers • DISPC_VID1_ROW_INC Register (Offset = 21Ch) [reset = 1h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.39 DISPC_VID1_SIZE Register (Offset = 220h) [reset = 0h]

DISPC_VID1_SIZE is shown in Figure 11-272 and described in Table 11-608.

Return to Summary Table.

The register configures the size of the video window. Shadow register.

Table 11-607. DISPC_VID1_SIZE Instances

Instance	Physical Address
DISPC_VID1	0255 7220h

Figure 11-272. DISPC_VID1_SIZE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				SIZEY											
R-0h				R/W-0h											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				SIZEX											
R-0h				R/W-0h											

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-608. DISPC_VID1_SIZE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-16	SIZEY	R/W	0h	Number of lines of the video window. Encoded value (from 1 to 4096) to specify the number of lines of the video window (program size -1).
15-12	RESERVED	R	0h	Reserved
11-0	SIZEX	R/W	0h	Number of pixels of the video window. Encoded value (from 1 to 4096) to specify the number of pixels of the video window (program size -1).

Table 11-609. Register Call Summary for DISPC_VID1_SIZE

DSS Functional Description • DISPC DMA Addressing and Bursts: [0]
DISPC_VID1 Registers • DISPC_VID1_SIZE Register (Offset = 220h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.3.40 DISPC_VID1_CLUT Register (Offset = 224h) [reset = 0h]

DISPC_VID1_CLUT is shown in Figure 11-273 and described in Table 11-611.

Return to Summary Table.

The register configures the Color Look Up Table (CLUT) for VID pipeline. CLUT is used in conjunction with bitmap formats

Table 11-610. DISPC_VID1_CLUT Instances

Instance	Physical Address
DISPC_VID1	0255 7224h

Figure 11-273. DISPC_VID1_CLUT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INDEX								VALUE_R								VALUE_G								VALUE_B							
W-0h								W-0h								W-0h								W-0h							

LEGEND: W = Write Only; *-n* = value after reset

Table 11-611. DISPC_VID1_CLUT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	INDEX	W	0h	Defines the location in the table where the bit-field VALUE is stored
23-16	VALUE_R	W	0h	8-bit value used to defined the value to store at the location in the table defined by the bit-field INDEX.
15-8	VALUE_G	W	0h	8-bit value used to defined the value to store at the location in the table defined by the bit-field INDEX.
7-0	VALUE_B	W	0h	8-bit value used to defined the value to store at the location in the table defined by the bit-field INDEX.

Table 11-612. Register Call Summary for DISPC_VID1_CLUT

DSS Functional Description • DISPC VID1 Color Look-Up Table (CLUT): [0][1]
DISPC_VID1 Registers • DISPC_VID1_CLUT Register (Offset = 224h) [reset = 0h]: [0] • DISPC_VID1 Registers: [0]

11.3.5.4 DISPC_OVR1 Registers

Table 11-614 lists the memory-mapped registers for the DISPC_OVR1. All register offset addresses not listed in Table 11-614 should be considered as reserved locations and the register contents should not be modified.

This section describes the DISPC_OVR1 instances registers.

Table 11-613. DISPC_OVR1 Instances

Instance	Base Address
DISPC_OVR1	0255 A800h

Table 11-614. DISPC_OVR1 Registers

Offset	Acronym	Register Name	DISPC_OVR1 Physical Address	Section
0h	DISPC_OVR1_CONFIG	The control register configures the Display Controller module for the VP output. Shadow register. **Note: Feature is not supported in this family of devices.**	0255 A800h	Section 11.3.5.4.1
4h	RESERVED0	Reserved	0255 A804h	
8h	DISPC_OVR1_DEFAULT_COLOR	The control register configures the default solid background color bits [31-0]. Shadow register.	0255 A808h	Section 11.3.5.4.2
Ch	DISPC_OVR1_DEFAULT_COLOR2	The control register configures the default solid background color bits [47-32]. Shadow register.	0255 A80Ch	Section 11.3.5.4.3
10h	DISPC_OVR1_TRANS_COLOR_MAX	The register sets the max transparency color value for the overlays. Shadow register. **Note: Feature is not supported in this family of devices.**	0255 A810h	Section 11.3.5.4.4
14h	DISPC_OVR1_TRANS_COLOR_MAX2	The register sets the max transparency color value for the overlays. Shadow register. **Note: Feature is not supported in this family of devices.**	0255 A814h	Section 11.3.5.4.5
18h	DISPC_OVR1_TRANS_COLOR_MIN	The register sets the min transparency color value for the overlays. Shadow register. **Note: Feature is not supported in this family of devices.**	0255 A818h	Section 11.3.5.4.6
1Ch	DISPC_OVR1_TRANS_COLOR_MIN2	The register sets the min transparency color value for the overlays. Shadow register. **Note: Feature is not supported in this family of devices.**	0255 A81Ch	Section 11.3.5.4.7

11.3.5.4.1 DISPC_OVR1_CONFIG Register (Offset = 0h) [reset = 0h]

DISPC_OVR1_CONFIG is shown in Figure 11-274 and described in Table 11-616.

Return to Summary Table.

The control register configures the Display Controller module for the VP output. Shadow register.
Note: Feature is not supported in this family of devices.

Table 11-615. DISPC_OVR1_CONFIG Instances

Instance	Physical Address
DISPC_OVR1	0255 A800h

Figure 11-274. DISPC_OVR1_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		RESERVED	RESERVED	TCKLCDSELECTION	TCKLCDENABLE	INTERLEAVED3DMODE	
R-0h		R-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-616. DISPC_OVR1_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13	RESERVED	R	0h	Reserved
12	RESERVED	R	0h	Reserved
11	TCKLCDSELECTION	R/W	0h	Transparency Color Key Selection Shadow bit-field. 0h (R/W) = Destination transparency color key selected 1h (R/W) = Source transparency color key selected
10	TCKLCDENABLE	R/W	0h	Transparency Color Key Enabled Shadow bit-field. 0h (R/W) = Disable the transparency color key 1h (R/W) = Enable the transparency color key
9-8	INTERLEAVED3DMODE	R/W	0h	Define which layer contributes to odd/even lines of the line interleaving 3D format 0h (R/W) = No interleaving happens in the overlay manager 1h (R/W) = RESERVED 2h (R/W) = At even lines (all pixels) have a contribution from even z-order pipes and odd lines (all pixels) have a contribution from the odd z-order pipes. 3h (R/W) = A even pixels (for all lines) have a contribution from even z-order pipes and odd pixels (for all lines) have a contribution from the odd z-order pipes
7-0	RESERVED	R	0h	Reserved

Table 11-617. Register Call Summary for DISPC_OVR1_CONFIG

DISPC_OVR1 Registers
• DISPC_OVR1_CONFIG Register (Offset = 0h) [reset = 0h]: [0] • DISPC_OVR1 Registers: [0]

11.3.5.4.2 DISPC_OVR1_DEFAULT_COLOR Register (Offset = 8h) [reset = 0h]

DISPC_OVR1_DEFAULT_COLOR is shown in Figure 11-275 and described in Table 11-619.

Return to Summary Table.

The control register configures the default solid background color bits [31-0]. Shadow register.

Table 11-618. DISPC_OVR1_DEFAULT_COLOR Instances

Instance	Physical Address
DISPC_OVR1	0255 A808h

Figure 11-275. DISPC_OVR1_DEFAULT_COLOR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEFAULTCOLOR																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-619. DISPC_OVR1_DEFAULT_COLOR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DEFAULTCOLOR	R/W	0h	48-bit ARGB color value to specify the default solid color to display when there is no data from the overlays. Only [31-0] is defined in this register. Refer to DEFAULT_COLOR2 for [47-32] bits.

Table 11-620. Register Call Summary for DISPC_OVR1_DEFAULT_COLOR

DSS Functional Description • DISPC Overlay Manager: [0]
DISPC_OVR1 Registers • DISPC_OVR1 Registers: [0] • DISPC_OVR1_DEFAULT_COLOR Register (Offset = 8h) [reset = 0h]: [0]

11.3.5.4.3 DISPC_OVR1_DEFAULT_COLOR2 Register (Offset = Ch) [reset = 0h]

DISPC_OVR1_DEFAULT_COLOR2 is shown in Figure 11-276 and described in Table 11-622.

Return to Summary Table.

The control register configures the default solid background color bits [47-32]. Shadow register.

Table 11-621. DISPC_OVR1_DEFAULT_COLOR2 Instances

Instance	Physical Address
DISPC_OVR1	0255 A80Ch

Figure 11-276. DISPC_OVR1_DEFAULT_COLOR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																DEFAULTCOLOR															
R-0h																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-622. DISPC_OVR1_DEFAULT_COLOR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	DEFAULTCOLOR	R/W	0h	48-bit ARGB color value to specify the default solid color to display when there is no data from the overlays. Only [47-32] is defined in this register. refer to DEFAULT_COLOR for [31-0] bits

Table 11-623. Register Call Summary for DISPC_OVR1_DEFAULT_COLOR2

DSS Functional Description • DISPC Overlay Manager: [0]
DISPC_OVR1 Registers • DISPC_OVR1 Registers: [0] • DISPC_OVR1_DEFAULT_COLOR2 Register (Offset = Ch) [reset = 0h]: [0]

11.3.5.4.4 DISPC_OVR1_TRANS_COLOR_MAX Register (Offset = 10h) [reset = 0h]

DISPC_OVR1_TRANS_COLOR_MAX is shown in Figure 11-277 and described in Table 11-625.

Return to Summary Table.

The register sets the max transparency color value for the overlays. Shadow register.
Note: Feature is not supported in this family of devices.

Table 11-624. DISPC_OVR1_TRANS_COLOR_MAX Instances

Instance	Physical Address
DISPC_OVR1	0255 A810h

Figure 11-277. DISPC_OVR1_TRANS_COLOR_MAX Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRANSCOLORKEY																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-625. DISPC_OVR1_TRANS_COLOR_MAX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TRANSCOLORKEY	R/W	0h	[31-0] Transparency Color Key Value in 36-bit RGB format

Table 11-626. Register Call Summary for DISPC_OVR1_TRANS_COLOR_MAX

DISPC_OVR1 Registers
- DISPC_OVR1 Registers: [0]
- DISPC_OVR1_TRANS_COLOR_MAX Register (Offset = 10h) [reset = 0h]: [0]

11.3.5.4.5 DISPC_OVR1_TRANS_COLOR_MAX2 Register (Offset = 14h) [reset = 0h]

DISPC_OVR1_TRANS_COLOR_MAX2 is shown in Figure 11-278 and described in Table 11-628.

Return to Summary Table.

The register sets the max transparency color value for the overlays. Shadow register.
Note: Feature is not supported in this family of devices.

Table 11-627. DISPC_OVR1_TRANS_COLOR_MAX2 Instances

Instance	Physical Address
DISPC_OVR1	0255 A814h

Figure 11-278. DISPC_OVR1_TRANS_COLOR_MAX2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				TRANSCOLORKEY			
R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-628. DISPC_OVR1_TRANS_COLOR_MAX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-0	TRANSCOLORKEY	R/W	0h	[35-32] Transparency Color Key Value in 36-bit RGB format

Table 11-629. Register Call Summary for DISPC_OVR1_TRANS_COLOR_MAX2

DISPC_OVR1 Registers
- DISPC_OVR1_TRANS_COLOR_MAX2 Register (Offset = 14h) [reset = 0h]: [0]
- DISPC_OVR1 Registers: [0]

11.3.5.4.6 DISPC_OVR1_TRANS_COLOR_MIN Register (Offset = 18h) [reset = 0h]

DISPC_OVR1_TRANS_COLOR_MIN is shown in Figure 11-279 and described in Table 11-631.

Return to Summary Table.

The register sets the min transparency color value for the overlays. Shadow register.
Note: Feature is not supported in this family of devices.

Table 11-630. DISPC_OVR1_TRANS_COLOR_MIN Instances

Instance	Physical Address
DISPC_OVR1	0255 A818h

Figure 11-279. DISPC_OVR1_TRANS_COLOR_MIN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRANSCOLORKEY																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-631. DISPC_OVR1_TRANS_COLOR_MIN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TRANSCOLORKEY	R/W	0h	[31-0] Transparency Color Key Value in 36-bit RGB format

Table 11-632. Register Call Summary for DISPC_OVR1_TRANS_COLOR_MIN

DISPC_OVR1 Registers
- DISPC_OVR1_TRANS_COLOR_MIN Register (Offset = 18h) [reset = 0h]: [0]
- DISPC_OVR1 Registers: [0]

11.3.5.4.7 DISPC_OVR1_TRANS_COLOR_MIN2 Register (Offset = 1Ch) [reset = 0h]

DISPC_OVR1_TRANS_COLOR_MIN2 is shown in Figure 11-280 and described in Table 11-634.

Return to Summary Table.

The register sets the min transparency color value for the overlays. Shadow register.
Note: Feature is not supported in this family of devices.

Table 11-633. DISPC_OVR1_TRANS_COLOR_MIN2 Instances

Instance	Physical Address
DISPC_OVR1	0255 A81Ch

Figure 11-280. DISPC_OVR1_TRANS_COLOR_MIN2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				TRANSCOLORKEY			
R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-634. DISPC_OVR1_TRANS_COLOR_MIN2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-0	TRANSCOLORKEY	R/W	0h	[35-32] Transparency Color Key Value in 36-bit RGB format

Table 11-635. Register Call Summary for DISPC_OVR1_TRANS_COLOR_MIN2

DISPC_OVR1 Registers
- DISPC_OVR1_TRANS_COLOR_MIN2 Register (Offset = 1Ch) [reset = 0h]: [0]
- DISPC_OVR1 Registers: [0]

11.3.5.5 DISPC_VP1 Registers

Table 11-637 lists the memory-mapped registers for the DISPC_VP1. All register offset addresses not listed in Table 11-637 should be considered as reserved locations and the register contents should not be modified.

This section describes the DISPC_VP1 instances registers.

Table 11-636. DISPC_VP1 Instances

Instance	Base Address
DISPC_VP1	0255 AC00h

Table 11-637. DISPC_VP1 Registers

Offset	Acronym	Register Name	DISPC_VP1 Physical Address	Section
0h	DISPC_VP1_CONFIG	The control register configures the Display Controller module for the VP output. Shadow register.	0255 AC00h	Section 11.3.5.5.1
4h	DISPC_VP1_CONTROL	The control register configures the Display Controller module for the VP output.	0255 AC04h	Section 11.3.5.5.2
8h	DISPC_VP1_CPR_COEF_B	The register configures the color phase rotation matrix coefficients for the Blue component. Shadow register.	0255 AC08h	Section 11.3.5.5.3
Ch	DISPC_VP1_CPR_COEF_G	The register configures the color phase rotation matrix coefficients for the Green component. Shadow register.	0255 AC0Ch	Section 11.3.5.5.4
10h	DISPC_VP1_CPR_COEF_R	The register configures the color phase rotation matrix coefficients for the Red component. Shadow register.	0255 AC10h	Section 11.3.5.5.5
14h to 1Ch	DISPC_VP1_DATA_CYCLE_0 to DISPC_VP1_DATA_CYCLE_2	The control register configures the output data format over up to 3 cycles. Shadow register.	0255 AC13h to 0255 AC1Ch	Section 11.3.5.5.6
20h	DISPC_VP1_GAMMA_TABLE	The register configures the gamma table on VP output.	0255 AC20h	Section 11.3.5.5.7
3Ch	DISPC_VP1_IRQENABLE	This register allows to mask/unmask the module internal sources of interrupt, on an event-by-event basis	0255 AC3Ch	Section 11.3.5.5.8
40h	DISPC_VP1_IRQSTATUS	This register regroups all the status of the module internal events that generate an interrupt. Write 1 to a given bit resets this bit	0255 AC40h	Section 11.3.5.5.9
44h	DISPC_VP1_LINE_NUMBER	The control register indicates the panel display line number for the interrupt and the DMA request. Shadow register.	0255 AC44h	Section 11.3.5.5.10
4Ch	DISPC_VP1_POL_FREQ	The register configures the signal configuration. Shadow register.	0255 AC4Ch	Section 11.3.5.5.11
50h	DISPC_VP1_SIZE_SCREEN	The register configures the panel size (horizontal and vertical). Shadow register. A delta value is used to indicate if the odd field has same vertical size as the even field or +/- one line.	0255 AC50h	Section 11.3.5.5.12

Table 11-637. DISPC_VP1 Registers (continued)

Offset	Acronym	Register Name	DISPC_VP1 Physical Address	Section
54h	DISPC_VP1_TIMING_H	The register configures the timing logic for the HSYNC signal. Shadow register.	0255 AC54h	Section 11.3.5.5.13
58h	DISPC_VP1_TIMING_V	The register configures the timing logic for the VSYNC signal. Shadow register.	0255 AC58h	Section 11.3.5.5.14

11.3.5.5.1 DISPC_VP1_CONFIG Register (Offset = 0h) [reset = 0h]

DISPC_VP1_CONFIG is shown in Figure 11-281 and described in Table 11-639.

Return to Summary Table.

The control register configures the Display Controller module for the VP output. Shadow register.

Table 11-638. DISPC_VP1_CONFIG Instances

Instance	Physical Address
DISPC_VP1	0255 AC00h

Figure 11-281. DISPC_VP1_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED						FULLRANGE	COLORCONVENABLE
R-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
FIDFIRST	OUTPUTMODEENABLE	BT1120ENABLE	BT656ENABLE	RESERVED			BUFFERHANDSHAKE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h			R/W-0h
15	14	13	12	11	10	9	8
CPR	RESERVED						EXTERNALSYNCEN
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
VSYNCGATED	HSYNCGATED	PIXELCLOCKGATED	PIXELDATAGATED	HDMIMODE	GAMMAENABLE	DATAENABLEGATED	PIXELGATED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-639. DISPC_VP1_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	FULLRANGE	R/W	0h	Color Space Conversion full range setting. 0h (R/W) = Limited range selected. 1h (R/W) = Full range selected.
24	COLORCONVENABLE	R/W	0h	Enable the color space conversion. It shall be reset when CPR bit-field is set to 0x1. 0h (R/W) = Disable Color Space Conversion RGB to YUV 1h (R/W) = Enable Color Space Conversion RGB to YUV
23	FIDFIRST	R/W	0h	Selects the first field to output in case of interlace mode. In case of progressive mode, the value is not used. 0h (R/W) = First field is even. 1h (R/W) = Odd field is first.
22	OUTPUTMODEENABLE	R/W	0h	Selects between progressive and interlace mode for the VP output. 0h (R/W) = Progressive mode selected. 1h (R/W) = Interlace mode selected.
21	BT1120ENABLE	R/W	0h	Selects BT-1120 format on the VP output. It is not posssible to enable BT656 and BT1120 at the same time on the same VP output. 0h (R/W) = BT-1120 is disabled. 1h (R/W) = BT-1120 is enabled.

Table 11-639. DISPC_VP1_CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	BT656ENABLE	R/W	0h	Selects BT-656 format on the VP output. It is not posssible to enable BT656 and BT1120 at the same time on the same VP output. 0h (R/W) = BT-656 is disabled. 1h (R/W) = BT-656 is enabled.
19-17	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
16	BUFFERHANDSHAKE	R/W	0h	Controls the handshake between DMA buffer and STALL signal in order to prevent from underflow. The bit shall be set to 0 when the module is not in STALL mode. 0h (R/W) = Only the STALL signal (generated by RFBI) is used regardless of the DMA buffer fullness information in order to provide data to the RFBI module. 1h (R/W) = The STALL signal (generated by RFBI) is used in combination with the DMA buffer fullness information in order to provide data to the RFBI module only when it does not generated buffer underflow.
15	CPR	R/W	0h	Color Phase Rotation Control VP output). It shall be reset when ColorConvEnable bit-field is set to 1. Shadow bit-field. 0h (R/W) = Color Phase Rotation Disabled 1h (R/W) = Color Phase Rotation Enabled
14-9	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
8	EXTERNALSYNCEN	R/W	0h	Selects between external sync and internal sync mode for the VP output. 0h (R/W) = Internal sync mode selected 1h (R/W) = External sync mode selected.
7	VSYNCGATED	R/W	0h	VSYNC Gated Enabled (VP output) Shadow bit-field. 0h (R/W) = VSYNC Gated Disabled 1h (R/W) = VSYNC Gated Enabled
6	HSYNCGATED	R/W	0h	HSYNC Gated Enabled (VP output) Shadow bit-field. 0h (R/W) = HSYNC Gated Disabled 1h (R/W) = HSYNC Gated Enabled
5	PIXELCLOCKGATED	R/W	0h	Pixel Clock Gated Enabled (VP output) Shadow bit-field. 0h (R/W) = Pixel Clock Gated Disabled 1h (R/W) = Pixel Clock Gated Enabled
4	PIXELDATAGATED	R/W	0h	Pixel Data Gated Enabled (VP output) Shadow bit-field. 0h (R/W) = Pixel Data Gated Disabled 1h (R/W) = Pixel Data Gated Enabled
3	HDMIMODE	R/W	0h	Configures the timing generator in HDMI compatible mode to generate same timings as HDMI wrapper timings. **Note: Feature is not supported in this family of devices.** 0h (R/W) = Disable HDMI mode. 1h (R/W) = Enable HDMI mode timings: - vertical FSM starts with the VFP period (instead of VSYNC) - last frame ends with the last pixel of data state.
2	GAMMAENABLE	R/W	0h	Enable the gamma Shadow bit-field. 0h (R/W) = Gamma disabled 1h (R/W) = Gamma enabled
1	DATAENABLEGATED	R/W	0h	DE Gated Enable Shadow bit-field. 0h (R/W) = DE signal is not gated 1h (R/W) = DE signal is gated.
0	PIXELGATED	R/W	0h	Pixel Gated Enable Shadow bit-field. 0h (R/W) = Pixel clock always toggles (only in TFT mode) 1h (R/W) = Pixel clock only toggles when there is valid data to display. (only in TFT mode)

Table 11-640. Register Call Summary for DISPC_VP1_CONFIG

DISPC_VP1 Registers • DISPC_VP1 Registers: [0] • DISPC_VP1_CONFIG Register (Offset = 0h) [reset = 0h]: [0]
DSS Functional Description • DISPC Stall Mode: [0] • RFBI Stall Mechanism: [0] • DISPC VP1 BT.656 and BT.1120 Modes: [0] • DISPC VP1 Color Space Conversion: [0][1][2] • DISPC Clock Configuration: [0] • DISPC VP1 Color Phase Rotation Unit: [0] • DISPC VP1 Timing Generator and Panel Settings: [0][1][2][3][4] • DISPC VP1 Gamma Correction Unit: [0]

11.3.5.5.2 DISPC_VP1_CONTROL Register (Offset = 4h) [reset = 0h]

DISPC_VP1_CONTROL is shown in Figure 11-282 and described in Table 11-642.

Return to Summary Table.

The control register configures the Display Controller module for the VP output.

Table 11-641. DISPC_VP1_CONTROL Instances

Instance	Physical Address
DISPC_VP1	0255 AC04h

Figure 11-282. DISPC_VP1_CONTROL Register

31	30	29	28	27	26	25	24
SPATIALTEMPORALDITHERINGFRAMES		RESERVED			TDMUNUSEDBITS		TDMCYCLEFORMAT
R/W-0h		R-0h			R/W-0h		R/W-0h
23	22	21	20	19	18	17	16
TDMCYCLEFORMAT	TDMPARALLELMODE		TDMENABLE	RESERVED			HT
R/W-0h	R/W-0h		R/W-0h	R-0h			R/W-0h
15	14	13	12	11	10	9	8
HT		RESERVED	RESERVED	STALLMODE	DATALINES		
R/W-0h		R-0h	R-0h	R/W-0h	R/W-0h		
7	6	5	4	3	2	1	0
STDITHERENABLE	RESERVED	GOBIT	M8B	STN	MONOCOLOR	VPPROGLINENUMBERMODULO	VPENABLE
R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-642. DISPC_VP1_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	SPATIALTEMPORALDITHERINGFRAMES	R/W	0h	Spatial/Temporal dithering number of frames for the VP output. Shadow bit-field. 0h (R/W) = Spatial only 1h (R/W) = Spatial and temporal over 2 frames 2h (R/W) = Spatial and temporal over 4 frames 3h (R/W) = Reserved
29-27	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
26-25	TDMUNUSEDBITS	R/W	0h	State of unused bits (TDM mode only) for the VP output. Shadow bit-field. 0h (R/W) = low level (0) 1h (R/W) = high level (1) 2h (R/W) = unchanged from previous state 3h (R/W) = reserved
24-23	TDMCYCLEFORMAT	R/W	0h	Cycle format (TDM mode only) for the VP output. Shadow bit-field. 0h (R/W) = 1 cycle for 1 pixel 1h (R/W) = 2 cycles for 1 pixel 2h (R/W) = 3 cycles for 1 pixel 3h (R/W) = 3 cycles for 2 pixels

Table 11-642. DISPC_VP1_CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22-21	TDMPARALLELMODE	R/W	0h	Output Interface width (TDM mode only) for the VP output. Shadow bit-field. 0h (R/W) = 8-bit parallel output interface selected 1h (R/W) = 9-bit parallel output interface selected 2h (R/W) = 12-bit parallel output interface selected 3h (R/W) = 16-bit parallel output interface selected
20	TDMENABLE	R/W	0h	Enable the multiple cycle format for the VP output. Shadow bit-field. 0h (R/W) = TDM disabled 1h (R/W) = TDM enabled
19-17	RESERVED	R	0h	Reserved
16-14	HT	R/W	0h	Hold Time for VP output. Shadow bit-field. Encoded value (from 1 to 8) to specify the number of external digital clock periods to hold the data (programmed value = value minus one)
13	RESERVED	R	0h	Reserved
12	RESERVED	R	0h	Reserved
11	STALLMODE	R/W	0h	STALL Mode for the VP output. Shadow bit-field. 0h (R/W) = Normal mode selected 1h (R/W) = STALL mode selected. The Display Controller sends the data without considering the VSYNC/HSYNC. The VP output is disabled at the end of the transfer of the frame. The S/W has to re-enable the VP output in order to generate a new frame.
10-8	DATALINES	R/W	0h	Width of the data bus on VP output. Shadow bit-field. 0h (R/W) = 12-bit output aligned on the LSB of the pixel data interface 1h (R/W) = 16-bit output aligned on the LSB of the pixel data interface 2h (R/W) = 18-bit output aligned on the LSB of the pixel data interface 3h (R/W) = 24-bit output aligned on the LSB of the pixel data interface 4h (R/W) = 30-bit output aligned on the LSB of the pixel data interface 5h (R/W) = 36-bit output aligned on the LSB of the pixel data interface
7	STDITHERENABLE	R/W	0h	Spatial Temporal dithering enable for the VP output Shadow bit-field. 0h (R/W) = Spatial/Temporal dithering logic disabled 1h (R/W) = Spatial/Temporal dithering logic enabled
6	RESERVED	R	0h	Reserved
5	GOBIT	R/W	0h	GO Command for the VP output. It is used to synchronized the pipelines associated with the VP output. wr:immediate 0h (R/W) = The hardware has finished updating the internal shadow registers of the pipeline(s) connected to the VP output using the user values. The hardware resets the bit when the update is completed. 1h (R/W) = The user has finished to program the shadow registers of the pipeline(s) associated with the VP output and the hardware can update the internal registers at the VFP start period
4	M8B	R/W	0h	Mono 8-bit mode of the primary LCD wr: VFP start period of primary LCD output 0h (R/W) = Pixel data [3:0] is used to output four pixel values to the panel at each pixel clock transition. (only in Passive Mono 4-bit mode) 1h (R/W) = Pixel data [7:0] is used to output eight pixel values to the panel each pixel clock transition. (only in Passive Mono 8-bit mode)

Table 11-642. DISPC_VP1_CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	STN	R/W	0h	LCD Display type of the primary LCD wr: VFP start period of primary LCD output 0h (R/W) = Passive or STN display operation disabled. 1h (R/W) = Passive or STN display operation enabled. STN dither logic is enabled.
2	MONOCOLOR	R/W	0h	Monochrome/Color selection for the primary LCD wr: VFP start period of primary LCD output only) 0h (R/W) = Color operation enabled (STN mode only) 1h (R/W) = Monochrome operation enabled (STN mode only)
1	VPPROGLINENUMBERMODULO	R/W	0h	Enable the modulo of the line number interrupt generation 0h (R/W) = Disable modulo 1h (R/W) = Enable Modulo
0	VPENABLE	R/W	0h	Enable the video port output wr:immediate 0h (R/W) = VP output disabled (at the end of the frame when the bit is reset) 1h (R/W) = VP output enabled

Table 11-643. Register Call Summary for DISPC_VP1_CONTROL

DISPC_VP1 Registers • DISPC_VP1 Registers: [0] • DISPC_VP1_CONTROL Register (Offset = 4h) [reset = 0h]: [0]
DSS Functional Description • DISPC Shadow Registers: [0] • DISPC VP1 Multiple Cycle Output Format (TDM): [0][1][2][3][4] • RFBI Stall Mechanism: [0][1] • DISPC VP1 Configuration for TV Support: [0][1] • DISPC Stall Mode: [0][1][2] • DISPC VP1 Spatial/Temporal Dithering: [0][1]
DSS Environment • DISPC VP1 Output and Data Formats: [0]

11.3.5.5.3 DISPC_VP1_CPR_COEF_B Register (Offset = 8h) [reset = 0h]

DISPC_VP1_CPR_COEF_B is shown in Figure 11-283 and described in Table 11-645.

Return to Summary Table.

The register configures the color phase rotation matrix coefficients for the Blue component. Shadow register.

Table 11-644. DISPC_VP1_CPR_COEF_B Instances

Instance	Physical Address
DISPC_VP1	0255 AC08h

Figure 11-283. DISPC_VP1_CPR_COEF_B Register

31	30	29	28	27	26	25	24
BR							
R/W-0h							
23	22	21	20	19	18	17	16
BR		RESERVED	BG				
R/W-0h		R-0h	R/W-0h				
15	14	13	12	11	10	9	8
BG					RESERVED	BB	
R/W-0h					R-0h	R/W-0h	
7	6	5	4	3	2	1	0
BB							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-645. DISPC_VP1_CPR_COEF_B Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	BR	R/W	0h	BR Coefficient Encoded signed value (from -512 to 511).
21	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
20-11	BG	R/W	0h	BG Coefficient Encoded signed value (from -512 to 511).
10	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
9-0	BB	R/W	0h	BB Coefficient Encoded signed value (from -512 to 511).

Table 11-646. Register Call Summary for DISPC_VP1_CPR_COEF_B

DISPC_VP1 Registers • DISPC_VP1 Registers: [0] • DISPC_VP1_CPR_COEF_B Register (Offset = 8h) [reset = 0h]: [0]
DSS Functional Description • DISPC VP1 Color Phase Rotation Unit: [0][1]

11.3.5.5.4 DISPC_VP1_CPR_COEF_G Register (Offset = Ch) [reset = 0h]

DISPC_VP1_CPR_COEF_G is shown in Figure 11-284 and described in Table 11-648.

Return to Summary Table.

The register configures the color phase rotation matrix coefficients for the Green component. Shadow register.

Table 11-647. DISPC_VP1_CPR_COEF_G Instances

Instance	Physical Address
DISPC_VP1	0255 AC0Ch

Figure 11-284. DISPC_VP1_CPR_COEF_G Register

31	30	29	28	27	26	25	24
GR							
R/W-0h							
23	22	21	20	19	18	17	16
GR		RESERVED	GG				
R/W-0h		R-0h	R/W-0h				
15	14	13	12	11	10	9	8
GG					RESERVED	GB	
R/W-0h					R-0h	R/W-0h	
7	6	5	4	3	2	1	0
GB							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-648. DISPC_VP1_CPR_COEF_G Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	GR	R/W	0h	GR Coefficient Encoded signed value (from -512 to 511).
21	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
20-11	GG	R/W	0h	GG Coefficient Encoded signed value (from -512 to 511).
10	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
9-0	GB	R/W	0h	GB Coefficient Encoded signed value (from -512 to 511).

Table 11-649. Register Call Summary for DISPC_VP1_CPR_COEF_G

DISPC_VP1 Registers • DISPC_VP1 Registers: [0] • DISPC_VP1_CPR_COEF_G Register (Offset = Ch) [reset = 0h]: [0]
DSS Functional Description • DISPC VP1 Color Phase Rotation Unit: [0][1]

11.3.5.5.5 DISPC_VP1_CPR_COEF_R Register (Offset = 10h) [reset = 0h]

DISPC_VP1_CPR_COEF_R is shown in Figure 11-285 and described in Table 11-651.

Return to Summary Table.

The register configures the color phase rotation matrix coefficients for the Red component. Shadow register.

Table 11-650. DISPC_VP1_CPR_COEF_R Instances

Instance	Physical Address
DISPC_VP1	0255 AC10h

Figure 11-285. DISPC_VP1_CPR_COEF_R Register

31	30	29	28	27	26	25	24
RR							
R/W-0h							
23	22	21	20	19	18	17	16
RR		RESERVED	RG				
R/W-0h		R-0h	R/W-0h				
15	14	13	12	11	10	9	8
RG					RESERVED	RB	
R/W-0h					R-0h	R/W-0h	
7	6	5	4	3	2	1	0
RB							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-651. DISPC_VP1_CPR_COEF_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RR	R/W	0h	RR Coefficient Encoded signed value (from -512 to 511).
21	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
20-11	RG	R/W	0h	RG Coefficient Encoded signed value (from -512 to 511).
10	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
9-0	RB	R/W	0h	RB Coefficient Encoded signed value (from -512 to 511).

Table 11-652. Register Call Summary for DISPC_VP1_CPR_COEF_R

DISPC_VP1 Registers • DISPC_VP1 Registers: [0] • DISPC_VP1_CPR_COEF_R Register (Offset = 10h) [reset = 0h]: [0]
DSS Functional Description • DISPC VP1 Color Phase Rotation Unit: [0][1]

11.3.5.5.6 DISPC_VP1_DATA_CYCLE_0 to DISPC_VP1_DATA_CYCLE_2 Register (Offset = 14h to 1Ch) [reset = 0h]

DISPC_VP1_DATA_CYCLE_0 to DISPC_VP1_DATA_CYCLE_2 is shown in Figure 11-286 and described in Table 11-654.

Return to Summary Table.

The control register configures the output data format over up to 3 cycles. Shadow register.

Table 11-653. DISPC_VP1_DATA_CYCLE_0 to DISPC_VP1_DATA_CYCLE_2 Instances

Instance	Physical Address
DISPC_VP1	0255 AC14h to 0255 AC1Ch

Figure 11-286. DISPC_VP1_DATA_CYCLE_0 to DISPC_VP1_DATA_CYCLE_2 Register

31	30	29	28	27	26	25	24
RESERVED				BITALIGNMENTPIXEL2			
R-0h				R/W-0h			
23	22	21	20	19	18	17	16
RESERVED			NBBITSPIXEL2				
R-0h			R/W-0h				
15	14	13	12	11	10	9	8
RESERVED				BITALIGNMENTPIXEL1			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
RESERVED			NBBITSPIXEL1				
R-0h			R/W-0h				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-654. DISPC_VP1_DATA_CYCLE_0 to DISPC_VP1_DATA_CYCLE_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
27-24	BITALIGNMENTPIXEL2	R/W	0h	Bit alignment Alignment of the bits from pixel#2 on the output interface
23-21	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
20-16	NBBITSPIXEL2	R/W	0h	Number of bits Number of bits from the pixel #2 (value from 0 to 16 bits). The values from 17 to 31 are invalid.
15-12	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
11-8	BITALIGNMENTPIXEL1	R/W	0h	Bit alignment Alignment of the bits from pixel#1 on the output interface
7-5	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
4-0	NBBITSPIXEL1	R/W	0h	Number of bits Number of bits from the pixel #1 (value from 0 to 16 bits). The values from 17 to 31 are invalid.

Table 11-655. Register Call Summary for DISPC_VP1_DATA_CYCLE_0

DISPC_VP1 Registers • DISPC_VP1 Registers: [0] • DISPC_VP1_DATA_CYCLE_0 to DISPC_VP1_DATA_CYCLE_2 Register (Offset = 14h to 1Ch) [reset = 0h]: [0]
DSS Functional Description • DISPC VP1 Multiple Cycle Output Format (TDM): [0]

11.3.5.5.7 DISPC_VP1_GAMMA_TABLE Register (Offset = 20h) [reset = 0h]

DISPC_VP1_GAMMA_TABLE is shown in Figure 11-287 and described in Table 11-657.

Return to Summary Table.

The register configures the gamma table on VP output.

Table 11-656. DISPC_VP1_GAMMA_TABLE Instances

Instance	Physical Address
DISPC_VP1	0255 AC20h

Figure 11-287. DISPC_VP1_GAMMA_TABLE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INDEX								VALUE_R								VALUE_G								VALUE_B							
W-0h								W-0h								W-0h								W-0h							

LEGEND: W = Write Only; *-n* = value after reset

Table 11-657. DISPC_VP1_GAMMA_TABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	INDEX	W	0h	Defines the location in the table where the bit-field VALUE is stored
23-16	VALUE_R	W	0h	8-bit value used to defined the value to be stored in the gamma table
15-8	VALUE_G	W	0h	8-bit value used to defined the value to be stored in the gamma table
7-0	VALUE_B	W	0h	8-bit value used to defined the value to be stored in the gamma table

Table 11-658. Register Call Summary for DISPC_VP1_GAMMA_TABLE

DISPC_VP1 Registers • DISPC_VP1 Registers: [0] • DISPC_VP1_GAMMA_TABLE Register (Offset = 20h) [reset = 0h]: [0]
DSS Functional Description • DISPC VP1 Gamma Correction Unit: [0][1]

11.3.5.5.8 DISPC_VP1_IRQENABLE Register (Offset = 3Ch) [reset = 0h]

DISPC_VP1_IRQENABLE is shown in Figure 11-288 and described in Table 11-660.

Return to Summary Table.

This register allows to mask/unmask the module internal sources of interrupt, on an event-by-event basis

Table 11-659. DISPC_VP1_IRQENABLE Instances

Instance	Physical Address
DISPC_VP1	0255 AC3Ch

Figure 11-288. DISPC_VP1_IRQENABLE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		ACBIASCOUNTSTATUS_EN	VPSYNCLOST_EN	VPPROGRAMMEDLINENUMBER_EN	VPVSYNC_ODD_EN	VPVSYNC_EN	VPFRAMEDONE_EN
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-660. DISPC_VP1_IRQENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	ACBIASCOUNTSTATUS_EN	R/W	0h	AC BIAS transition counter has decremented to zero 0h (R/W) = ACBIASCOUNTSTATUS for the primary LCD output is masked 1h (R/W) = ACBIASCOUNTSTATUS for the primary LCD output generates an interrupt when it occurs
4	VPSYNCLOST_EN	R/W	0h	Synchronization Lost for Video Port. 0h (R/W) = SyncLost for the primary VP output is masked 1h (R/W) = SyncLost for the primary VP output generates an interrupt when it occurs
3	VPPROGRAMMEDLINENUMBER_EN	R/W	0h	Programmed Line Number. It indicates that the scan of the display has reached the programmed user line number. 0h (R/W) = ProgrammedLineNumber is masked 1h (R/W) = ProgrammedLineNumber generates an interrupt when it occurs
2	VPVSYNC_ODD_EN	R/W	0h	VSYNC for odd field from interlace mode only. 0h (R/W) = EVSYNC_ODD for the VP output is masked 1h (R/W) = EVSYNC_ODD for the VP output generates an interrupt when it occurs
1	VPVSYNC_EN	R/W	0h	Vertical Synchronization for VP. 0h (R/W) = VSYNC for the primary VP output is masked 1h (R/W) = VSYNC for the primary VP output generates an interrupt when it occurs

Table 11-660. DISPC_VP1_IRQENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	VPFRAMEDONE_EN	R/W	0h	Frame Done for Video Port. VP output has been disabled by user. All the data have been sent. 0h (R/W) = FrameDone for the primary VP output is masked 1h (R/W) = FrameDone for the primary VP output generates an interrupt when it occurs

Table 11-661. Register Call Summary for DISPC_VP1_IRQENABLE

DISPC_VP1 Registers • DISPC_VP1 Registers: [0] • DISPC_VP1_IRQENABLE Register (Offset = 3Ch) [reset = 0h]: [0]
DSS Functional Description • DISPC Interrupt Requests: [0]

11.3.5.5.9 DISPC_VP1_IRQSTATUS Register (Offset = 40h) [reset = 0h]

DISPC_VP1_IRQSTATUS is shown in Figure 11-289 and described in Table 11-663.

Return to Summary Table.

This register regroups all the status of the module internal events that generate an interrupt. Write 1 to a given bit resets this bit

Table 11-662. DISPC_VP1_IRQSTATUS Instances

Instance	Physical Address
DISPC_VP1	0255 AC40h

Figure 11-289. DISPC_VP1_IRQSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		ACBIASCOUNTSTATUS_IRQ	VPSYNCLOST_IRQ	VPPROGRAMMEDLINENUMBER_IRQ	VPVSYNC_ODD_IRQ	VPVSYNC_IRQ	VPFRAMEDONE_IRQ
R-0h		R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-663. DISPC_VP1_IRQSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	ACBIASCOUNTSTATUS_IRQ	R/W1C	0h	AC BIAS transition counter has decremented to zero 0h (R/W) = READS: Event is false. WRITES: Status bit unchanged. 1h (R/W) = READS: Event is true (pending). WRITES: Status bit is reset.
4	VPSYNCLOST_IRQ	R/W1C	0h	Synchronization Lost on VP output. The required data are not output at the correct time due to too short blanking periods or stall of at least one pipelines associated with VP output. 0h (R/W) = READS: Event is false. WRITES: Status bit unchanged. 1h (R/W) = READS: Event is true (pending). WRITES: Status bit is reset.
3	VPPROGRAMMEDLINENUMBER_IRQ	R/W1C	0h	Programmed Line Number. It indicates that the scan of the display has reached the programmed user line number. 0h (R/W) = READS: Event is false. WRITES: Status bit unchanged. 1h (R/W) = READS: Event is true (pending). WRITES: Status bit is reset.
2	VPVSYNC_ODD_IRQ	R/W1C	0h	VSYNC for odd field from interlace mode only. 0h (R/W) = READS: Event is false. WRITES: Status bit unchanged. 1h (R/W) = READS: Event is true (pending). WRITES: Status bit is reset.

Table 11-663. DISPC_VP1_IRQSTATUS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	VPVSYNC_IRQ	R/W1C	0h	Vertical Synchronization for VP output. It is used as VSYNC_EVEN in case of interlace mode. 0h (R/W) = READS: Event is false. WRITES: Status bit unchanged. 1h (R/W) = READS: Event is true (pending). WRITES: Status bit is reset.
0	VPFRAMEDONE_IRQ	R/W1C	0h	Frame Done for VP. VP output has been disabled by user. All the data have been sent. 0h (R/W) = READS: Event is false. WRITES: Status bit unchanged. 1h (R/W) = READS: Event is true (pending). WRITES: Status bit is reset.

Table 11-664. Register Call Summary for DISPC_VP1_IRQSTATUS

DISPC_VP1 Registers • DISPC_VP1 Registers: [0] • DISPC_VP1_IRQSTATUS Register (Offset = 40h) [reset = 0h]: [0]
DSS Functional Description • DISPC Interrupt Requests: [0]

11.3.5.5.10 DISPC_VP1_LINE_NUMBER Register (Offset = 44h) [reset = 0h]

DISPC_VP1_LINE_NUMBER is shown in Figure 11-290 and described in Table 11-666.

Return to Summary Table.

The control register indicates the panel display line number for the interrupt and the DMA request. Shadow register.

Table 11-665. DISPC_VP1_LINE_NUMBER Instances

Instance	Physical Address
DISPC_VP1	0255 AC44h

Figure 11-290. DISPC_VP1_LINE_NUMBER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				LINENUMBER											
R-0h																				R/W-0h											

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-666. DISPC_VP1_LINE_NUMBER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11-0	LINENUMBER	R/W	0h	Display panel line number programming. Line number defines the line on which the programmable interrupt is generated and the DMA request occurs.

Table 11-667. Register Call Summary for DISPC_VP1_LINE_NUMBER

DISPC_VP1 Registers • DISPC_VP1 Registers: [0] • DISPC_VP1_LINE_NUMBER Register (Offset = 44h) [reset = 0h]: [0]
DSS Functional Description • DISPC DMA Requests: [0]

11.3.5.5.11 DISPC_VP1_POL_FREQ Register (Offset = 4Ch) [reset = 0h]

DISPC_VP1_POL_FREQ is shown in Figure 11-291 and described in Table 11-669.

Return to Summary Table.

The register configures the signal configuration. Shadow register.

Table 11-668. DISPC_VP1_POL_FREQ Instances

Instance	Physical Address
DISPC_VP1	0255 AC4Ch

Figure 11-291. DISPC_VP1_POL_FREQ Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					ALIGN	ONOFF	RF
R-0h					R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
IEO	IPC	IHS	IVS	ACBI			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h			
7	6	5	4	3	2	1	0
ACB							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-669. DISPC_VP1_POL_FREQ Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0
18	ALIGN	R/W	0h	Defines the alignment betwwen HSYNC and VSYNC assertion. 0h (R/W) = VSYNC and HSYNC are not aligned 1h (R/W) = VSYNC and HSYNC assertions are aligned.
17	ONOFF	R/W	0h	HSYNC/VSYNC Pixel clock Control On/Off 0h (R/W) = HSYNC and VSYNC are driven on opposite edges of pixel clock than pixel data 1h (R/W) = HSYNC and VSYNC are driven according to bit 16
16	RF	R/W	0h	Program HSYNC/VSYNC Rise or Fall 0h (R/W) = HSYNC and VSYNC are driven on falling edge of pixel clock (if bit 17 set to 1) 1h (R/W) = HSYNC and VSYNC are driven on rising edge of pixel clock (if bit 17 set to 1)
15	IEO	R/W	0h	Invert output enable 0h (R/W) = Ac-bias is active high (active display mode) 1h (R/W) = Ac-bias is active low (active display mode)
14	IPC	R/W	0h	Invert pixel clock 0h (R/W) = Data is driven on the VP data lines on the rising-edge of the pixel clock 1h (R/W) = Data is driven on the VP data lines on the falling-edge of the pixel clock
13	IHS	R/W	0h	Invert HSYNC 0h (R/W) = Line clock pin is active high and inactive low 1h (R/W) = Line clock pin is active low and inactive high

Table 11-669. DISPC_VP1_POL_FREQ Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	IVS	R/W	0h	Invert VSYNC 0h (R/W) = Frame clock pin is active high and inactive low 1h (R/W) = Frame clock pin is active low and inactive high
11-8	ACBI	R/W	0h	AC Bias Pin transitions per interrupt Value (from 0 to 15) used to specify the number of AC Bias pin transitions 0h (R/W) = Line clock pin is active high and inactive low 1h (R/W) = Line clock pin is active low and inactive high
7-0	ACB	R/W	0h	AC Bias Pin Frequency Value (from 0 to 255) used to specify the number of line clocks to count before transitioning the AC Bias pin. This pin is used to periodically invert the polarity of the power supply to prevent DC charge build-up within the display. 0h (R/W) = Line clock pin is active high and inactive low 1h (R/W) = Line clock pin is active low and inactive high

Table 11-670. Register Call Summary for DISPC_VP1_POL_FREQ

DISPC_VP1 Registers • DISPC_VP1 Registers: [0] • DISPC_VP1_POL_FREQ Register (Offset = 4Ch) [reset = 0h]: [0]
DSS Functional Description • DISPC VP1 Timing Generator and Panel Settings: [0][1][2][3][4][5]
DSS Environment • DISPC VP1 Active Marix Display Timing Diagrams: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25][26][27]

11.3.5.5.12 DISPC_VP1_SIZE_SCREEN Register (Offset = 50h) [reset = 0h]

DISPC_VP1_SIZE_SCREEN is shown in Figure 11-292 and described in Table 11-672.

Return to Summary Table.

The register configures the panel size (horizontal and vertical). Shadow register. A delta value is used to indicate if the odd field has same vertical size as the even field or +/- one line.

Table 11-671. DISPC_VP1_SIZE_SCREEN Instances

Instance	Physical Address
DISPC_VP1	0255 AC50h

Figure 11-292. DISPC_VP1_SIZE_SCREEN Register

31	30	29	28	27	26	25	24
RESERVED				LPP			
R-0h				R/W-0h			
23	22	21	20	19	18	17	16
LPP							
R/W-0h							
15	14	13	12	11	10	9	8
DELTA_LPP		RESERVED		PPL			
R/W-0h		R-0h		R/W-0h			
7	6	5	4	3	2	1	0
PPL							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-672. DISPC_VP1_SIZE_SCREEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-16	LPP	R/W	0h	Lines per panel Encoded value (from 1 to 4096) to specify the number of lines per panel (program to value minus one).
15-14	DELTA_LPP	R/W	0h	Indicates the delta size value of the odd field compared to the even field 0h (R/W) = same size 1h (R/W) = odd size = even size +1 2h (R/W) = Odd size = even size -1
13-12	RESERVED	R	0h	Reserved
11-0	PPL	R/W	0h	Pixels per line Encoded value (from 1 to 4096) to specify the number of pixels contains within each line on the display (program to value minus one). In STALL mode, any value is valid. In non STALL mode, only values multiple of 8 pixels are valiid.

Table 11-673. Register Call Summary for DISPC_VP1_SIZE_SCREEN

DISPC_VP1 Registers • DISPC_VP1 Registers: [0] • DISPC_VP1_SIZE_SCREEN Register (Offset = 50h) [reset = 0h]: [0]
DSS Functional Description • DISPC VP1 Configuration for TV Support: [0][1][2][3][4] • DISPC VP1 Timing Generator and Panel Settings: [0][1][2][3][4][5][6]
DSS Environment • DISPC VP1 Active Marix Display Timing Diagrams: [0][1]

11.3.5.5.13 DISPC_VP1_TIMING_H Register (Offset = 54h) [reset = 0h]

DISPC_VP1_TIMING_H is shown in Figure 11-293 and described in Table 11-675.

Return to Summary Table.

The register configures the timing logic for the HSYNC signal. Shadow register.

Table 11-674. DISPC_VP1_TIMING_H Instances

Instance	Physical Address
DISPC_VP1	0255 AC54h

Figure 11-293. DISPC_VP1_TIMING_H Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HBP												HFP												HSW							
R/W-0h												R/W-0h												R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-675. DISPC_VP1_TIMING_H Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	HBP	R/W	0h	Horizontal Back Porch. Encoded value (from 1 to 4096) to specify the number of pixel clock periods to add to the beginning of a line transmission before the first set of pixels is output to the displaydisplay (program to value minus one). When in BT mode and interlaced, this field corresponds to the vertical field blanking No 2 for Even Field.
19-8	HFP	R/W	0h	Horizontal front porch. Encoded value (from 1 to 4096) to specify the number of pixel clock periods to add to the end of a line transmission before line clock is asserted display (program to value minus one). When in BT mode and interlaced, this field corresponds to the vertical field blanking No 1 for Even Field.
7-0	HSW	R/W	0h	Horizontal synchronization pulse width Encoded value (from 1 to 256) to specify the number of pixel clock periods to pulse the line clock at the end of each line display (program to value minus one). When in BT mode, this field corresponds to the LSB 8-bits of the 12-bit horizontal blanking(BT_HBLANK[11:0]= {VSW[3:0],HSW[7:0]}).

Table 11-676. Register Call Summary for DISPC_VP1_TIMING_H

DISPC_VP1 Registers • DISPC_VP1 Registers: [0] • DISPC_VP1_TIMING_H Register (Offset = 54h) [reset = 0h]: [0]
DSS Functional Description • DISPC VP1 Timing Generator and Panel Settings: [0][1][2][3][4][5][6][7]
DSS Environment • DISPC VP1 Active Marix Display Timing Diagrams: [0][1][2]

11.3.5.5.14 DISPC_VP1_TIMING_V Register (Offset = 58h) [reset = 0h]

DISPC_VP1_TIMING_V is shown in Figure 11-294 and described in Table 11-678.

Return to Summary Table.

The register configures the timing logic for the VSYNC signal. Shadow register.

Table 11-677. DISPC_VP1_TIMING_V Instances

Instance	Physical Address
DISPC_VP1	0255 AC58h

Figure 11-294. DISPC_VP1_TIMING_V Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VBP												VFP												VSW							
R/W-0h												R/W-0h												R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-678. DISPC_VP1_TIMING_V Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	VBP	R/W	0h	Vertical back porch Encoded value (from 0 to 4095) to specify the number of line clock periods to add to the beginning of a frame. When in BT mode and interlaced, this field corresponds to the vertical field blanking No 2 for Odd Field. When in BT and in progressive mode, this field corresponds to the Vertical frame blanking No 2 . before the first set of pixels is output to the display.
19-8	VFP	R/W	0h	Vertical front porch Encoded value (from 0 to 4095) to specify the number of line clock periods to add to the end of each frame. When in BT mode and interlaced, this field corresponds to the vertical field blanking No 1 for Odd Field. When in BT and in progressive mode, this field corresponds to the Vertical frame blanking No 2 .
7-0	VSW	R/W	0h	Vertical synchronization pulse width Encoded value (from 1 to 256) to specify the number of line clock periods to pulse the frame clock (VSYNC) pin at the end of each frame after the end of frame wait (VFP) period elapses. Frame clock uses as VSYNC signal in active mode. When in BT mode, the lsb 4-bits of this field (VSW[3:0]) corresponds to the MSB 4-bits of the 12-bit horizontal blanking(BT_HBLANK= {VSW[3:0],HSW[7:0]}).

Table 11-679. Register Call Summary for DISPC_VP1_TIMING_V

DISPC_VP1 Registers • DISPC_VP1 Registers: [0] • DISPC_VP1_TIMING_V Register (Offset = 58h) [reset = 0h]: [0]
DSS Functional Description • DISPC VP1 Timing Generator and Panel Settings: [0][1][2][3][4][5][6][7][8][9]
DSS Environment • DISPC VP1 Active Marix Display Timing Diagrams: [0][1][2]

11.3.5.6 RFBI Registers

Table 11-681 lists the memory-mapped registers for the RFBI. All register offset addresses not listed in Table 11-681 should be considered as reserved locations and the register contents should not be modified.

Table 11-680. RFBI Instances

Instance	Base Address
RFBI	0254 6000h

Table 11-681. RFBI Registers

Offset	Acronym	Register Name	RFBI Physical Address	Section
0h	RFBI_REVISION	This register contains the IP revision code.	0254 6000h	Section 11.3.5.6.1
10h	RFBI_SYSCONFIG	This register controls various parameters of the OCP interface.	0254 6010h	Section 11.3.5.6.2
14h	RFBI_SYSSTATUS	This register provides status information about the module, excluding the interrupt status information.	0254 6014h	Section 11.3.5.6.3
40h	RFBI_CONTROL	The register configures the RFBI module.	0254 6040h	Section 11.3.5.6.4
44h	RFBI_PIXEL_CNT	The register configures the RFBI pixel count value.	0254 6044h	Section 11.3.5.6.5
48h	RFBI_LINE_NUMBER	The register configures the number of lines to synchronize the beginning of the transfer.	0254 6048h	Section 11.3.5.6.6
4Ch	RFBI_CMD	The register configures the RFBI command.	0254 604Ch	Section 11.3.5.6.7
50h	RFBI_PARAM	The register configures the RFBI parameter.	0254 6050h	Section 11.3.5.6.8
54h	RFBI_DATA	The register configures the RFBI data.	0254 6054h	Section 11.3.5.6.9
58h	RFBI_READ	The register configures the RFBI read.	0254 6058h	Section 11.3.5.6.10
5Ch	RFBI_STATUS	The register configures the RFBI status.	0254 605Ch	Section 11.3.5.6.11
60h	RFBI_CONFIG__0	The control register sets the configuration for the LCD#0 and LCD#1.	0254 6060h	Section 11.3.5.6.12
64h	RFBI_ONOFF_TIME__0	The control register configures the RFBI timings for the LCD#0 and LCD#1.	0254 6064h	Section 11.3.5.6.13
68h	RFBI_CYCLE_TIME__0	The control register configures the RFBI timings for the LCD#0 and LCD#1.	0254 6068h	Section 11.3.5.6.14
6Ch	RFBI_DATA_CYCLE1__0	The control register configures the RFBI data format for 1st cycle for the LCD#0 and LCD#1.	0254 606Ch	Section 11.3.5.6.15
70h	RFBI_DATA_CYCLE2__0	The control register configures the RFBI data format for 2nd cycle for the LCD#0 and LCD#1.	0254 6070h	Section 11.3.5.6.16
74h	RFBI_DATA_CYCLE3__0	The control register configures the RFBI data format for 3rd cycle for the LCD#0 and LCD#1.	0254 6074h	Section 11.3.5.6.17
78h	RFBI_CONFIG__1	The control register sets the configuration for the LCD#0 and LCD#1.	0254 6078h	Section 11.3.5.6.18
7Ch	RFBI_ONOFF_TIME__1	The control register configures the RFBI timings for the LCD#0 and LCD#1.	0254 607Ch	Section 11.3.5.6.19
80h	RFBI_CYCLE_TIME__1	The control register configures the RFBI timings for the LCD#0 and LCD#1.	0254 6080h	Section 11.3.5.6.20
84h	RFBI_DATA_CYCLE1__1	The control register configures the RFBI data format for 1st cycle for the LCD#0 and LCD#1.	0254 6084h	Section 11.3.5.6.21
88h	RFBI_DATA_CYCLE2__1	The control register configures the RFBI data format for 2nd cycle for the LCD#0 and LCD#1.	0254 6088h	Section 11.3.5.6.22
8Ch	RFBI_DATA_CYCLE3__1	The control register configures the RFBI data format for 3rd cycle for the LCD#0 and LCD#1.	0254 608Ch	Section 11.3.5.6.23
90h	RFBI_VSYNC_WIDTH	The register configures the RFBI VSYNC minimum pulse width.	0254 6090h	Section 11.3.5.6.24

Table 11-681. RFBI Registers (continued)

Offset	Acronym	Register Name	RFBI Physical Address	Section
94h	RFBI_HSYNC_WIDTH	The register configures the RFBI HSYNC minimum pulse width.	0254 6094h	Section 11.3.5.6.25

11.3.5.6.1 RFBI_REVISION Register (Offset = 0h) [reset = 10h]

Register mask: FFFFFFFFh

RFBI_REVISION is shown in Figure 11-295 and described in Table 11-683.

Return to Summary Table.

This register contains the IP revision code.

Table 11-682. RFBI_REVISION Instances

Instance	Physical Address
RFBI	0254 6000h

Figure 11-295. RFBI_REVISION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							
R-0h																								R-10h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-683. RFBI_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reads returns 0
7-0	REV	R	10h	TI internal data. Identifies revision of peripheral.

Table 11-684. Register Call Summary for RFBI_REVISION

RFBI Registers
- RFBI_REVISION Register (Offset = 0h) [reset = 10h]: [0]
- RFBI Registers: [0]

11.3.5.6.2 RFBI_SYSCONFIG Register (Offset = 10h) [reset = 1h]

Register mask: FFFFFFFFh

RFBI_SYSCONFIG is shown in Figure 11-296 and described in Table 11-686.

Return to Summary Table.

This register controls various parameters of the OCP interface.

Table 11-685. RFBI_SYSCONFIG Instances

Instance	Physical Address
RFBI	0254 6010h

Figure 11-296. RFBI_SYSCONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	SIDLEMODE		RESERVED	SOFTRESET	AUTOIDLE
R-0h	R-0h	R-0h	R/W-0h		R-0h	R/W-0h	R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-686. RFBI_SYSCONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Write 0's for future compatibility. Reads returns 0.
6	RESERVED	R	0h	Write 0's for future compatibility. Reads returns 0.
5	RESERVED	R	0h	Write 0's for future compatibility. Reads returns 0.
4-3	SIDLEMODE	R/W	0h	Slave interface power management, Idle req/ack control. 0h (R/W) = Force-idle. An idle request is acknowledged unconditionally. 1h (R/W) = No-idle. An idle request is never acknowledged. **Feature is not supported in this family of devices.** 2h (R/W) = Smart-idle. Acknowledgement to an idle request is given based on the internal activity of the module.
2	RESERVED	R	0h	Write 0's for future compatibility. Reads returns 0.
1	SOFTRESET	R/W	0h	Software reset. Sets this bit to 1 to trigger a module reset. The bit is automatically reset by the hardware. During reads, it always returns 0. 0h (R/W) = Normal mode 1h (R/W) = The module is reset
0	AUTOIDLE	R/W	1h	Internal clock gating strategy (OCP clock and display controller clock) 0h (R/W) = OCP clock and display controller clock are free-running 1h (R/W) = Automatic clock gating strategy is applied for the OCP clock and display controller clock, based on the OCP interface and internal activity.

Table 11-687. Register Call Summary for RFBI_SYSCONFIG

RFBI Registers • RFBI Registers: [0] • RFBI_SYSSTATUS Register (Offset = 14h) [reset = 1h]: [0] • RFBI_SYSCONFIG Register (Offset = 10h) [reset = 1h]: [0]
DSS Functional Description • RFBI Software Reset: [0][1]

11.3.5.6.3 RFBI_SYSSTATUS Register (Offset = 14h) [reset = 1h]

Register mask: FFFFFFFFh

RFBI_SYSSTATUS is shown in Figure 11-297 and described in Table 11-689.

Return to Summary Table.

This register provides status information about the module, excluding the interrupt status information.

Table 11-688. RFBI_SYSSTATUS Instances

Instance	Physical Address
RFBI	0254 6014h

Figure 11-297. RFBI_SYSSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						BUSYRFBIDATA	BUSY
R-0h						R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED							RESETDONE
R-0h							R-1h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-689. RFBI_SYSSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved for module-specific status information. Read returns 0.
9	BUSYRFBIDATA	R	0h	Data are pending to be processed from internal FIFO. 0h (R) = No data pending 1h (R) = Some data are pending
8	BUSY	R	0h	OCP Slave port busy status bit 0h (R) = The access to the following register is not stall: RFBI_CMD, RFBI_STATUS, RFBI_PARAM, RFBI_READ. 1h (R) = The access to any of the following registers is stall: RFBI_CMD, RFBI_STATUS, RFBI_PARAM, RFBI_READ.
7-1	RESERVED	R	0h	Reserved for OCP socket status information. Read returns 0.
0	RESETDONE	R	1h	Internal reset monitoring. It can be used to determine when a HW reset is completed or when a SW reset is completed (software has set RFBI_SYSCONFIG.SOFTRESET to 1). 0h (R) = Internal module reset is on-going 1h (R) = Reset completed

Table 11-690. Register Call Summary for RFBI_SYSSTATUS

RFBI Registers
- RFBI Registers: [0]
- RFBI_SYSSTATUS Register (Offset = 14h) [reset = 1h]: [0]

Table 11-690. Register Call Summary for RFBI_SYSSTATUS (continued)

DSS Functional Description • RFBI Software Reset: [0] • RFBI State-Machine: [0][1][2][3][4][5][6][7][8] • RFBI Hardware Status Features: [0][1]

11.3.5.6.4 RFBI_CONTROL Register (Offset = 40h) [reset = 2h]

Register mask: FFFFFFFFh

RFBI_CONTROL is shown in Figure 11-298 and described in Table 11-692.

Return to Summary Table.

The register configures the RFBI module.

Table 11-691. RFBI_CONTROL Instances

Instance	Physical Address
RFBI	0254 6040h

Figure 11-298. RFBI_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							SMART_DMA_REQ
R-0h							R/W-0h
7	6	5	4	3	2	1	0
DISABLE_DMA_REQ	HIGHTHRESHOLD		ITE	CONFIGSELECT		BYPASSMODE	ENABLE
R/W-0h	R/W-0h		R/W-0h	R/W-0h		R/W-1h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-692. RFBI_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Write 0's for future compatibility. Reads returns 0.
8	SMART_DMA_REQ	R/W	0h	Smart DMA request 0h (R/W) = The dmareq is asserted and de-asserted depending on FIFO space even if MIdlereq is high in smart idle/no-idle mode and the entire burst gets error responses from the module. 1h (R/W) = The dmareq is de-asserted after 2 clk cycles if it has been asserted for more than or equal to 2 clk cycles and MIdlereq is high in smart idle or no idle mode. No more burst requests will be given even if the space is available in the FIFO.
7	DISABLE_DMA_REQ	R/W	0h	Disable DMA request 0h (R/W) = The dmareq is enabled and the signal is generated based on the space available and the request coming into the data register 1h (R/W) = The dmareq is disabled and the signal is not generated at all based on space in FIFO. It stays high until the DISABLE DMAREQ is high even if there is space in FIFO to take requests
6-5	HIGHTHRESHOLD	R/W	0h	Defines the FIFO high threshold used by HW to assert DMA request. Used only if data written to RFBI_DATA are sent using system DMA. 0h (R/W) = Size of the transfer of 4 words of 32-bit wide 1h (R/W) = Size of the transfer of 8 words of 32-bit wide 2h (R/W) = Size of the transfer of 16 words of 32-bit wide

Table 11-692. RFBI_CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	ITE	R/W	0h	Internal Trigger 0h (R/W) = H/W waits for ITE bit to be set if in internal trigger mode for the configuration in use. 1h (R/W) = User sets the ITE bit to start the transfer, when H/W takes into account the bit, the H/W resets it.
3-2	CONFIGSELECT	R/W	0h	Select the CS and configuration 0h (R/W) = No CS selected 1h (R/W) = CS0 selected and configuration #0 2h (R/W) = CS1 selected and configuration #1 3h (R/W) = CS0 and CS1 both selected (only the configuration for CS0 is used)
1	BYPASSMODE	R/W	1h	Bypass Mode 0h (R/W) = The bypass mode not selected 1h (R/W) = The bypass mode is selected
0	ENABLE	R/W	0h	Enable/Disable flag 0h (R/W) = Disable the RFBI module 1h (R/W) = Enable the RFBI module

Table 11-693. Register Call Summary for RFBI_CONTROL

RFBI Registers • RFBI_CONTROL Register (Offset = 40h) [reset = 2h]: [0] • RFBI Registers: [0] • RFBI_CONFIG__0 Register (Offset = 60h) [reset = 00310000h]: [0]
DSS Functional Description • RFBI Number of Pixels to Transfer: [0][1] • RFBI Trigger Mode: [0] • RFBI Configuration Selection: [0] • RFBI Stall Mechanism: [0][1][2] • RFBI Video Port Description: [0] • RFBI Timings: [0][1] • RFBI Internal Trigger Mode: [0][1][2] • RFBI Using DMA Request with Interconnect FIFO: [0][1][2][3][4][5] • RFBI DMA Requests: [0]
DSS Environment • RFBI Description of the Tearing Effect Pulse Signal: [0] • RFBI Environment: [0]

11.3.5.6.5 RFBI_PIXEL_CNT Register (Offset = 44h) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_PIXEL_CNT is shown in Figure 11-299 and described in Table 11-695.

Return to Summary Table.

The register configures the RFBI pixel count value.

Table 11-694. RFBI_PIXEL_CNT Instances

Instance	Physical Address
RFBI	0254 6044h

Figure 11-299. RFBI_PIXEL_CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PIXELCNT																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-695. RFBI_PIXEL_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PIXELCNT	R/W	0h	Pixel counter value. The SW indicates the number of pixels to transfer to the LCD panel frame buffer. The value is set when the module is disabled. During the transfer the HW decrements the register when a pixel has been sent to the RFB.

Table 11-696. Register Call Summary for RFBI_PIXEL_CNT

RFBI Registers • RFBI Registers: [0] • RFBI_PIXEL_CNT Register (Offset = 44h) [reset = 0h]: [0]
DSS Functional Description • RFBI Number of Pixels to Transfer: [0][1]

11.3.5.6.6 RFBI_LINE_NUMBER Register (Offset = 48h) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_LINE_NUMBER is shown in Figure 11-300 and described in Table 11-698.

Return to Summary Table.

The register configures the number of lines to synchronize the beginning of the transfer.

Table 11-697. RFBI_LINE_NUMBER Instances

Instance	Physical Address
RFBI	0254 6048h

Figure 11-300. RFBI_LINE_NUMBER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					LINENUMBER										
R-0h																					R/W-0h										

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-698. RFBI_LINE_NUMBER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Write 0's for future compatibility. Reads returns 0.
10-0	LINENUMBER	R/W	0h	Programmable line number Line number from 0 to 2047. Number of HSYNC after the VSYNC occurs before the beginning of the transfer.

Table 11-699. Register Call Summary for RFBI_LINE_NUMBER

RFBI Registers • RFBI_LINE_NUMBER Register (Offset = 48h) [reset = 0h]: [0] • RFBI Registers: [0] • RFBI_CONFIG__0 Register (Offset = 60h) [reset = 00310000h]: [0][1] • RFBI_CONFIG__1 Register (Offset = 78h) [reset = 00310000h]: [0][1]
DSS Functional Description • RFBI External Trigger Mode: [0]

11.3.5.6.7 RFBI_CMD Register (Offset = 4Ch) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_CMD is shown in Figure 11-301 and described in Table 11-701.

Return to Summary Table.

The register configures the RFBI command

Table 11-700. RFBI_CMD Instances

Instance	Physical Address
RFBI	0254 604Ch

Figure 11-301. RFBI_CMD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CMD															
R-0h																W-0h															

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 11-701. RFBI_CMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Write 0's for future compatibility. Reads returns 0.
15-0	CMD	W	0h	Command Value. 8/9/12/16 bit value depending on the parallelMode. [7:0] 8-bit DataType [8:0] 9-bit DataType [11:0] 12-bit DataType [15:0] 16-bit DataType

Table 11-702. Register Call Summary for RFBI_CMD

RFBI Registers • RFBI_CMD Register (Offset = 4Ch) [reset = 0h]: [0] • RFBI Registers: [0] • RFBI_SYSSTATUS Register (Offset = 14h) [reset = 1h]: [0][1]
DSS Functional Description • RFBI State-Machine: [0][1][2] • RFBI Hardware Status Features: [0]
DSS Environment • RFBI Environment: [0][1]

11.3.5.6.8 RFBI_PARAM Register (Offset = 50h) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_PARAM is shown in Figure 11-302 and described in Table 11-704.

Return to Summary Table.

The register configures the RFBI parameter.

Table 11-703. RFBI_PARAM Instances

Instance	Physical Address
RFBI	0254 6050h

Figure 11-302. RFBI_PARAM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PARAM															
R-0h																W-0h															

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 11-704. RFBI_PARAM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Write 0's for future compatibility. Reads returns 0.
15-0	PARAM	W	0h	Param Value. 8/9/12/16 bit value depending on the parallelMode. [7:0] 8-bit DataType [8:0] 9-bit DataType [11:0] 12-bit DataType [15:0] 16-bit DataType

Table 11-705. Register Call Summary for RFBI_PARAM

RFBI Registers • RFBI_PARAM Register (Offset = 50h) [reset = 0h]: [0] • RFBI Registers: [0] • RFBI_SYSSTATUS Register (Offset = 14h) [reset = 1h]: [0][1]
DSS Functional Description • RFBI State-Machine: [0][1][2] • RFBI Hardware Status Features: [0]
DSS Environment • RFBI Environment: [0][1]

11.3.5.6.9 RFBI_DATA Register (Offset = 54h) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_DATA is shown in Figure 11-303 and described in Table 11-707.

Return to Summary Table.

The register configures the RFBI data.

Table 11-706. RFBI_DATA Instances

Instance	Physical Address
RFBI	0254 6054h

Figure 11-303. RFBI_DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 11-707. RFBI_DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	W	0h	Data value. 12/16/18/24/2x16 bit value depending on the DataType. [11:0] 12-bit DataType [15:0] 16-bit DataType [17:0] 18-bit DataType [23:0] 24-bit DataType [31:0] 2x16-bit DataType

Table 11-708. Register Call Summary for RFBI_DATA

RFBI Registers • RFBI_CONTROL Register (Offset = 40h) [reset = 2h]: [0] • RFBI Registers: [0] • RFBI_DATA Register (Offset = 54h) [reset = 0h]: [0]
DSS Functional Description • RFBI State-Machine: [0][1][2][3][4] • RFBI Using DMA Request with Interconnect FIFO: [0][1][2] • RFBI FIFO Description: [0][1]
DSS Environment • RFBI Environment: [0][1]

11.3.5.6.10 RFBI_READ Register (Offset = 58h) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_READ is shown in Figure 11-304 and described in Table 11-710.

Return to Summary Table.

The register configures the RFBI read.

Table 11-709. RFBI_READ Instances

Instance	Physical Address
RFBI	0254 6058h

Figure 11-304. RFBI_READ Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																READ															
R-0h																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-710. RFBI_READ Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Write 0's for future compatibility. Reads returns 0.
15-0	READ	R/W	0h	Read Value. 8/9/12/16 bit value depending on the parallelMode. [7:0] 8-bit DataType [8:0] 9-bit DataType [11:0] 12-bit DataType [15:0] 16-bit DataType

Table 11-711. Register Call Summary for RFBI_READ

RFBI Registers • RFBI Registers: [0] • RFBI_READ Register (Offset = 58h) [reset = 0h]: [0] • RFBI_SYSSTATUS Register (Offset = 14h) [reset = 1h]: [0][1]
DSS Functional Description • RFBI State-Machine: [0][1][2] • RFBI Hardware Status Features: [0]
DSS Environment • RFBI Environment: [0][1]

11.3.5.6.11 RFBI_STATUS Register (Offset = 5Ch) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_STATUS is shown in Figure 11-305 and described in Table 11-713.

Return to Summary Table.

The register configures the RFBI status.

Table 11-712. RFBI_STATUS Instances

Instance	Physical Address
RFBI	0254 605Ch

Figure 11-305. RFBI_STATUS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																STATUS															
R-0h																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-713. RFBI_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Write 0's for future compatibility. Reads returns 0.
15-0	STATUS	R/W	0h	Status value. 8/9/12/16 bit value depending on the parallelMode. [7:0] 8-bit DataType [8:0] 9-bit DataType [11:0] 12-bit DataType [15:0] 16-bit DataType

Table 11-714. Register Call Summary for RFBI_STATUS

RFBI Registers • RFBI_STATUS Register (Offset = 5Ch) [reset = 0h]: [0] • RFBI Registers: [0] • RFBI_SYSSTATUS Register (Offset = 14h) [reset = 1h]: [0][1]
DSS Functional Description • RFBI State-Machine: [0][1][2] • RFBI Hardware Status Features: [0]
DSS Environment • RFBI Environment: [0][1]

11.3.5.6.12 RFBI_CONFIG__0 Register (Offset = 60h) [reset = 00310000h]

Register mask: FFFFFFFFh

RFBI_CONFIG__0 is shown in Figure 11-306 and described in Table 11-716.

Return to Summary Table.

The control register sets the configuration for the LCD#0 and LCD#1.

Table 11-715. RFBI_CONFIG__0 Instances

Instance	Physical Address
RFBI	0254 6060h

Figure 11-306. RFBI_CONFIG__0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED		HSYNCPOLARITY	TE_VSYNC_POLARITY	CSPOLARITY	WEPOLARITY	REPOLARITY	A0POLARITY
R-0h		R/W-1h	R/W-1h	R/W-0h	R/W-0h	R/W-0h	R/W-1h
15	14	13	12	11	10	9	8
RESERVED			UNUSEDBITS		CYCLEFORMAT		OCPFORMAT
R-0h			R/W-0h		R/W-0h		R/W-0h
7	6	5	4	3	2	1	0
OCPFORMAT	DATATYPE		TIMEGRANULARITY	TRIGGERMODE		PARALLELMODE	
R/W-0h	R/W-0h		R/W-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-716. RFBI_CONFIG__0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Write 0's for future compatibility. Reads returns 0.
21	HSYNCPOLARITY	R/W	1h	HSYNC polarity 0h (R/W) = HSYNC active low 1h (R/W) = HSYNC active high
20	TE_VSYNC_POLARITY	R/W	1h	TE or VSYNC Polarity 0h (R/W) = active low 1h (R/W) = active high
19	CSPOLARITY	R/W	0h	CS Polarity 0h (R/W) = CS active low 1h (R/W) = CS active high
18	WEPOLARITY	R/W	0h	WE Polarity 0h (R/W) = active low 1h (R/W) = active high
17	REPOLARITY	R/W	0h	RE Polarity 0h (R/W) = active low 1h (R/W) = active high
16	A0POLARITY	R/W	1h	A0 Polarity 0h (R/W) = A0 active low 1h (R/W) = A0 active high
15-13	RESERVED	R	0h	Write 0's for future compatibility. Reads returns 0.

Table 11-716. RFBI_CONFIG__0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12-11	UNUSEDBITS	R/W	0h	State of unused bits 0h (R/W) = low level (0) 1h (R/W) = high level (1) 2h (R/W) = unchanged from previous state
10-9	CYCLEFORMAT	R/W	0h	Cycle format 0h (R/W) = 1 cycle for 1 pixel 1h (R/W) = 2 cycles for 1 pixel 2h (R/W) = 3 cycles for 1 pixel 3h (R/W) = 3 cycles for 2 pixels
8-7	OCPFORMAT	R/W	0h	OCP Write Access format 0h (R/W) = 1 pixel per OCP access to the register data 2h (R/W) = 2 pixels per OCP access to the register data with 1st pixel at the position [15:0] 3h (R/W) = 2 pixels per OCP access to the register data with 1st pixel at the position [31:16]
6-5	DATATYPE	R/W	0h	Data type from the display controller and OCP slave port 0h (R/W) = 12-bit 1h (R/W) = 16-bit 2h (R/W) = 18-bit 3h (R/W) = 24-bit
4	TIMEGRANULARITY	R/W	0h	Multiplies signal timing latencies by two 0h (R/W) = x2 latencies disabled 1h (R/W) = x2 latencies enabled
3-2	TRIGGERMODE	R/W	0h	Trigger Mode 0h (R/W) = 00 Internal trigger mode (RFBI_CONTROL.ITE bit mode) 1h (R/W) = Tearing Effect Signal, RFBI_TEVSYNC0 is used with programmable line counter defined in RFBI_LINE_NUMBER register 2h (R/W) = External trigger mode (RFBI_TEVSYNC/RFBI_HSYNC with programmable line counter defined in RFBI_LINE_NUMBER.LINENUMBER bit-field)
1-0	PARALLELMODE	R/W	0h	Parallel Mode 0h (R/W) = 8-bit parallel output interface selected 1h (R/W) = 9-bit parallel output interface selected 2h (R/W) = 12-bit parallel output interface selected 3h (R/W) = 16-bit parallel output interface selected

Table 11-717. Register Call Summary for RFBI_CONFIG__0

RFBI Registers • RFBI Registers: [0] • RFBI_CONFIG__0 Register (Offset = 60h) [reset = 00310000h]: [0]
DSS Functional Description • RFBI Read/Write: [0][1] • RFBI Configuration Selection: [0] • RFBI Stall Mechanism: [0] • RFBI State-Machine: [0] • RFBI Timings: [0][1] • RFBI Trigger Mode: [0] • RFBI Unmodified Bits: [0] • RFBI Input Formats: [0] • RFBI Cycle Mode Selection: [0][1][2][3][4] • RFBI Internal Trigger Mode: [0]

Table 11-717. Register Call Summary for RFBI_CONFIG__0 (continued)

DSS Environment • RFBI Description of the Tearing Effect Pulse Signal: [0][1][2][3] • RFBI Environment: [0][1][2]

11.3.5.6.13 RFBI_ONOFF_TIME__0 Register (Offset = 64h) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_ONOFF_TIME__0 is shown in Figure 11-307 and described in Table 11-719.

Return to Summary Table.

The control register configures the RFBI timings for the LCD#0 and LCD#1.

Table 11-718. RFBI_ONOFF_TIME__0 Instances

Instance	Physical Address
RFBI	0254 6064h

Figure 11-307. RFBI_ONOFF_TIME__0 Register

31	30	29	28	27	26	25	24
RESERVED		REOFFTIME					
R-0h		R/W-0h					
23	22	21	20	19	18	17	16
REONTIME				WEOFFTIME			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
WEOFFTIME		WEONTIME				CSOFFTIME	
R/W-0h		R/W-0h				R/W-0h	
7	6	5	4	3	2	1	0
CSOFFTIME				CSONTIME			
R/W-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-719. RFBI_ONOFF_TIME__0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
29-24	REOFFTIME	R/W	0h	Read Enable de-assertion time from start access time. Number of OCPClk cycles
23-20	REONTIME	R/W	0h	Read Enable assertion time from start access time. Number of OCPClk cycles
19-14	WEOFFTIME	R/W	0h	Write Enable de-assertion time from start access time. Number of OCPClk cycles
13-10	WEONTIME	R/W	0h	Write Enable assertion time from start access time. Number of OCPClk cycles
9-4	CSOFFTIME	R/W	0h	CS de-assertion time from start access time. Number of OCPClk cycles
3-0	CSONTIME	R/W	0h	CS assertion time from start access time. Number of OCPClk cycles

Table 11-720. Register Call Summary for RFBI_ONOFF_TIME__0

RFBI Registers • RFBI_ONOFF_TIME__0 Register (Offset = 64h) [reset = 0h]: [0] • RFBI Registers: [0]
DSS Functional Description • RFBI Timings: [0][1][2][3][4][5][6][7][8][9][10][11][12][13] • RFBI Configuration Selection: [0]
DSS Environment • RFBI Transaction Timing Diagrams: [0][1][2][3][4][5]

11.3.5.6.14 RFBI_CYCLE_TIME__0 Register (Offset = 68h) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_CYCLE_TIME__0 is shown in Figure 11-308 and described in Table 11-722.

Return to Summary Table.

The control register configures the RFBI timings for the LCD#0 and LCD#1.

Table 11-721. RFBI_CYCLE_TIME__0 Instances

Instance	Physical Address
RFBI	0254 6068h

Figure 11-308. RFBI_CYCLE_TIME__0 Register

31	30	29	28	27	26	25	24
RESERVED				ACCESSTIME			
R-0h				R/W-0h			
23	22	21	20	19	18	17	16
ACCESSTIME		WRENABLE	WWENABLE	RRENABLE	RWENABLE	CSPULSEWIDTH	
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
CSPULSEWIDTH				RECYCLETIME			
R/W-0h				R/W-0h			
7	6	5	4	3	2	1	0
RECYCLETIME		WECYCLETIME					
R/W-0h		R/W-0h					

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-722. RFBI_CYCLE_TIME__0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
27-22	ACCESSTIME	R/W	0h	Access Time. Number of OCPClk cycles.
21	WRENABLE	R/W	0h	Write to Read Pulse Width Enable (same CS) 0h (R/W) = CSPulseWidth does not apply on Write to Read access 1h (R/W) = CSPulseWidth applies on Write to Read access
20	WWENABLE	R/W	0h	Write to Write Pulse Width Enable (same CS) 0h (R/W) = CSPulseWidth does not apply on Write to Write access 1h (R/W) = CSPulseWidth applies on Write to Write access
19	RRENABLE	R/W	0h	Read to Read Pulse Width Enable (same CS) 0h (R/W) = CSPulseWidth does not apply on Read to Read access 1h (R/W) = CSPulseWidth applies on Read to Read access
18	RWENABLE	R/W	0h	Read to Write Pulse Width Enable (same CS). 0h (R/W) = CSPulseWidth does not apply on Read to Write access 1h (R/W) = CSPulseWidth applies on Read to Write access
17-12	CSPULSEWIDTH	R/W	0h	CS Pulse Width. Number of OCPClk cycles.
11-6	RECYCLETIME	R/W	0h	RE Cycle Time. Number of OCPClk cycles.
5-0	WECYCLETIME	R/W	0h	WE Cycle Time. Number of OCPClk cycles.

Table 11-723. Register Call Summary for RFBI_CYCLE_TIME__0

RFBI Registers • RFBI Registers: [0] • RFBI_CYCLE_TIME__0 Register (Offset = 68h) [reset = 0h]: [0]
DSS Functional Description • RFBI Timings: [0][1][2][3][4][5][6][7][8][9][10][11][12] • RFBI Configuration Selection: [0]
DSS Environment • RFBI Transaction Timing Diagrams: [0][1][2]

11.3.5.6.15 RFBI_DATA_CYCLE1__0 Register (Offset = 6Ch) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_DATA_CYCLE1__0 is shown in Figure 11-309 and described in Table 11-725.

Return to Summary Table.

The control register configures the RFBI data format for 1st cycle for the LCD#0 and LCD#1.

Table 11-724. RFBI_DATA_CYCLE1__0 Instances

Instance	Physical Address
RFBI	0254 606Ch

Figure 11-309. RFBI_DATA_CYCLE1__0 Register

31	30	29	28	27	26	25	24
RESERVED				BITALIGNMENTPIXEL2			
R-0h				R/W-0h			
23	22	21	20	19	18	17	16
RESERVED				NBBITSPIXEL2			
R-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED				BITALIGNMENTPIXEL1			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
RESERVED				NBBITSPIXEL1			
R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-725. RFBI_DATA_CYCLE1__0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
27-24	BITALIGNMENTPIXEL2	R/W	0h	Bit alignment. Alignment of the bits from pixel#2 on the output interface.
23-21	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
20-16	NBBITSPIXEL2	R/W	0h	Number of bits. Number of bits from the pixel #2 (value from 0 to 16 bits). The values from 17 to 31 are invalid.
15-12	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
11-8	BITALIGNMENTPIXEL1	R/W	0h	Bit alignment. Alignment of the bits from pixel#1 on the output interface.
7-5	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
4-0	NBBITSPIXEL1	R/W	0h	Number of bits. Number of bits from the pixel #1 (value from 0 to 16 bits). The values from 17 to 31 are invalid.

Table 11-726. Register Call Summary for RFBI_DATA_CYCLE1__0

RFBI Registers • RFBI Registers: [0] • RFBI_DATA_CYCLE1__0 Register (Offset = 6Ch) [reset = 0h]: [0]
DSS Functional Description • RFBI Configuration Selection: [0] • RFBI Cycle Mode Selection: [0]

11.3.5.6.16 RFBI_DATA_CYCLE2__0 Register (Offset = 70h) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_DATA_CYCLE2__0 is shown in Figure 11-310 and described in Table 11-728.

Return to Summary Table.

The control register configures the RFBI data format for 2nd cycle for the LCD#0 and LCD#1.

Table 11-727. RFBI_DATA_CYCLE2__0 Instances

Instance	Physical Address
RFBI	0254 6070h

Figure 11-310. RFBI_DATA_CYCLE2__0 Register

31	30	29	28	27	26	25	24
RESERVED				BITALIGNMENTPIXEL2			
R-0h				R/W-0h			
23	22	21	20	19	18	17	16
RESERVED			NBBITSPIXEL2				
R-0h			R/W-0h				
15	14	13	12	11	10	9	8
RESERVED				BITALIGNMENTPIXEL1			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
RESERVED			NBBITSPIXEL1				
R-0h			R/W-0h				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-728. RFBI_DATA_CYCLE2__0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
27-24	BITALIGNMENTPIXEL2	R/W	0h	Bit alignment. Alignment of the bits from pixel#2 on the output interface.
23-21	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
20-16	NBBITSPIXEL2	R/W	0h	Number of bits. Number of bits from the pixel #2 (value from 0 to 16 bits). The values from 17 to 31 are invalid.
15-12	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
11-8	BITALIGNMENTPIXEL1	R/W	0h	Bit alignment. Alignment of the bits from pixel#1 on the output interface.
7-5	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
4-0	NBBITSPIXEL1	R/W	0h	Number of bits. Number of bits from the pixel #1 (value from 0 to 16 bits). The values from 17 to 31 are invalid.

Table 11-729. Register Call Summary for RFBI_DATA_CYCLE2__0

RFBI Registers • RFBI_DATA_CYCLE2__0 Register (Offset = 70h) [reset = 0h]: [0] • RFBI Registers: [0]
DSS Functional Description • RFBI Configuration Selection: [0] • RFBI Cycle Mode Selection: [0]

11.3.5.6.17 RFBI_DATA_CYCLE3__0 Register (Offset = 74h) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_DATA_CYCLE3__0 is shown in Figure 11-311 and described in Table 11-731.

Return to Summary Table.

The control register configures the RFBI data format for 3rd cycle for the LCD#0 and LCD#1.

Table 11-730. RFBI_DATA_CYCLE3__0 Instances

Instance	Physical Address
RFBI	0254 6074h

Figure 11-311. RFBI_DATA_CYCLE3__0 Register

31	30	29	28	27	26	25	24
RESERVED				BITALIGNMENTPIXEL2			
R-0h				R/W-0h			
23	22	21	20	19	18	17	16
RESERVED				NBBITSPIXEL2			
R-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED				BITALIGNMENTPIXEL1			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
RESERVED				NBBITSPIXEL1			
R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-731. RFBI_DATA_CYCLE3__0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
27-24	BITALIGNMENTPIXEL2	R/W	0h	Bit alignment. Alignment of the bits from pixel#2 on the output interface.
23-21	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
20-16	NBBITSPIXEL2	R/W	0h	Number of bits. Number of bits from the pixel #2 (value from 0 to 16 bits). The values from 17 to 31 are invalid.
15-12	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
11-8	BITALIGNMENTPIXEL1	R/W	0h	Bit alignment. Alignment of the bits from pixel#1 on the output interface.
7-5	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
4-0	NBBITSPIXEL1	R/W	0h	Number of bits. Number of bits from the pixel #1 (value from 0 to 16 bits). The values from 17 to 31 are invalid.

Table 11-732. Register Call Summary for RFBI_DATA_CYCLE3__0

RFBI Registers • RFBI_DATA_CYCLE3__0 Register (Offset = 74h) [reset = 0h]: [0] • RFBI Registers: [0]
DSS Functional Description • RFBI Configuration Selection: [0] • RFBI Cycle Mode Selection: [0]

11.3.5.6.18 RFBI_CONFIG__1 Register (Offset = 78h) [reset = 00310000h]

Register mask: FFFFFFFFh

RFBI_CONFIG__1 is shown in Figure 11-312 and described in Table 11-734.

Return to Summary Table.

The control register sets the configuration for the LCD#0 and LCD#1.

Table 11-733. RFBI_CONFIG__1 Instances

Instance	Physical Address
RFBI	0254 6078h

Figure 11-312. RFBI_CONFIG__1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED		HSYNCPOLARITY	TE_VSYNC_POLARITY	CSPOLARITY	WEPOLARITY	REPOLARITY	A0POLARITY
R-0h		R/W-1h	R/W-1h	R/W-0h	R/W-0h	R/W-0h	R/W-1h
15	14	13	12	11	10	9	8
RESERVED			UNUSEDBITS		CYCLEFORMAT		OCPFORMAT
R-0h			R/W-0h		R/W-0h		R/W-0h
7	6	5	4	3	2	1	0
OCPFORMAT	DATATYPE		TIMEGRANULARITY	TRIGGERMODE		PARALLELMODE	
R/W-0h	R/W-0h		R/W-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-734. RFBI_CONFIG__1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Write 0's for future compatibility Reads return 0
21	HSYNCPOLARITY	R/W	1h	HSYNC polarity 0h (R/W) = HSYNC active low 1h (R/W) = HSYNC active high
20	TE_VSYNC_POLARITY	R/W	1h	TE or VSYNC Polarity 0h (R/W) = active low 1h (R/W) = active high
19	CSPOLARITY	R/W	0h	CS Polarity 0h (R/W) = CS active low 1h (R/W) = CS active high
18	WEPOLARITY	R/W	0h	WE Polarity 0h (R/W) = active low 1h (R/W) = active high
17	REPOLARITY	R/W	0h	RE Polarity 0h (R/W) = active low 1h (R/W) = active high
16	A0POLARITY	R/W	1h	A0 Polarity 0h (R/W) = A0 active low 1h (R/W) = A0 active high
15-13	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.

Table 11-734. RFBI_CONFIG__1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12-11	UNUSEDBITS	R/W	0h	State of unused bits 0h (R/W) = low level (0) 1h (R/W) = high level (1) 2h (R/W) = unchanged from previous state
10-9	CYCLEFORMAT	R/W	0h	Cycle format 0h (R/W) = 1 cycle for 1 pixel 1h (R/W) = 2 cycles for 1 pixel 2h (R/W) = 3 cycles for 1 pixel 3h (R/W) = 3 cycles for 2 pixels
8-7	OCPFORMAT	R/W	0h	OCP Write Access format 0h (R/W) = 1 pixel per OCP access to the register data 2h (R/W) = 2 pixels per OCP access to the register data with 1st pixel at the position [15:0] 3h (R/W) = 2 pixels per OCP access to the register data with 1st pixel at the position [31:16]
6-5	DATATYPE	R/W	0h	Data type from the display controller and OCP slave port 0h (R/W) = 12-bit 1h (R/W) = 16-bit 2h (R/W) = 18-bit 3h (R/W) = 24-bit
4	TIMEGRANULARITY	R/W	0h	Multiplies signal timing latencies by two 0h (R/W) = x2 latencies disabled 1h (R/W) = x2 latencies enabled
3-2	TRIGGERMODE	R/W	0h	Trigger Mode 0h (R/W) = 00 Internal trigger mode (ITE bit mode) 1h (R/W) = Tearing Effect Signal, RFBI_TE_VSYNC0 is used with programmable line counter defined in RFBI_LINE_NUMBER register 2h (R/W) = External trigger mode (RFB_TE_VSYNC/RFB_HSYNC with programmable line counter defined in RFBI_LINE_NUMBER.LINENUMBER)
1-0	PARALLELMODE	R/W	0h	Parallel Mode 0h (R/W) = 8-bit parallel output interface selected 1h (R/W) = 9-bit parallel output interface selected 2h (R/W) = 12-bit parallel output interface selected 3h (R/W) = 16-bit parallel output interface selected

Table 11-735. Register Call Summary for RFBI_CONFIG__1

RFBI Registers • RFBI Registers: [0] • RFBI_CONFIG__1 Register (Offset = 78h) [reset = 00310000h]: [0]
DSS Functional Description • RFBI Read/Write: [0][1] • RFBI Configuration Selection: [0] • RFBI Stall Mechanism: [0] • RFBI State-Machine: [0] • RFBI Timings: [0][1] • RFBI Trigger Mode: [0] • RFBI Unmodified Bits: [0] • RFBI Input Formats: [0] • RFBI Cycle Mode Selection: [0][1][2][3][4] • RFBI Internal Trigger Mode: [0]

Table 11-735. Register Call Summary for RFBI_CONFIG__1 (continued)

DSS Environment • RFBI Description of the Tearing Effect Pulse Signal: [0][1][2][3] • RFBI Environment: [0][1][2]

11.3.5.6.19 RFBI_ONOFF_TIME__1 Register (Offset = 7Ch) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_ONOFF_TIME__1 is shown in Figure 11-313 and described in Table 11-737.

Return to Summary Table.

The control register configures the RFBI timings for the LCD#0 and LCD#1.

Table 11-736. RFBI_ONOFF_TIME__1 Instances

Instance	Physical Address
RFBI	0254 607Ch

Figure 11-313. RFBI_ONOFF_TIME__1 Register

31	30	29	28	27	26	25	24
RESERVED		REOFFTIME					
R-0h		R/W-0h					
23	22	21	20	19	18	17	16
REONTIME				WEOFFTIME			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
WEOFFTIME		WEONTIME				CSOFFTIME	
R/W-0h		R/W-0h				R/W-0h	
7	6	5	4	3	2	1	0
CSOFFTIME				CSONTIME			
R/W-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-737. RFBI_ONOFF_TIME__1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
29-24	REOFFTIME	R/W	0h	Read Enable de-assertion time from start access time. Number of OCPClk cycles.
23-20	REONTIME	R/W	0h	Read Enable assertion time from start access time. Number of OCPClk cycles.
19-14	WEOFFTIME	R/W	0h	Write Enable de-assertion time from start access time. Number of OCPClk cycles.
13-10	WEONTIME	R/W	0h	Write Enable assertion time from start access time. Number of OCPClk cycles.
9-4	CSOFFTIME	R/W	0h	CS de-assertion time from start access time. Number of OCPClk cycles.
3-0	CSONTIME	R/W	0h	CS assertion time from start access time. Number of OCPClk cycles.

Table 11-738. Register Call Summary for RFBI_ONOFF_TIME__1

RFBI Registers • RFBI Registers: [0] • RFBI_ONOFF_TIME__1 Register (Offset = 7Ch) [reset = 0h]: [0]
DSS Functional Description • RFBI Timings: [0][1][2][3][4][5][6][7][8][9][10][11][12][13] • RFBI Configuration Selection: [0]
DSS Environment • RFBI Transaction Timing Diagrams: [0][1][2][3][4][5]

11.3.5.6.20 RFBI_CYCLE_TIME__1 Register (Offset = 80h) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_CYCLE_TIME__1 is shown in Figure 11-314 and described in Table 11-740.

Return to Summary Table.

The control register configures the RFBI timings for the LCD#0 and LCD#1.

Table 11-739. RFBI_CYCLE_TIME__1 Instances

Instance	Physical Address
RFBI	0254 6080h

Figure 11-314. RFBI_CYCLE_TIME__1 Register

31	30	29	28	27	26	25	24
RESERVED				ACCESSTIME			
R-0h				R/W-0h			
23	22	21	20	19	18	17	16
ACCESSTIME		WRENABLE	WWENABLE	RRENABLE	RWENABLE	CSPULSEWIDTH	
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
CSPULSEWIDTH				RECYCLETIME			
R/W-0h				R/W-0h			
7	6	5	4	3	2	1	0
RECYCLETIME		WECYCLETIME					
R/W-0h		R/W-0h					

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-740. RFBI_CYCLE_TIME__1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
27-22	ACCESSTIME	R/W	0h	Access Time. Number of OCPClk cycles
21	WRENABLE	R/W	0h	Write to Read Pulse Width Enable (same CS) 0h (R/W) = CSPulseWidth does not apply on Write to Read access 1h (R/W) = CSPulseWidth applies on Write to Read access
20	WWENABLE	R/W	0h	Write to Write Pulse Width Enable (same CS) 0h (R/W) = CSPulseWidth does not apply on Write to Write access 1h (R/W) = CSPulseWidth applies on Write to Write access
19	RRENABLE	R/W	0h	Read to Read Pulse Width Enable (same CS) 0h (R/W) = CSPulseWidth does not apply on Read to Read access 1h (R/W) = CSPulseWidth applies on Read to Read access
18	RWENABLE	R/W	0h	Read to Write Pulse Width Enable (same CS). 0h (R/W) = CSPulseWidth does not apply on Read to Write access 1h (R/W) = CSPulseWidth applies on Read to Write access
17-12	CSPULSEWIDTH	R/W	0h	CS Pulse. Width Number of OCPClk cycles.
11-6	RECYCLETIME	R/W	0h	RE Cycle. Time Number of OCPClk cycles.
5-0	WECYCLETIME	R/W	0h	WE Cycle. Time Number of OCPClk cycles.

Table 11-741. Register Call Summary for RFBI_CYCLE_TIME__1

RFBI Registers • RFBI_CYCLE_TIME__1 Register (Offset = 80h) [reset = 0h]: [0] • RFBI Registers: [0]
DSS Functional Description • RFBI Timings: [0][1][2][3][4][5][6][7][8][9][10][11][12] • RFBI Configuration Selection: [0]
DSS Environment • RFBI Transaction Timing Diagrams: [0][1][2]

11.3.5.6.21 RFBI_DATA_CYCLE1__1 Register (Offset = 84h) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_DATA_CYCLE1__1 is shown in Figure 11-315 and described in Table 11-743.

Return to Summary Table.

The control register configures the RFBI data format for 1st cycle for the LCD#0 and LCD#1.

Table 11-742. RFBI_DATA_CYCLE1__1 Instances

Instance	Physical Address
RFBI	0254 6084h

Figure 11-315. RFBI_DATA_CYCLE1__1 Register

31	30	29	28	27	26	25	24
RESERVED				BITALIGNMENTPIXEL2			
R-0h				R/W-0h			
23	22	21	20	19	18	17	16
RESERVED				NBBITSPIXEL2			
R-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED				BITALIGNMENTPIXEL1			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
RESERVED				NBBITSPIXEL1			
R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-743. RFBI_DATA_CYCLE1__1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
27-24	BITALIGNMENTPIXEL2	R/W	0h	Bit alignment. Alignment of the bits from pixel#2 on the output interface.
23-21	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
20-16	NBBITSPIXEL2	R/W	0h	Number of bits. Number of bits from the pixel #2 (value from 0 to 16 bits). The values from 17 to 31 are invalid.
15-12	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
11-8	BITALIGNMENTPIXEL1	R/W	0h	Bit alignment. Alignment of the bits from pixel#1 on the output interface.
7-5	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
4-0	NBBITSPIXEL1	R/W	0h	Number of bits. Number of bits from the pixel #1 (value from 0 to 16 bits). The values from 17 to 31 are invalid.

Table 11-744. Register Call Summary for RFBI_DATA_CYCLE1__1

RFBI Registers • RFBI Registers: [0] • RFBI_DATA_CYCLE1__1 Register (Offset = 84h) [reset = 0h]: [0]
DSS Functional Description • RFBI Configuration Selection: [0] • RFBI Cycle Mode Selection: [0]

11.3.5.6.22 RFBI_DATA_CYCLE2__1 Register (Offset = 88h) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_DATA_CYCLE2__1 is shown in Figure 11-316 and described in Table 11-746.

Return to Summary Table.

The control register configures the RFBI data format for 2nd cycle for the LCD#0 and LCD#1.

Table 11-745. RFBI_DATA_CYCLE2__1 Instances

Instance	Physical Address
RFBI	0254 6088h

Figure 11-316. RFBI_DATA_CYCLE2__1 Register

31	30	29	28	27	26	25	24
RESERVED				BITALIGNMENTPIXEL2			
R-0h				R/W-0h			
23	22	21	20	19	18	17	16
RESERVED				NBBITSPIXEL2			
R-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED				BITALIGNMENTPIXEL1			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
RESERVED				NBBITSPIXEL1			
R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-746. RFBI_DATA_CYCLE2__1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
27-24	BITALIGNMENTPIXEL2	R/W	0h	Bit alignment. Alignment of the bits from pixel#2 on the output interface.
23-21	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
20-16	NBBITSPIXEL2	R/W	0h	Number of bits. Number of bits from the pixel #2 (value from 0 to 16 bits). The values from 17 to 31 are invalid.
15-12	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
11-8	BITALIGNMENTPIXEL1	R/W	0h	Bit alignment. Alignment of the bits from pixel#1 on the output interface.
7-5	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
4-0	NBBITSPIXEL1	R/W	0h	Number of bits. Number of bits from the pixel #1 (value from 0 to 16 bits). The values from 17 to 31 are invalid.

Table 11-747. Register Call Summary for RFBI_DATA_CYCLE2__1

RFBI Registers • RFBI_DATA_CYCLE2__1 Register (Offset = 88h) [reset = 0h]: [0] • RFBI Registers: [0]
DSS Functional Description • RFBI Configuration Selection: [0] • RFBI Cycle Mode Selection: [0]

11.3.5.6.23 RFBI_DATA_CYCLE3__1 Register (Offset = 8Ch) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_DATA_CYCLE3__1 is shown in Figure 11-317 and described in Table 11-749.

Return to Summary Table.

The control register configures the RFBI data format for 3rd cycle for the LCD#0 and LCD#1.

Table 11-748. RFBI_DATA_CYCLE3__1 Instances

Instance	Physical Address
RFBI	0254 608Ch

Figure 11-317. RFBI_DATA_CYCLE3__1 Register

31	30	29	28	27	26	25	24
RESERVED				BITALIGNMENTPIXEL2			
R-0h				R/W-0h			
23	22	21	20	19	18	17	16
RESERVED				NBBITSPIXEL2			
R-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED				BITALIGNMENTPIXEL1			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
RESERVED				NBBITSPIXEL1			
R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-749. RFBI_DATA_CYCLE3__1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
27-24	BITALIGNMENTPIXEL2	R/W	0h	Bit alignment. Alignment of the bits from pixel#2 on the output interface.
23-21	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
20-16	NBBITSPIXEL2	R/W	0h	Number of bits. Number of bits from the pixel #2 (value from 0 to 16 bits). The values from 17 to 31 are invalid.
15-12	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
11-8	BITALIGNMENTPIXEL1	R/W	0h	Bit alignment. Alignment of the bits from pixel#1 on the output interface.
7-5	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
4-0	NBBITSPIXEL1	R/W	0h	Number of bits. Number of bits from the pixel #1 (value from 0 to 16 bits). The values from 17 to 31 are invalid.

Table 11-750. Register Call Summary for RFBI_DATA_CYCLE3__1

RFBI Registers • RFBI Registers: [0] • RFBI_DATA_CYCLE3__1 Register (Offset = 8Ch) [reset = 0h]: [0]
DSS Functional Description • RFBI Configuration Selection: [0] • RFBI Cycle Mode Selection: [0]

11.3.5.6.24 RFBI_VSYNC_WIDTH Register (Offset = 90h) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_VSYNC_WIDTH is shown in Figure 11-318 and described in Table 11-752.

Return to Summary Table.

The register configures the RFBI VSYNC minimum pulse width.

Table 11-751. RFBI_VSYNC_WIDTH Instances

Instance	Physical Address
RFBI	0254 6090h

Figure 11-318. RFBI_VSYNC_WIDTH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MINVSYNCPULSEWIDTH															
R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-752. RFBI_VSYNC_WIDTH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
15-0	MINVSYNCPULSEWIDTH	R/W	0h	Programmable min VSYNC pulse width. Minimum VSYNC pulse width from 0 to 65535. Number of OCP clock cycles to determine when VSYNC pulse occurs. The values 0 and 1 are invalid.

Table 11-753. Register Call Summary for RFBI_VSYNC_WIDTH

RFBI Registers • RFBI_VSYNC_WIDTH Register (Offset = 90h) [reset = 0h]: [0] • RFBI Registers: [0]
DSS Functional Description • RFBI External Trigger Mode: [0][1]
DSS Environment • RFBI Description of the Tearing Effect Pulse Signal: [0]

11.3.5.6.25 RFBI_HSYNC_WIDTH Register (Offset = 94h) [reset = 0h]

Register mask: FFFFFFFFh

RFBI_HSYNC_WIDTH is shown in Figure 11-319 and described in Table 11-755.

Return to Summary Table.

The register configures the RFBI HSYNC minimum pulse width.

Table 11-754. RFBI_HSYNC_WIDTH Instances

Instance	Physical Address
RFBI	0254 6094h

Figure 11-319. RFBI_HSYNC_WIDTH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MINHSYNCPULSEWIDTH															
R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-755. RFBI_HSYNC_WIDTH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Write 0's for future compatibility. Reads return 0.
15-0	MINHSYNCPULSEWIDTH	R/W	0h	Programmable min HSYNC pulse width. Minimum HSYNC pulse width from 0 to 65535. Number of OCP clock cycles to determine when HSYNC pulse occurs.The values 0 and 1 are invalid.

Table 11-756. Register Call Summary for RFBI_HSYNC_WIDTH

RFBI Registers • RFBI Registers: [0] • RFBI_HSYNC_WIDTH Register (Offset = 94h) [reset = 0h]: [0]
DSS Functional Description • RFBI External Trigger Mode: [0][1]
DSS Environment • RFBI Description of the Tearing Effect Pulse Signal: [0]

11.4 Enhanced Capture (eCAP) Module

This section describes the Enhanced Capture (eCAP) module for the device.

11.4.1 eCAP Overview

11.4.1.1 Purpose of the eCAP Peripheral

The enhanced Capture (eCAP) module can be used for:

- Sample rate measurements of audio inputs
- Speed measurements of rotating machinery (for example, toothed sprockets sensed via Hall sensors)
- Elapsed time measurements between position sensor pulses
- Period and duty cycle measurements of pulse train signals
- Decoding current or voltage amplitude derived from duty cycle encoded current/voltage sensors.

11.4.1.2 eCAP Features

The eCAP module includes the following features:

- 32-bit time base counter
- 4-event time-stamp registers (each 32 bits)
- Edge polarity selection for up to four sequenced time-stamp capture events
- Interrupt on either of the four events
- Single shot capture of up to four event time-stamps
- Continuous mode capture of time-stamps in a four-deep circular buffer
- Absolute time-stamp capture
- Difference (Delta) mode time-stamp capture
- All above resources dedicated to a single input pin
- When not used in capture mode, the eCAP module can be configured as a single channel PWM output.

11.4.2 eCAP Environment

11.4.2.1 eCAP I/O Interface

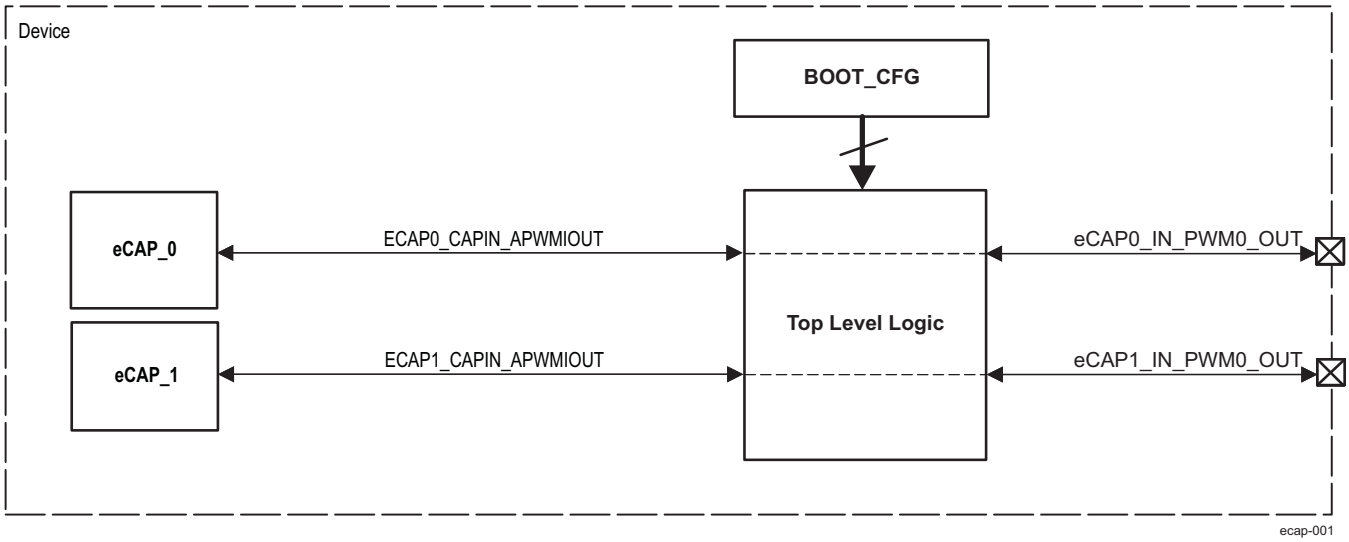
Table 11-757 shows the device integrated eCAP subsystems interface signals to external devices.

Table 11-757. eCAP Subsystems I/O Signals

eCAP Modules Level Signal Name	Device Level Signal Name	I/O Type[1]	Description	Module Pin Reset Value
eCAP_0				
ECAP0_CAPIN_APWMOUT	eCAP0_IN_PWM0_OUT	I/O	eCAP_0 Capture input / PWM0 output	HiZ
eCAP_1				
ECAP1_CAPIN_APWMOUT	eCAP1_IN_PWM1_OUT	I/O	eCAP_1 Capture input / PWM1 output	HiZ

(1) I = Input; O = Output

Figure 11-320. eCAP External Interface I/Os



11.4.3 eCAP Integration

This section describes module integration in the device, including information about clocks, resets, and hardware requests.

Figure 11-437 shows the eCAP integration.

There are two instances of the Enhanced Capture (eCAP) module integrated in the device.

Figure 11-321. eCAP Integration

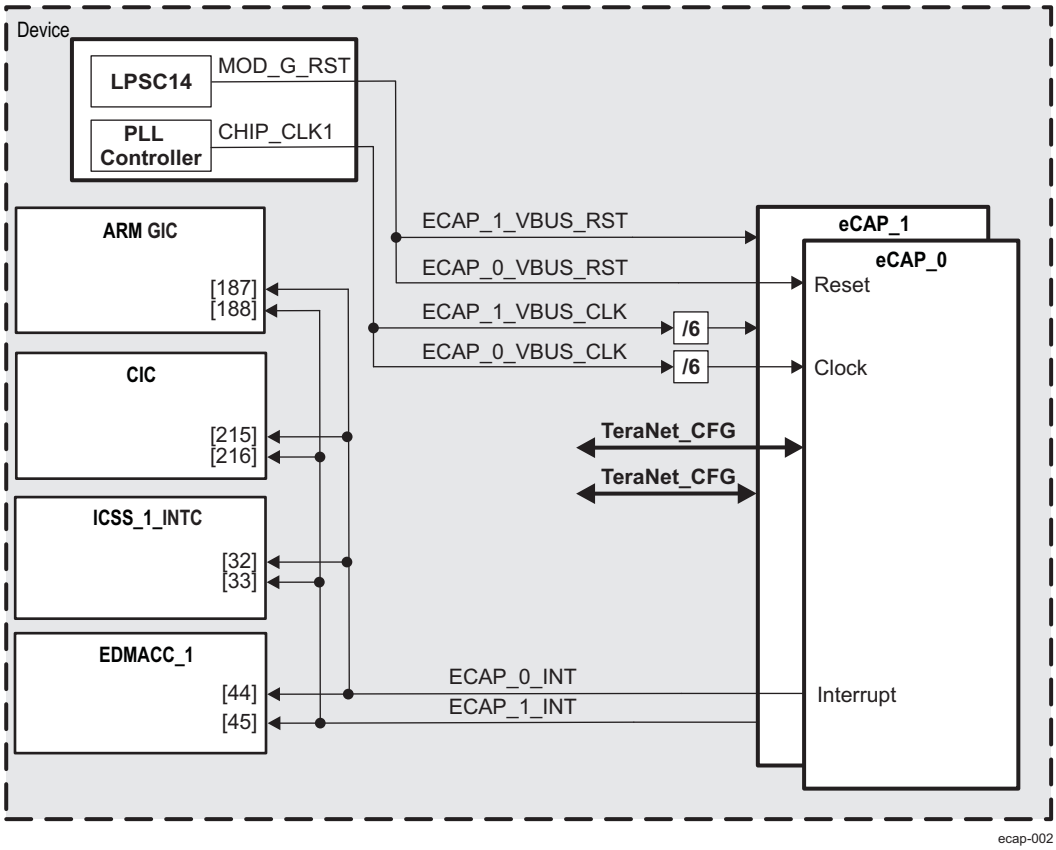


Table 11-924 through Table 11-926 summarize the integration of the module in the device.

Table 11-758. eCAP Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
eCAP_0	PD5	LPSC14	TeraNet_CFG
eCAP_1	PD5	LPSC14	TeraNet_CFG

Table 11-759. eCAP Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
eCAP_0	ECAP_0_VBUS_CLK	CHIP_CLK1 / 6	PLL_CONTROLLER	ECAP_0 functional and interface clock
eCAP_1	ECAP_1_VBUS_CLK	CHIP_CLK1 / 6	PLL_CONTROLLER	ECAP_1 functional and interface clock
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
eCAP_0	ECAP_0_VBUS_RST	MOD_G_RST	LPSC14	Module Reset
eCAP_1	ECAP_1_VBUS_RST	MOD_G_RST	LPSC14	Module Reset

Table 11-760. eCAP Hardware Requests

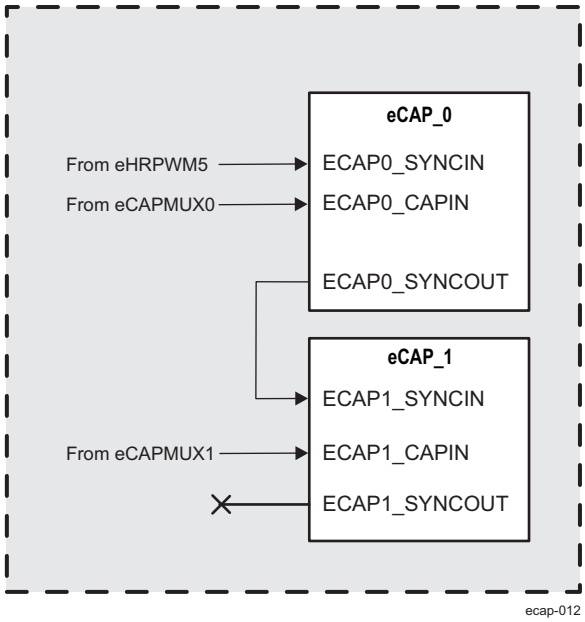
Interrupt Requests					
Module Instance	**Event Name**	**Mapped To Input Event [Number]**			**Description**
		ARM_GIC	**CIC**	**ICSS_1_INTC**	
eCAP_0	ECAP_0_INT	[187]	[215]	[32]	eCAP_0 interrupt
eCAP_1	ECAP_1_INT	[188]	[216]	[33]	eCAP_1 interrupt
DMA Requests					
Module Instance	**Event Name**	**Mapped To Input Event [Number]**			**Description**
		EDMACC_0		**EDMACC_1**	
eCAP_0	ECAP_0_INT	–		[44]	ECAP_0 event to the EDMA_1
eCAP_1	ECAP_1_INT	–		[45]	ECAP_1 event to the EDMA_1

NOTE: For more information about interrupts, see Section 11.4.4.2.1.6, *eCAP Interrupt Control*.

11.4.3.1 Daisy-Chain Connectivity between eCAP Modules

The eCAP module is provided with input and output synchronisation signals allowing synchronisation with other modules or events. In the device, these signals are connected in a daisy-chain fashion as shown in Figure 11-322.

Figure 11-322. eCAP Daisy-Chain Connectivity



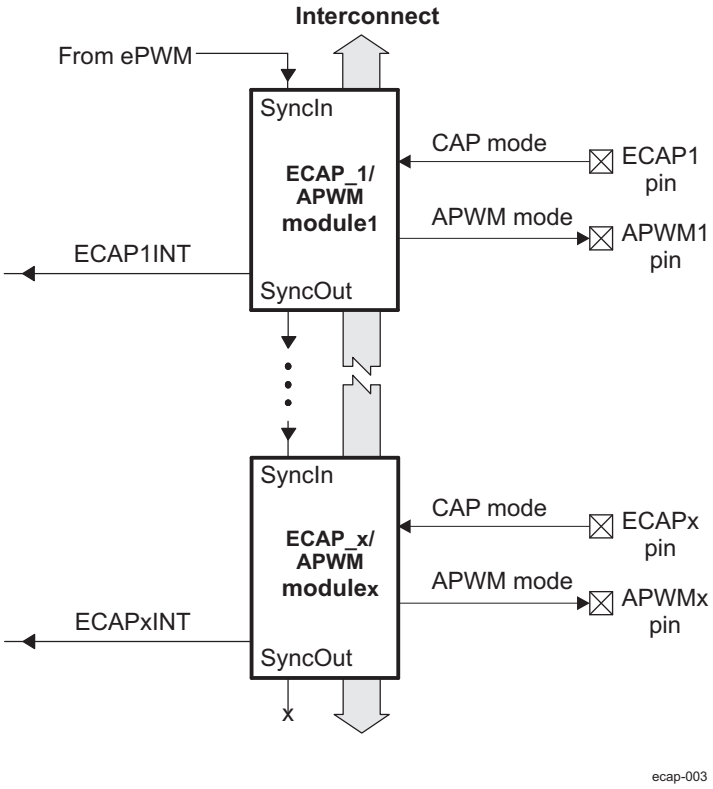
11.4.4 eCAP Functional Description

The eCAP module represents one complete capture channel that can be instantiated multiple times depending on the target device. In the context of this guide, one eCAP channel has the following independent key resources:

- Dedicated input capture pin
- 32 events selection mapping capability to the input capture pin
- 32-bit time base counter
- 4 × 32-bit time-stamp capture registers (PWMSS_ECAP_CAP1 through PWMSS_ECAP_CAP4)
- 4-stage sequencer (Mod4 counter) that is synchronized to external events, ECAP pin rising/falling edges.
- Independent edge polarity (rising/falling edge) selection for all 4 events
- Input capture signal prescaling (from 2-62)
- One-shot compare register (2 bits) to freeze captures after 1 to 4 time-stamp events
- Control for continuous time-stamp captures using a 4-deep circular buffer (PWMSS_ECAP_CAP1 through PWMSS_ECAP_CAP4) scheme
- Interrupt capabilities on any of the 4 capture events.

Multiple identical eCAP modules can be contained in a system as shown in Figure 11-323. For actual number of eCAP modules integrated in the device, refer to the Section 11.6.3, *eCAP Integration*. The letter x within a signal or module name is used to indicate a generic eCAP instance on a device. For example, output interrupt request, ECAP1INT belongs to eCAP_1, ECAP2INT belongs to eCAP_2, and so forth.

Figure 11-323. Multiple eCAP Modules



11.4.4.1 eCAP Input Capture Events

The eCAP input capture events can be selected from 2 external eCAP input pins or 30 different internal events as selected through the eCAPMUX0 and eCAPMUX1 event muxes, using the ECAP_CAPEVT_CTL register in the BOOT_CFG module. Figure 11-324 shows the eCAP input events mapping.

Figure 11-324. eCAP Input Events

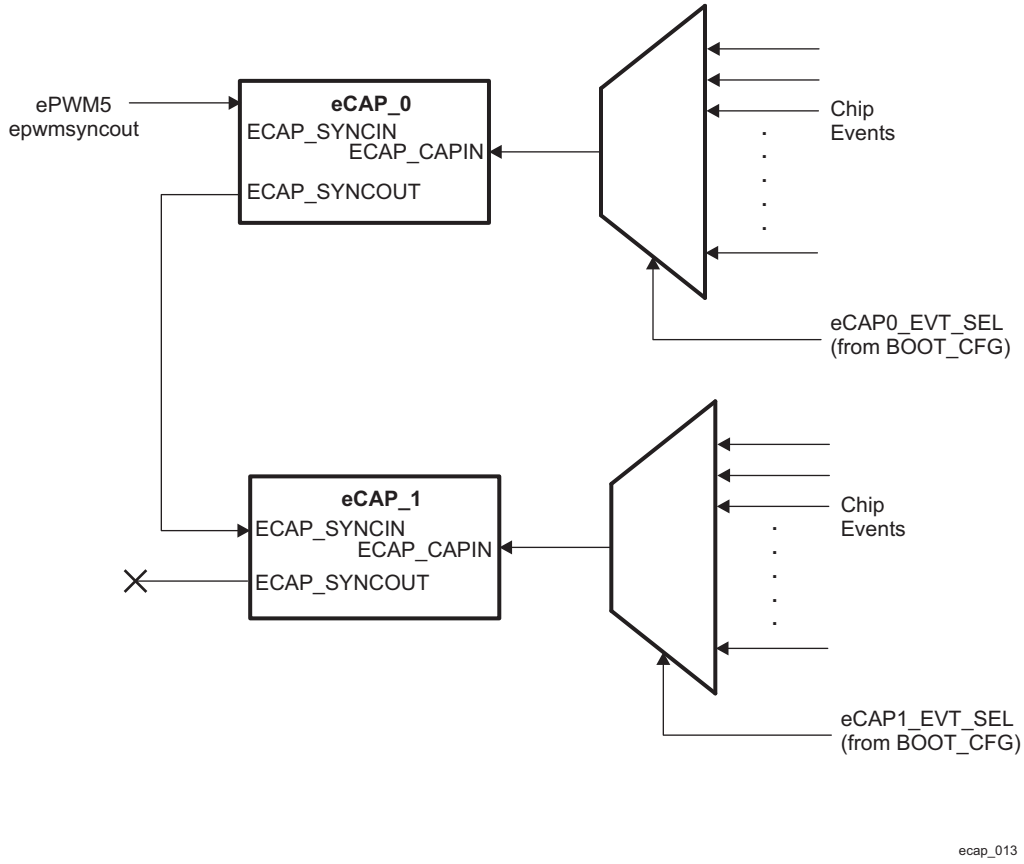


Figure 11-352 summarizes the input event mapping.

Table 11-761. eCAPMUX0 and eCAPMUX1 Input Event Mapping

Input event	Event Name	Description	Source
0	eCAP0_IN_PWM0_OUT	ECAP_0 input event from external pin	ePWM_0
1	eCAP1_IN_PWM1_OUT	ECAP_1 input event from external pin	ePWM_1
2	McASP_0_XEVT	MCASP_0 Transmit event	MCASP_0
3	McASP_0_REVT	MCASP_0 Receive event	MCASP_0
4	McASP_1_XEVT	MCASP_1 Transmit event	MCASP_1
5	McASP_1_REVT	MCASP_1 Receive event	MCASP_1
6	McASP_2_XEVT	MCASP_2 Transmit event	MCASP_2
7	McASP_2_REVT	MCASP_2 Receive event	MCASP_2
8	DCAN_0_INTERFACE1_EVT	DCAN_0 Interface 1 event	DCAN_0
9	DCAN_0_INTERFACE2_EVT	DCAN_0 Interface 2 event	DCAN_0
10	DCAN_0_INTERFACE3_EVT	DCAN_0 Interface 3 event	DCAN_0
11	DCAN_1_INTERFACE1_EVT	DCAN_1 Interface 1 event	DCAN_1
12	DCAN_1_INTERFACE2_EVT	DCAN_1 Interface 2 event	DCAN_1

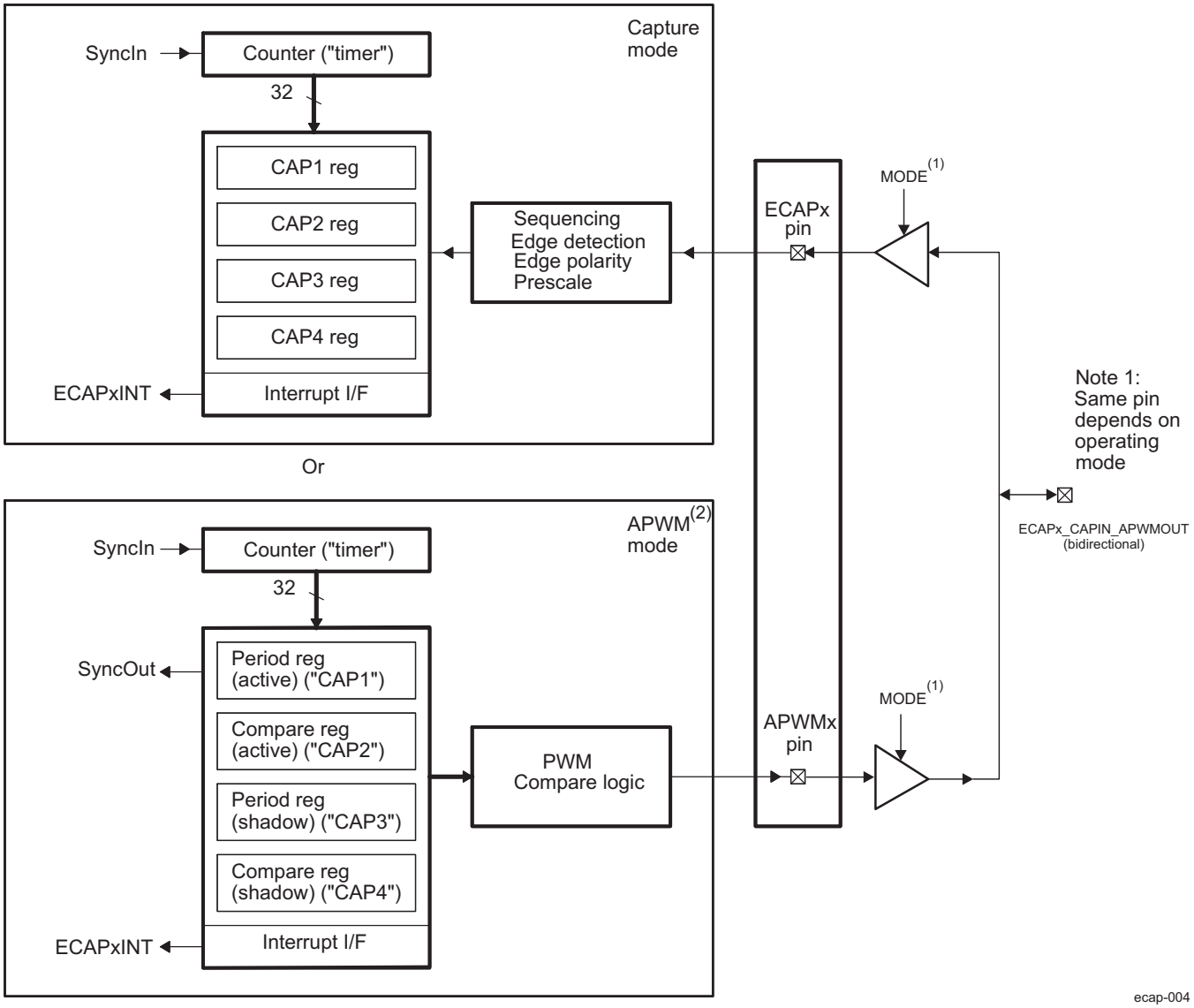
Table 11-761. eCAPMUX0 and eCAPMUX1 Input Event Mapping (continued)

13	DCAN_1_INTERFACE3_EVT	DCAN_1 Interface 3 event	DCAN_1
14	McBSP_XEVT	MCBSP Transmit event	MCBSP
15	McBSP_REVT	MCBSP Receive event	MCBSP
16	RESERVED	-	-
17	RESERVED	-	-
18	RESERVED	-	-
19	RESERVED	-	-
20	RESERVED	-	-
21	RESERVED	-	-
22	RESERVED	-	-
23	RESERVED	-	-
24	RESERVED	-	-
25	RESERVED	-	-
26	RESERVED	-	-
27	RESERVED	-	-
28	RESERVED	-	-
29	RESERVED	-	-
30	GPIOMUX_INT54	GPIO0 pin 54 interrupt	GPIOMUX
31	GPIOMUX_INT55	GPIO0 pin 55 interrupt	GPIOMUX

11.4.4.2 Capture and APWM Operating Modes

The eCAP module resources can be used to implement a single-channel PWM generator (with 32 bit capabilities) when it is not being used for input captures. The counter operates in count-up mode, providing a time-base for asymmetrical pulse width modulation (PWM) waveforms. The PWMSS_ECAP_CAP1 and PWMSS_ECAP_CAP2 registers become the active period and compare registers, respectively, while PWMSS_ECAP_CAP3 and PWMSS_ECAP_CAP4 registers become the period and capture shadow registers, respectively. is a high-level view of both the capture and auxiliary pulse-width modulator (APWM) modes of operation.

Figure 11-325. Capture and APWM Modes of Operation



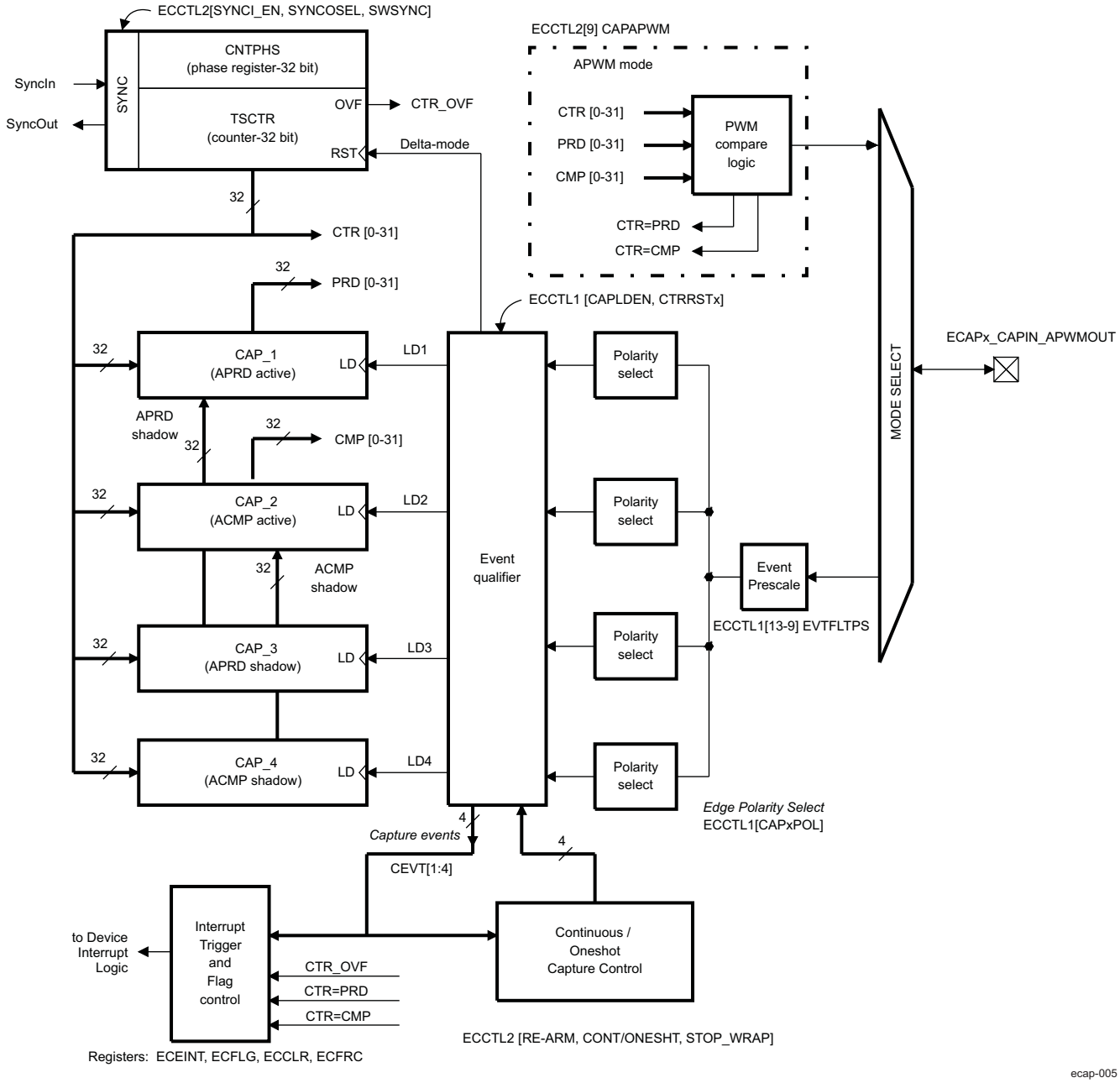
(1) A single pin is shared between CAP and APWM functions. In capture mode, it is an input; in APWM mode, it is an output.

(2) In APWM mode, writing any value to PWMSS_ECAP_CAP1/PWMSS_ECAP_CAP2 active registers also writes the same value to the corresponding shadow registers PWMSS_ECAP_CAP3/PWMSS_ECAP_CAP4. This emulates immediate mode. Writing to the shadow registers PWMSS_ECAP_CAP3/PWMSS_ECAP_CAP4 invokes the shadow mode.

11.4.4.2.1 eCAP Capture Mode Description

Figure 11-326 shows the various components that implement the capture function.

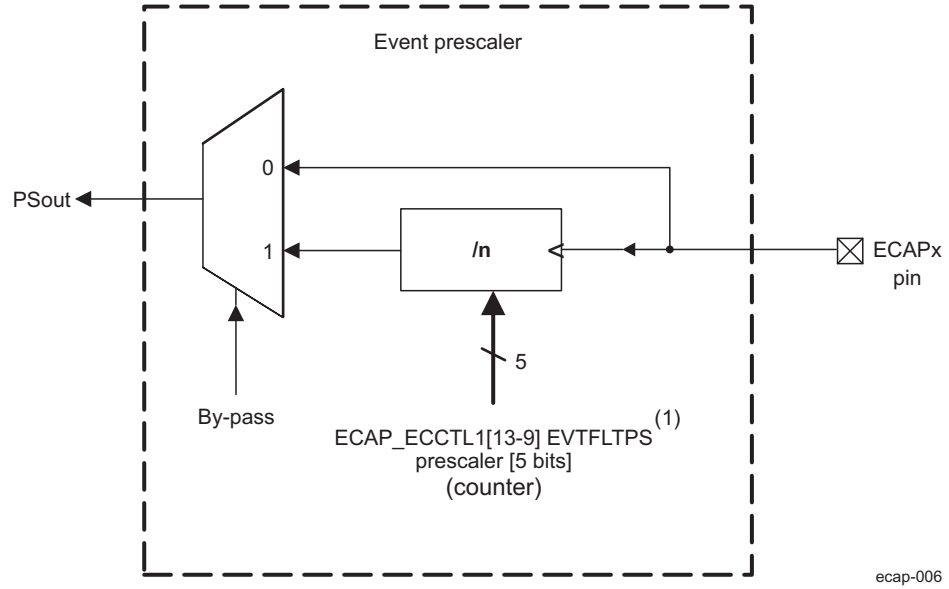
Figure 11-326. Capture Function Diagram



11.4.4.2.1.1 eCAP Event Prescaler

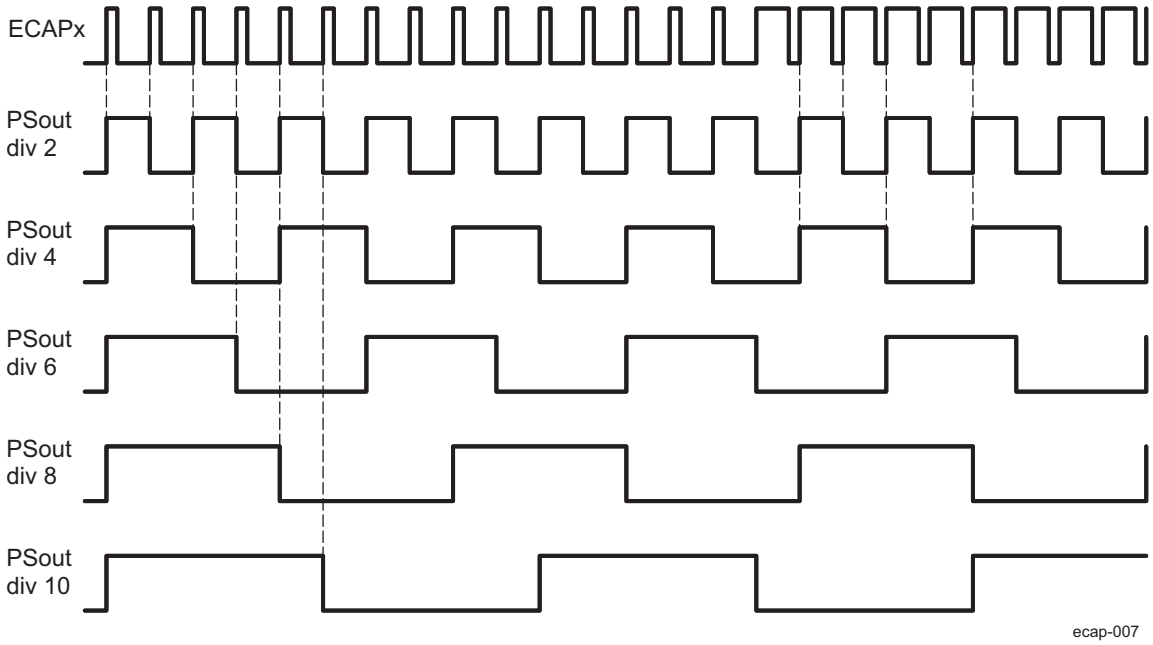
An input capture signal (pulse train) can be prescaled by N = 2-62 (in multiples of 2) or can bypass the prescaler. This is useful when very high frequency signals are used as inputs. Figure 11-327 shows a functional diagram and Figure 11-328 shows the operation of the prescale function.

Figure 11-327. Event Prescale Control



(1) When a prescale value of 1 is chosen (PWMSS_ECAP_ECCTL1[13-9] EVTFLTPS = 0b0000) the input capture signal by-passes the prescale logic completely.

Figure 11-328. Prescale Function Waveforms



11.4.4.2.1.2 eCAP Edge Polarity Select and Qualifier

- Four independent edge polarity (rising edge/falling edge) selection multiplexers are used, one for each capture event.
- Each edge (up to 4) is event qualified by the Mod4 sequencer.
- The edge event is gated to its respective CAP*n* register by the Mod4 counter. The CAP*n* register is loaded on the falling edge.

11.4.4.2.1.3 eCAP Continuous/One-Shot Control

- The Mod4 (2 bit) counter is incremented via edge qualified events CEVT1 through CEVT4 (see register PWMSS_ECAP_ECEINT [4-1] field).
- The Mod4 counter continues counting (0->1->2->3->0) and wraps around unless stopped.
- A 2-bit stop register is used to compare the Mod4 counter output; when equal the Mod4 counter stops and inhibits further loads of the PWMSS_ECAP_CAP1 through PWMSS_ECAP_CAP4 registers. This occurs during one-shot operation.

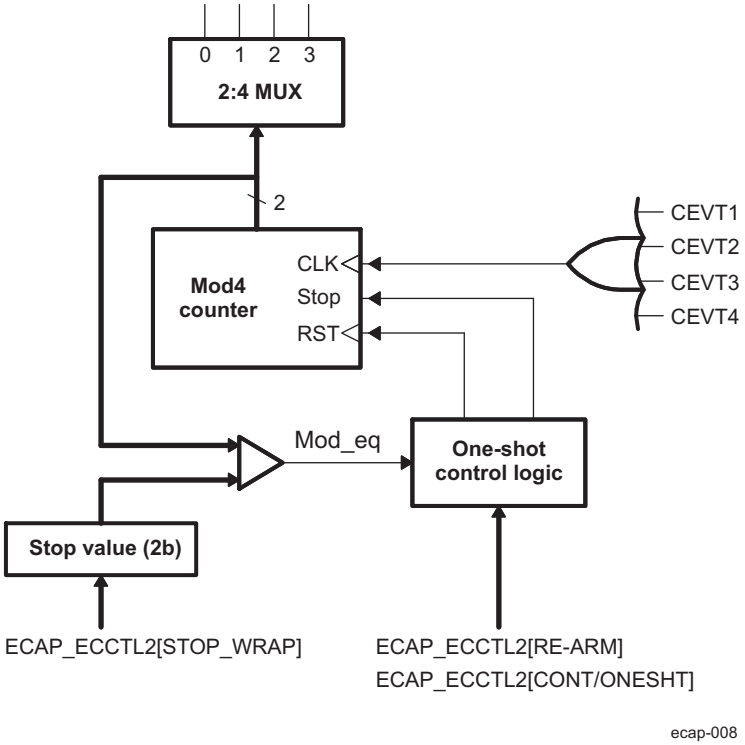
The continuous/one-shot block (Figure 11-329) controls the start/stop and reset (zero) functions of the Mod4 counter via a mono-shot type of action that can be triggered by the stop-value comparator and re-armed via software control.

Once armed, the eCAP module waits for 1-4 (defined by stop-value) capture events before freezing both the Mod4 counter and contents of PWMSS_ECAP_CAP1 through PWMSS_ECAP_CAP4 registers (time-stamps).

Re-arming prepares the eCAP module for another capture sequence. Also re-arming clears (to zero) the Mod4 counter and permits loading of PWMSS_ECAP_CAP1 through PWMSS_ECAP_CAP4 registers again, providing the CAPLDEN bit is set.

In continuous mode, the Mod4 counter continues to run (0->1->2->3->0), the one-shot action is ignored, and capture values continue to be written to PWMSS_ECAP_CAP1 through PWMSS_ECAP_CAP4 registers in a circular buffer sequence.

Figure 11-329. eCAP Continuous/One-shot Block Diagram



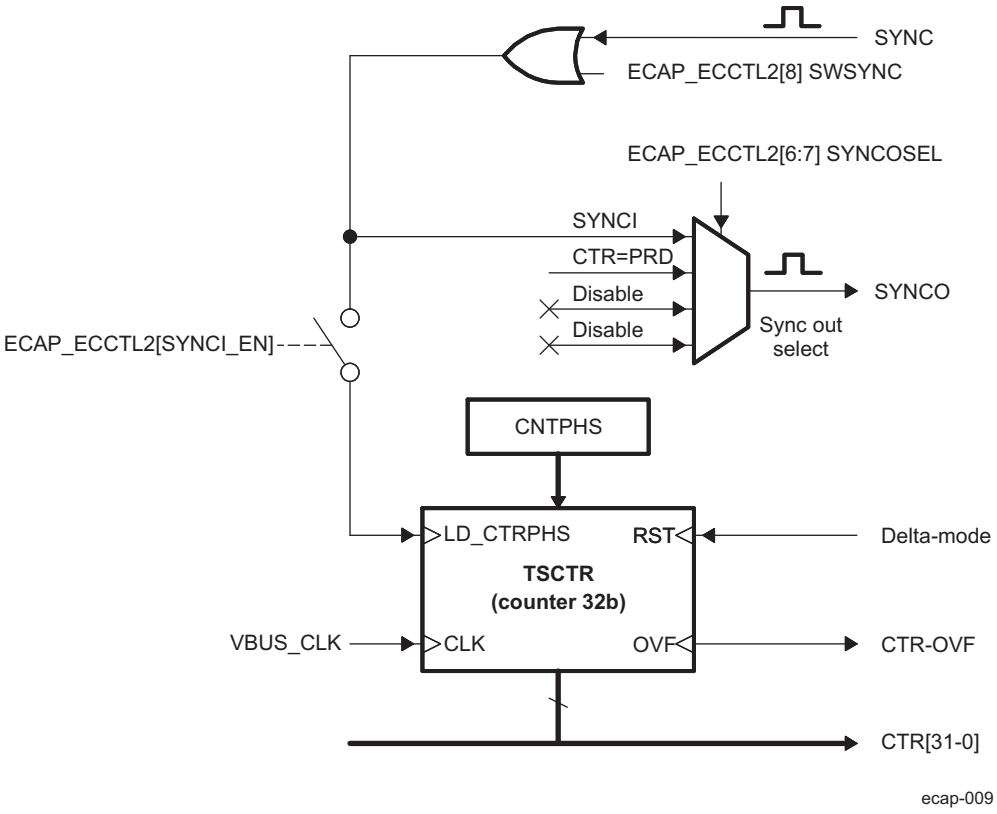
11.4.4.2.1.4 eCAP 32-Bit Counter and Phase Control

This counter (Figure 11-330) provides the time-base for event captures, and is clocked via the system clock.

A phase register is provided to achieve synchronization with other counters, via a hardware and software forced sync. This is useful in APWM mode when a phase offset between modules is needed.

On any of the four event loads, an option to reset the 32-bit counter is given. This is useful for time difference capture. The 32-bit counter value is captured first, then it is reset to 0 by any of the LD1 through LD4 signals.

Figure 11-330. eCAP Counter and Synchronization Block Diagram



11.4.4.2.1.5 CAP1-CAP4 Registers

These 32-bit registers are fed by the 32-bit counter timer bus, CTR[0-31] and are loaded (capture a time-stamp) when their respective LD inputs are strobed.

Loading of the capture registers can be inhibited via control bit PWMSS_ECAP_ECCTL1[8] CAPLDEN. During one-shot operation, this bit is cleared (loading is inhibited) automatically when a stop condition occurs, StopValue = Mod4.

PWMSS_ECAP_CAP1 and PWMSS_ECAP_CAP2 registers become the active period and compare registers, respectively, in APWM mode.

PWMSS_ECAP_CAP3 and PWMSS_ECAP_CAP4 registers become the respective shadow registers (APRD and ACMP) for PWMSS_ECAP_CAP1 and PWMSS_ECAP_CAP2 during APWM operation.

11.4.4.2.1.6 eCAP Interrupt Control

An Interrupt can be generated on capture events CEVT1 through CEVT4, CNTOVF (see PWMSS_ECAP_ECEINT[5] CNTOVF bit) or APWM events (TSCNT = PRD, TSCNT = CMP). See Figure 11-331.

A counter overflow event (FFFF FFFFh->0000 0000h) is also provided as an interrupt source (CNTOVF).

The capture events are edge and sequencer qualified (that is, ordered in time) by the polarity select and Mod4 gating, respectively.

One of these events can be selected as the interrupt source (from the eCAP*n* module) going to the interrupt controller.

Seven interrupt events (CEVT1, CEVT2, CEVT3, CEVT4, CNTOVF, TSCNT = PRD, TSCNT = CMP) can be generated. The interrupt enable register (PWMSS_ECAP_ECEINT) is used to enable/disable individual interrupt event sources. The interrupt flag register (PWMSS_ECAP_ECFLG) indicates if any interrupt event has been latched and contains the global interrupt flag bit PWMSS_ECAP_ECFLG[0] INT. An interrupt pulse is generated to the interrupt controller only if any of the interrupt events are enabled, the flag bit is 1, and the INT flag bit is 0. The interrupt service routine must clear the global interrupt flag bit and the serviced event via the interrupt clear register (PWMSS_ECAP_ECCLR) before any other interrupt pulses are generated. The interrupt force register (PWMSS_ECAP_ECFRC) can force an interrupt event. This is useful for test purposes.

11.4.4.2.1.7 eCAP Shadow Load and Lockout Control

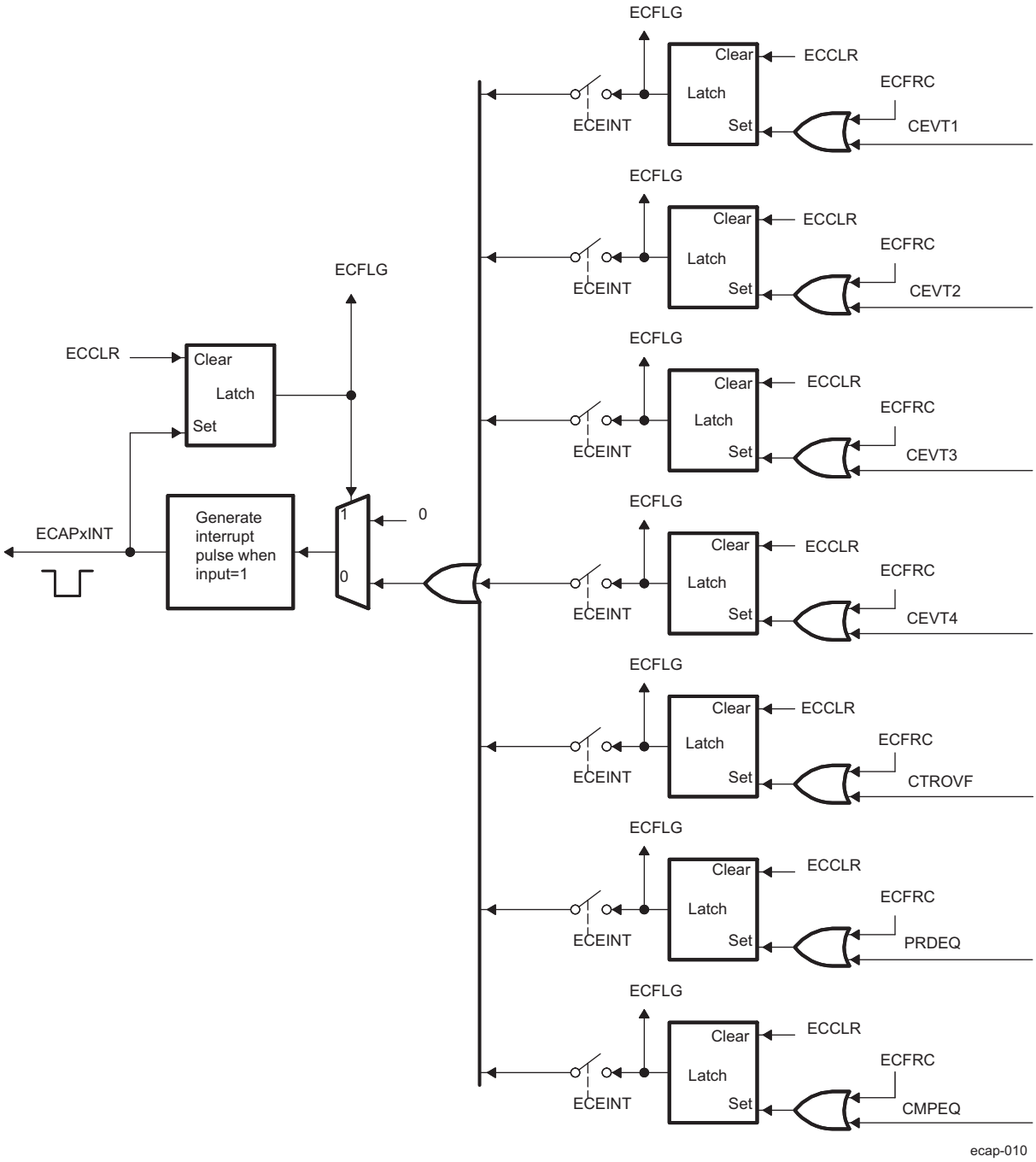
In capture mode, this logic inhibits (locks out) any shadow loading of PWMSS_ECAP_CAP1 or PWMSS_ECAP_CAP2 from APRD and ACMP registers, respectively.

In APWM mode, shadow loading is active and two choices are permitted:

- Immediate - APRD or ACMP are transferred to PWMSS_ECAP_CAP1 or PWMSS_ECAP_CAP2 immediately upon writing a new value.
- On period equal, CTR[31-0] = PRD[31-0]

NOTE: The CEVT1, CEVT2, CEVT3, CEVT4 flags are only active in capture mode (PWMSS_ECAP_ECCTL2[9] CAPAPWM == 0). The TSCNT = PRD, TSCNT = CMP flags are only valid in APWM mode (PWMSS_ECAP_ECCTL2[9] CAPAPWM == 1). CNTOVF flag is valid in both modes.

Figure 11-331. Interrupts in eCAP Module

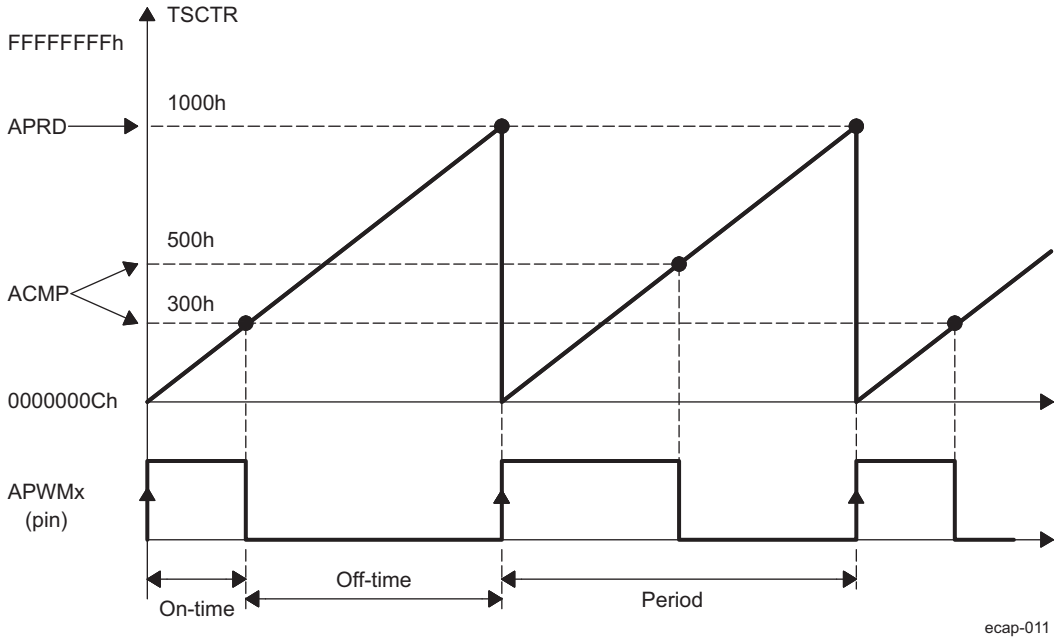


11.4.4.2.2 eCAP APWM Mode Operation

Main operating highlights of the APWM section:

- The time-stamp counter bus is made available for comparison via 2 digital (32-bit) comparators.
- When PWMSS_ECAP_CAP1/2 registers are not used in capture mode, their contents can be used as Period and Compare values in APWM mode.
- Double buffering is achieved via shadow registers APRD and ACMP (PWMSS_ECAP_CAP3/4). The shadow register contents are transferred over to PWMSS_ECAP_CAP1/2 registers either immediately upon a write, or on a TSCNT = PRD trigger.
- In APWM mode, writing to PWMSS_ECAP_CAP1/PWMSS_ECAP_CAP2 active registers will also write the same value to the corresponding shadow registers PWMSS_ECAP_CAP3/PWMSS_ECAP_CAP4. This emulates immediate mode. Writing to the shadow registers PWMSS_ECAP_CAP3/PWMSS_ECAP_CAP4 will invoke the shadow mode.
- During initialization, SW must write the PRD and CMP values to the active registers. This automatically copies the initial values into the shadow values. For subsequent compare updates, during run-time, SW should use only shadow registers.

Figure 11-332. PWM Waveform Details Of eCAP APWM Mode Operation



The behavior of APWM active-high mode (APWMPOL == 0) is:

```
CMP = 0x00000000, output low for duration of period (0% duty)
CMP = 0x00000001, output high 1 cycle
CMP = 0x00000002, output high 2 cycles
CMP = PERIOD, output high except for 1 cycle (<100% duty)
CMP = PERIOD+1, output high for complete period (100% duty)
CMP > PERIOD+1, output high for complete period
```

The behavior of APWM active-low mode (APWMPOL == 1) is:

```
CMP = 0x00000000, output high for duration of period (0% duty)
CMP = 0x00000001, output low 1 cycle
CMP = 0x00000002, output low 2 cycles
CMP = PERIOD, output low except for 1 cycle (<100% duty)
```

```
CMP = PERIOD+1, output low for complete period (100% duty)
CMP > PERIOD+1, output low for complete period
```

11.4.4.3 Summary of eCAP Functional Registers

Table 11-762 shows the eCAP module control and status register set. All 32-bit registers are aligned on even address boundaries and are organized in little-endian mode. The 16 least-significant bits of a 32-bit register are located on lowest address (even address).

NOTE: In APWM mode, writing to PWMSS_ECAP_CAP1/PWMSS_ECAP_CAP2 active registers also writes the same value to the corresponding shadow registers PWMSS_ECAP_CAP3/PWMSS_ECAP_CAP4. This emulates immediate mode. Writing to the shadow registers PWMSS_ECAP_CAP3/PWMSS_ECAP_CAP4 invokes the shadow mode.

Table 11-762. eCAP Control and Status Functional Registers

Offset	Register Name	Description	Size (×16)
0h	PWMSS_ECAP_TSCNT	Time-Stamp Counter Register	2
4h	PWMSS_ECAP_CNTPHS	Counter Phase Offset Value Register	2
8h	PWMSS_ECAP_CAP1	Capture 1 Register	2
Ch	PWMSS_ECAP_CAP2	Capture 2 Register	2
10h	PWMSS_ECAP_CAP3	Capture 3 Register	2
14h	PWMSS_ECAP_CAP4	Capture 4 Register	2
28h	PWMSS_ECAP_ECCTL1	Capture Control Register 1	1
2Ah	PWMSS_ECAP_ECCTL2	Capture Control Register 2	1
2Ch	PWMSS_ECAP_ECEINT	Capture Interrupt Enable Register	1
2Eh	PWMSS_ECAP_ECFLG	Capture Interrupt Flag Register	1
30h	PWMSS_ECAP_ECCLR	Capture Interrupt Clear Register	1
32h	PWMSS_ECAP_ECFRC	Capture Interrupt Force Register	1
5Ch	PWMSS_ECAP_PID	Revision ID Register	2

11.4.5 eCAP Registers

Table 11-764 lists the memory-mapped registers for the eCAP. All register offset addresses not listed in Table 11-764 should be considered as reserved locations and the register contents should not be modified.

This section describes the eCAP instances registers.

Table 11-763. eCAP Instances

Instance	Module Base Address
ECAP_0	021D 1800h
ECAP_1	021D 1C00h

Table 11-764. eCAP Registers

Offset	Acronym	Register Name	ECAP_0 Physical Address	ECAP_1 Physical Address	Section
0h	PWMSS_ECAP_TSCNT	Time Stamp Counter Register	021D 1800h	021D 1C00h	Section 11.4.5.1
4h	PWMSS_ECAP_CNTPHS	Counter Phase Control Register	021D 1804h	021D 1C04h	Section 11.4.5.2
8h	PWMSS_ECAP_CAP1	Capture-1 Register	021D 1808h	021D 1C08h	Section 11.4.5.3
Ch	PWMSS_ECAP_CAP2	Capture-2 Register	021D 180Ch	021D 1C0Ch	Section 11.4.5.4
10h	PWMSS_ECAP_CAP3	Capture-3 Register	021D 1810h	021D 1C10h	Section 11.4.5.5
14h	PWMSS_ECAP_CAP4	Capture-4 Register	021D 1814h	021D 1C14h	Section 11.4.5.6
28h	PWMSS_ECAP_ECCTL1	eCAP Control Register 1	021D 1828h	021D 1C28h	Section 11.4.5.7
2Ah	PWMSS_ECAP_ECCTL2	eCAP Control Register 2	021D 182Ah	021D 1C2Ah	Section 11.4.5.8
2Ch	PWMSS_ECAP_ECEINT	eCAP Interrupt Enable Register	021D 182Ch	021D 1C2Ch	Section 11.4.5.9
2Eh	PWMSS_ECAP_ECFLG	eCAP Interrupt Flag Register	021D 182Eh	021D 1C2Eh	Section 11.4.5.10
30h	PWMSS_ECAP_ECCLR	eCAP Interrupt Clear Register	021D 1830h	021D 1C30h	Section 11.4.5.11
32h	PWMSS_ECAP_ECFRC	eCAP Interrupt Forcing Register	021D 1832h	021D 1C32h	Section 11.4.5.12
5Ch	PWMSS_ECAP_PID	eCAP Revision ID	021D 185Ch	021D 1C5Ch	Section 11.4.5.13

11.4.5.1 PWMSS_ECAP_TSCNT Register (Offset = 0h) [reset = 0h]

PWMSS_ECAP_TSCNT is shown in Figure 11-333 and described in Table 11-766.

Time Stamp Counter Register.

Table 11-765. PWMSS_ECAP_TSCNT Instances

Instance	Physical Address
ECAP_0	021D 1800h
ECAP_1	021D 1C00h

Figure 11-333. PWMSS_ECAP_TSCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSCNT																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-766. PWMSS_ECAP_TSCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TSCNT	R/W	0h	Active 32 bit-counter register that is used as the capture time-base

11.4.5.2 PWMSS_ECAP_CNTPHS Register (Offset = 4h) [reset = 0h]

PWMSS_ECAP_CNTPHS is shown in Figure 11-334 and described in Table 11-768.

Counter Phase Control Register.

Table 11-767. PWMSS_ECAP_CNTPHS Instances

Instance	Physical Address
ECAP_0	021D 1804h
ECAP_1	021D 1C04h

Figure 11-334. PWMSS_ECAP_CNTPHS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNTPHS																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-768. PWMSS_ECAP_CNTPHS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CNTPHS	R/W	0h	Counter phase value register that can be programmed for phase lag/lead. This register shadows TSCNT and is loaded into PWMSS_ECAP_TSCNT upon either a SYNCI event or S/W force via a control bit. Used to achieve phase control synchronization with respect to other eCAP and EPWM time-bases.

11.4.5.3 PWMSS_ECAP_CAP1 Register (Offset = 8h) [reset = 0h]

PWMSS_ECAP_CAP1 is shown in Figure 11-335 and described in Table 11-770.

Capture-1 Register.

Table 11-769. PWMSS_ECAP_CAP1 Instances

Instance	Physical Address
ECAP_0	021D 1808h
ECAP_1	021D 1C08h

Figure 11-335. PWMSS_ECAP_CAP1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP1																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-770. PWMSS_ECAP_CAP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP1	R/W	0h	This register can be loaded (written) by the following. 1. Time-Stamp (that is, counter value) during a capture event. 2. Software may be useful for test purposes. 3. APRD active register when used in APWM mode.

11.4.5.4 PWMSS_ECAP_CAP2 Register (Offset = Ch) [reset = 0h]

PWMSS_ECAP_CAP2 is shown in Figure 11-336 and described in Table 11-772.

Capture-2 Register.

Table 11-771. PWMSS_ECAP_CAP2 Instances

Instance	Physical Address
ECAP_0	021D 180Ch
ECAP_1	021D 1C0Ch

Figure 11-336. PWMSS_ECAP_CAP2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP2																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-772. PWMSS_ECAP_CAP2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP2	R/W	0h	This register can be loaded (written) by the following. 1. Time- Stamp (that is, counter value) during a capture event. 2. Software may be useful for test purposes. 3. APRD active register when used in APWM mode.

11.4.5.5 PWMSS_ECAP_CAP3 Register (Offset = 10h) [reset = 0h]

PWMSS_ECAP_CAP3 is shown in Figure 11-337 and described in Table 11-774.

Capture-3 Register.

Table 11-773. PWMSS_ECAP_CAP3 Instances

Instance	Physical Address
ECAP_0	021D 1810h
ECAP_1	021D 1C10h

Figure 11-337. PWMSS_ECAP_CAP3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP3																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-774. PWMSS_ECAP_CAP3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP3	R/W	0h	In CMP mode, this is a time-stamp capture register. In APWM mode, this is the period shadow (APRD) register. User SW updates the PWM period value through this register. In this mode, CAP3 shadows CAP1.

11.4.5.6 PWMSS_ECAP_CAP4 Register (Offset = 14h) [reset = 0h]

PWMSS_ECAP_CAP4 is shown in Figure 11-338 and described in Table 11-776.

Capture-4 Register.

Table 11-775. PWMSS_ECAP_CAP4 Instances

Instance	Physical Address
ECAP_0	021D 1814h
ECAP_1	021D 1C14h

Figure 11-338. PWMSS_ECAP_CAP4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP4																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-776. PWMSS_ECAP_CAP4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP4	R/W	0h	In CMP mode, this is a time-stamp capture register. In APWM mode, this is the compare shadow (ACMP) register. User SW updates the PWM compare value through this register. In this mode, CAP4 shadows CAP2.

11.4.5.7 PWMSS_ECAP_ECCTL1 Register (Offset = 28h) [reset = 0h]

PWMSS_ECAP_ECCTL1 is shown in Figure 11-339 and described in Table 11-778.

eCAP Control Register 1.

Table 11-777. PWMSS_ECAP_ECCTL1 Instances

Instance	Physical Address
ECAP_0	021D 1828h
ECAP_1	021D 1C28h

Figure 11-339. PWMSS_ECAP_ECCTL1 Register

15	14	13	12	11	10	9	8
FREE_SOFT		EVTFLTPS					CAPLDEN
R/W-0h		R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
CTRRST4	CAP4POL	CTRRST3	CAP3POL	CTRRST2	CAP2POL	CTRRST1	CAP1POL
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-778. PWMSS_ECAP_ECCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	FREE_SOFT	R/W	0h	Emulation Control 0h = TSCNT counter stops immediately on emulation suspend. 1h = TSCNT counter runs until = 0. 2h = TSCNT counter is unaffected by emulation suspend (Run Free). 3h = TSCNT counter is unaffected by emulation suspend (Run Free).
13-9	EVTFLTPS	R/W	0h	Event Filter prescale select: 0h = Divide by 1 (i.e,. no prescale, by-pass the prescaler) 1h = Divide by 2 2h = Divide by 4 3h = Divide by 6 4h = Divide by 8 5h = Divide by 10 1Eh = Divide by 60 1Fh = Divide by 62
8	CAPLDEN	R/W	0h	Enable Loading of PWMSS_ECAP_CAP1 to PWMSS_ECAP_CAP4 registers on a capture event 0h = Disable PWMSS_ECAP_CAP1-PWMSS_ECAP_CAP4 register loads at capture event time. 1h = Enable PWMSS_ECAP_CAP1-PWMSS_ECAP_CAP4 register loads at capture event time.
7	CTRRST4	R/W	0h	Counter Reset on Capture Event 4 0h = Do not reset counter on Capture Event 4 (absolute time stamp operation) 1h = Reset counter after Capture Event 4 time-stamp has been captured (used in difference mode operation)
6	CAP4POL	R/W	0h	Capture Event 4 Polarity select 0h = Capture Event 4 triggered on a rising edge (RE) 1h = Capture Event 4 triggered on a falling edge (FE)

Table 11-778. PWMSS_ECAP_ECCTL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	CTRRST3	R/W	0h	Counter Reset on Capture Event 3 0h = Do not reset counter on Capture Event 3 (absolute time stamp) 1h = Reset counter after Event 3 time-stamp has been captured (used in difference mode operation)
4	CAP3POL	R/W	0h	Capture Event 3 Polarity select 0h = Capture Event 3 triggered on a rising edge (RE) 1h = Capture Event 3 triggered on a falling edge (FE)
3	CTRRST2	R/W	0h	Counter Reset on Capture Event 2 0h = Do not reset counter on Capture Event 2 (absolute time stamp) 1h = Reset counter after Event 2 time-stamp has been captured (used in difference mode operation)
2	CAP2POL	R/W	0h	Capture Event 2 Polarity select 0h = Capture Event 2 triggered on a rising edge (RE) 1h = Capture Event 2 triggered on a falling edge (FE)
1	CTRRST1	R/W	0h	Counter Reset on Capture Event 1 0h = Do not reset counter on Capture Event 1 (absolute time stamp) 1h = Reset counter after Event 1 time-stamp has been captured (used in difference mode operation)
0	CAP1POL	R/W	0h	Capture Event 1 Polarity select 0h = Capture Event 1 triggered on a rising edge (RE) 1h = Capture Event 1 triggered on a falling edge (FE)

11.4.5.8 PWMSS_ECAP_ECCTL2 Register (Offset = 2Ah) [reset = 6h]

PWMSS_ECAP_ECCTL2 is shown in Figure 11-340 and described in Table 11-780.

eCAP Control Register 2.

Table 11-779. PWMSS_ECAP_ECCTL2 Instances

Instance	Physical Address
ECAP_0	021D 182Ah
ECAP_1	021D 1C2Ah

Figure 11-340. PWMSS_ECAP_ECCTL2 Register

15	14	13	12	11	10	9	8
RESERVED					APWMPOL	CAPAPWM	SWSYNC
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
SYNCO_SEL		SYNCI_EN	TSCNTSTP	REARMRESET	STOPVALUE		CONTONESHT
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-3h		R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-780. PWMSS_ECAP_ECCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10	APWMPOL	R/W	0h	APWM output polarity select. This is applicable only in APWM operating mode 0h = Output is active high (Compare value defines high time) 1h = Output is active low (Compare value defines low time)
9	CAPAPWM	R/W	0h	CAP/APWM operating mode select **0h = ECAP module operates in capture mode. This mode forces the following configuration.** 1. Inhibits TSCNT resets via TSCNT = PRD event. 2. Inhibits shadow loads on PWMSS_ECAP_CAP1 and PWMSS_ECAP_CAP2 registers. 3. Permits user to enable PWMSS_ECAP_CAP1-PWMSS_ECAP_CAP4 register load. (d) ECAP input / APWM output pin operates as a capture input.
8	SWSYNC	R/W	0h	Software-forced Counter (TSCNT) Synchronizing. This provides a convenient software method to synchronize some or all ECAP time bases. In APWM mode, the synchronizing can also be done via the TSCNT = PRD event. Note: Selection TSCNT = PRD is meaningful only in APWM mode. However, you can choose it in CAP mode if you find doing so useful. 0h = Writing a zero has no effect. Reading always returns a zero 1h = Writing a one forces a TSCNT shadow load of current ECAP module and any ECAP modules down-stream providing the SYNCO_SEL bits are 0b00. After writing a 1, this bit returns to a zero.
7-6	SYNCO_SEL	R/W	0h	Sync-Out Select 0h = Select sync-in event to be the sync-out signal (pass through) 1h = Select TSCNT = PRD event to be the sync-out signal 2h = Disable sync out signal 3h = Disable sync out signal

Table 11-780. PWMSS_ECAP_ECCTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	SYNCI_EN	R/W	0h	Counter (TSCNT) Sync-In select mode 0h = Disable sync-in option 1h = Enable counter (TSCNT) to be loaded from PWMSS_ECAP_CNTPHS register upon either a SYNCI signal or a S/W force event.
4	TSCNTSTP	R/W	0h	Time Stamp (TSCNT) Counter Stop (freeze) Control 0h = TSCNT stopped 1h = TSCNT free-running
3	REARMRESET	R/W	0h	One-Shot Re-Arming Control, that is, wait for stop trigger. Note: The re-arm function is valid in one shot or continuous mode. 0h = Has no effect (reading always returns a 0) 1h = Arms the one-shot sequence as follows: 1) Resets the Mod4 counter to zero. 2) Unfreezes the Mod4 counter. 3) Enables capture register loads.
2-1	STOPVALUE	R/W	3h	Stop value for one-shot mode. This is the number (between 1 and 4) of captures allowed to occur before the CAP (1 through 4) registers are frozen, that is, capture sequence is stopped. Wrap value for continuous mode. This is the number (between 1 and 4) of the capture register in which the circular buffer wraps around and starts again. Notes: STOPVALUE is compared to Mod4 counter and, when equal, the following two actions occur. (1) Mod4 counter is stopped (frozen). (2) Capture register loads are inhibited. In one-shot mode, further interrupt events are blocked until re-armed. 0h = Stop after Capture Event 1 in one-shot mode. Wrap after Capture Event 1 in continuous mode. 1h = Stop after Capture Event 2 in one-shot mode. Wrap after Capture Event 2 in continuous mode. 2h = Stop after Capture Event 3 in one-shot mode. Wrap after Capture Event 3 in continuous mode. 3h = Stop after Capture Event 4 in one-shot mode. Wrap after Capture Event 4 in continuous mode.
0	CONTONESHT	R/W	0h	Continuous or one-shot mode control (applicable only in capture mode) 0h = Operate in continuous mode 1h = Operate in one-shot mode

11.4.5.9 PWMSS_ECAP_ECEINT Register (Offset = 2Ch) [reset = 0h]

PWMSS_ECAP_ECEINT is shown in Figure 11-341 and described in Table 11-782.

eCAP Interrupt Enable Register.

Table 11-781. PWMSS_ECAP_ECEINT Instances

Instance	Physical Address
ECAP_0	021D 182Ch
ECAP_1	021D 1C2Ch

Figure 11-341. PWMSS_ECAP_ECEINT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CMPEQ	PRDEQ	CNTOVF	CEVT4	CEVT3	CEVT2	CEVT1	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-782. PWMSS_ECAP_ECEINT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	CMPEQ	R/W	0h	Counter Equal **Compare** Interrupt Enable. 0h = Disable Compare Equal as an Interrupt source. 1h = Enable Compare Equal as an Interrupt source.
6	PRDEQ	R/W	0h	Counter Equal **Period** Interrupt Enable. 0h = Disable Period Equal as an Interrupt source. 1h = Enable Period Equal as an Interrupt source.
5	CNTOVF	R/W	0h	Counter Overflow Interrupt Enable. 0h = Disable counter Overflow as an Interrupt source. 1h = Enable counter Overflow as an Interrupt source.
4	CEVT4	R/W	0h	Capture Event 4 Interrupt Enable. 0h = Disable Capture Event 4 as an Interrupt source. 1h = Enable Capture Event 4 as an Interrupt source.
3	CEVT3	R/W	0h	Capture Event 3 Interrupt Enable. 0h = Disable Capture Event 3 as an Interrupt source. 1h = Enable Capture Event 3 as an Interrupt source.
2	CEVT2	R/W	0h	Capture Event 2 Interrupt Enable. 0h = Disable Capture Event 2 as an Interrupt source. 1h = Enable Capture Event 2 as an Interrupt source.
1	CEVT1	R/W	0h	Capture Event 1 Interrupt Enable . 0h = Disable Capture Event 1 as an Interrupt source. 1h = Enable Capture Event 1 as an Interrupt source.
0	RESERVED	R	0h	Reserved

11.4.5.10 PWMSS_ECAP_ECFLG Register (Offset = 2Eh) [reset = 0h]

PWMSS_ECAP_ECFLG is shown in Figure 11-342 and described in Table 11-784.

eCAP Interrupt Flag Register.

Table 11-783. PWMSS_ECAP_ECFLG Instances

Instance	Physical Address
ECAP_0	021D 182Eh
ECAP_1	021D 1C2Eh

Figure 11-342. PWMSS_ECAP_ECFLG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CMPEQ	PRDEQ	CNTOVF	CEVT4	CEVT3	CEVT2	CEVT1	INT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-784. PWMSS_ECAP_ECFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	CMPEQ	R	0h	Compare Equal Compare Status Flag. This flag is only active in APWM mode. 0h = Indicates no event occurred 1h = Indicates the counter (TSCNT) reached the compare register value (ACMP)
6	PRDEQ	R	0h	Counter Equal Period Status Flag. This flag is only active in APWM mode. 0h = Indicates no event occurred 1h = Indicates the counter (TSCNT) reached the period register value (APRD) and was reset.
5	CNTOVF	R	0h	Counter Overflow Status Flag. This flag is active in CAP and APWM mode. 0h = Indicates no event occurred. 1h = Indicates the counter (TSCNT) has made the transition from FFFF FFFFh to 0000 0000h
4	CEVT4	R	0h	Capture Event 4 Status Flag This flag is only active in CAP mode. 0h = Indicates no event occurred 1h = Indicates the fourth event occurred at ECAPn pin
3	CEVT3	R	0h	Capture Event 3 Status Flag. This flag is active only in CAP mode. 0h = Indicates no event occurred. 1h = Indicates the third event occurred at ECAPn pin.
2	CEVT2	R	0h	Capture Event 2 Status Flag. This flag is only active in CAP mode. 0h = Indicates no event occurred. 1h = Indicates the second event occurred at ECAPn pin.
1	CEVT1	R	0h	Capture Event 1 Status Flag. This flag is only active in CAP mode. 0h = Indicates no event occurred. 1h = Indicates the first event occurred at ECAPn pin.
0	INT	R	0h	Global Interrupt Status Flag 0h = Indicates no interrupt generated. 1h = Indicates that an interrupt was generated.

11.4.5.11 PWMSS_ECAP_ECCLR Register (Offset = 30h) [reset = 0h]

PWMSS_ECAP_ECCLR is shown in Figure 11-343 and described in Table 11-786.

eCAP Interrupt Clear Register.

Table 11-785. PWMSS_ECAP_ECCLR Instances

Instance	Physical Address
ECAP_0	021D 1830h
ECAP_1	021D 1C30h

Figure 11-343. PWMSS_ECAP_ECCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CMPEQ	PRDEQ	CNTOVF	CEVT4	CEVT3	CEVT2	CEVT1	INT
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-786. PWMSS_ECAP_ECCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	CMPEQ	R/W	0h	Counter Equal Compare Status Flag 0h = Writing a 0 has no effect. Always reads back a 0 1h = Writing a 1 clears the TSCNT=CMP flag condition
6	PRDEQ	R/W	0h	Counter Equal Period Status Flag 0h = Writing a 0 has no effect. Always reads back a 0 1h = Writing a 1 clears the TSCNT=PRD flag condition
5	CNTOVF	R/W	0h	Counter Overflow Status Flag 0h = Writing a 0 has no effect. Always reads back a 0 1h = Writing a 1 clears the CNTOVF flag condition
4	CEVT4	R/W	0h	Capture Event 4 Status Flag 0h = Writing a 0 has no effect. Always reads back a 0. 1h = Writing a 1 clears the CEVT3 flag condition.
3	CEVT3	R/W	0h	Capture Event 3 Status Flag 0h = Writing a 0 has no effect. Always reads back a 0. 1h = Writing a 1 clears the CEVT3 flag condition.
2	CEVT2	R/W	0h	Capture Event 2 Status Flag 0h = Writing a 0 has no effect. Always reads back a 0. 1h = Writing a 1 clears the CEVT2 flag condition.
1	CEVT1	R/W	0h	Capture Event 1 Status Flag 0h = Writing a 0 has no effect. Always reads back a 0. 1h = Writing a 1 clears the CEVT1 flag condition.
0	INT	R/W	0h	Global Interrupt Clear Flag 0h = Writing a 0 has no effect. Always reads back a 0. 1h = Writing a 1 clears the INT flag and enable further interrupts to be generated if any of the event flags are set to 1.

11.4.5.12 PWMSS_ECAP_ECFRC Register (Offset = 32h) [reset = 0h]

PWMSS_ECAP_ECFRC is shown in Figure 11-344 and described in Table 11-788.

eCAP Interrupt Forcing Register.

Table 11-787. PWMSS_ECAP_ECFRC Instances

Instance	Physical Address
ECAP_0	021D 1832h
ECAP_1	021D 1C32h

Figure 11-344. PWMSS_ECAP_ECFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CMPEQ	PRDEQ	CNTOVF	CEVT4	CEVT3	CEVT2	CEVT1	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-788. PWMSS_ECAP_ECFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	CMPEQ	R/W	0h	Force Counter Equal Compare Interrupt 0h = No effect. Always reads back a 0. 1h = Writing a 1 sets the TSCNT=CMP flag bit.
6	PRDEQ	R/W	0h	Force Counter Equal Period Interrupt 0h = No effect. Always reads back a 0. 1h = Writing a 1 sets the TSCNT=PRD flag bit.
5	CNTOVF	R/W	0h	Force Counter Overflow 0h = No effect. Always reads back a 0. 1h = Writing a 1 to this bit sets the CNTOVF flag bit.
4	CEVT4	R/W	0h	Force Capture Event 4 0h = No effect. Always reads back a 0. 1h = Writing a 1 sets the CEVT4 flag bit
3	CEVT3	R/W	0h	Force Capture Event 3 0h = No effect. Always reads back a 0. 1h = Writing a 1 sets the CEVT3 flag bit
2	CEVT2	R/W	0h	Force Capture Event 2 0h = No effect. Always reads back a 0. 1h = Writing a 1 sets the CEVT2 flag bit.
1	CEVT1	R/W	0h	Force Capture Event 1 0h = No effect. Always reads back a 0. 1h = Writing a 1 sets the CEVT1 flag bit.
0	RESERVED	R	0h	Reserved

11.4.5.13 PWMSS_ECAP_PID Register (Offset = 5Ch) [reset = 44D2 2100h]

PWMSS_ECAP_PID is shown in Figure 11-345 and described in Table 11-790.

eCAP Revision ID.

Table 11-789. PWMSS_ECAP_PID Instances

Instance	Physical Address
ECAP_0	021D 185Ch
ECAP_1	021D 1C5Ch

Figure 11-345. PWMSS_ECAP_PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REVISION																															
R-44D2 2100h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-790. PWMSS_ECAP_PID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REVISION	R	44D2 2100h	TI internal data. Identifies revision of peripheral.

11.5 Enhanced PWM (ePWM) Module

This chapter describes the Enhanced PWM (ePWM) module for the device.

11.5.1 ePWM Overview

An effective PWM peripheral must be able to generate complex pulse width waveforms with minimal CPU overhead or intervention. It needs to be highly programmable and very flexible while being easy to understand and use. The ePWM unit described here addresses these requirements by allocating all needed timing and control resources on a per PWM channel basis. Cross coupling or sharing of resources has been avoided; instead, the ePWM is built up from smaller single channel modules with separate resources and that can operate together as required to form a system. This modular approach results in an orthogonal architecture and provides a more transparent view of the peripheral structure, helping users to understand its operation quickly.

In the further description the letter x within a signal or module name is used to indicate a generic ePWM instance on a device. For example, output signals EPWMxA and EPWMxB refer to the output signals from the ePWM_x instance. Thus, EPWM1A and EPWM1B belong to ePWM_1, EPWM2A and EPWM2B belong to ePWM_2, and so forth.

The ePWM module represents one complete PWM channel composed of two PWM outputs: EPWMxA and EPWMxB. A given ePWM module functionality can be extended with the so called **High-Resolution Pulse Width modulator**. Refer to the Section 11.6.3, to determine which ePWM instances include the HRPWM feature. The HRPWM functionalities are described in Section 11.5.4.9.

As also described in Section 11.5.3.2, the ePWM modules are chained together via a clock synchronization scheme that allows them to operate as a single system when required. Additionally, the ePWM integration allows this synchronization scheme to be extended to the capture peripheral modules (eCAP). The number of modules is device-dependent and based on target application needs. Modules can also operate stand-alone.

Each ePWM module supports the following features:

- Dedicated 16-bit time-base counter with period and frequency control
- Two PWM outputs (EPWMxA and EPWMxB) that can be used in the following configurations:
  - Two independent PWM outputs with single-edge operation
  - Two independent PWM outputs with dual-edge symmetric operation
  - One independent PWM output with dual-edge asymmetric operation
- Asynchronous override control of PWM signals through software
- Programmable phase-control support for lag or lead operation relative to other ePWM modules
- Hardware-locked (synchronized) phase relationship on a cycle-by-cycle basis
- Dead-band generation with independent rising and falling edge delay control
- Programmable trip zone allocation of both cycle-by-cycle trip and one-shot trip on fault conditions
- A trip condition can force either high, low, or high-impedance state logic levels at PWM outputs
- Allows events to trigger both CPU interrupts and ADC start of conversions
- Programmable event prescaling minimizes CPU overhead on interrupts
- PWM chopping by high-frequency carrier signal, useful for pulse transformer gate drives.

Each ePWM module is connected to the input/output signals shown in Figure 11-346. The signals are described in detail in subsequent sections.

The order in which the ePWM modules are connected may differ from what is shown in Figure 11-346. See Section 11.5.3.2 for the actual synchronization scheme implemented in the device. Each ePWM module consists of eight submodules and is connected within a system via the signals shown in Figure 11-347.

Figure 11-346. Multiple ePWM Modules

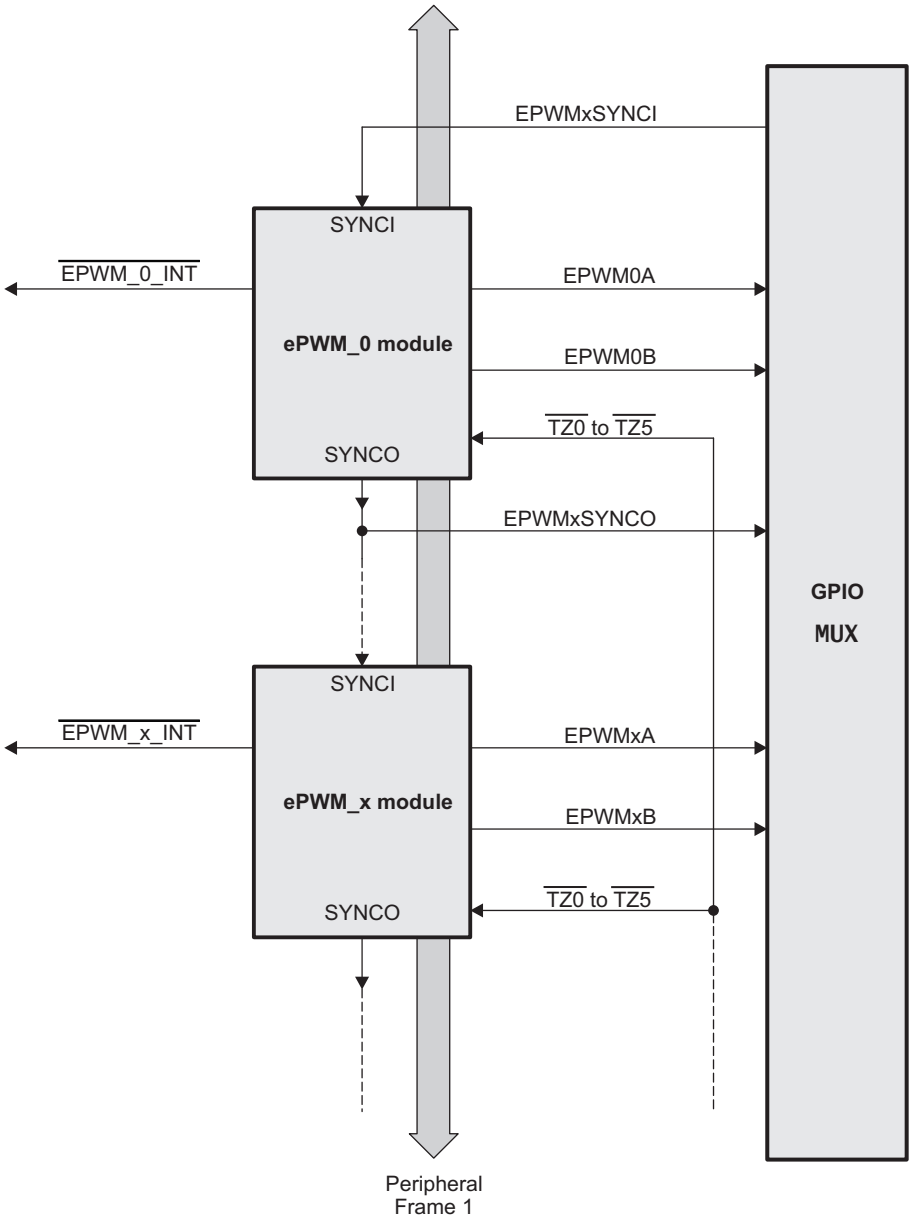


Figure 11-347. Submodules and Signal Connections for an ePWM Module

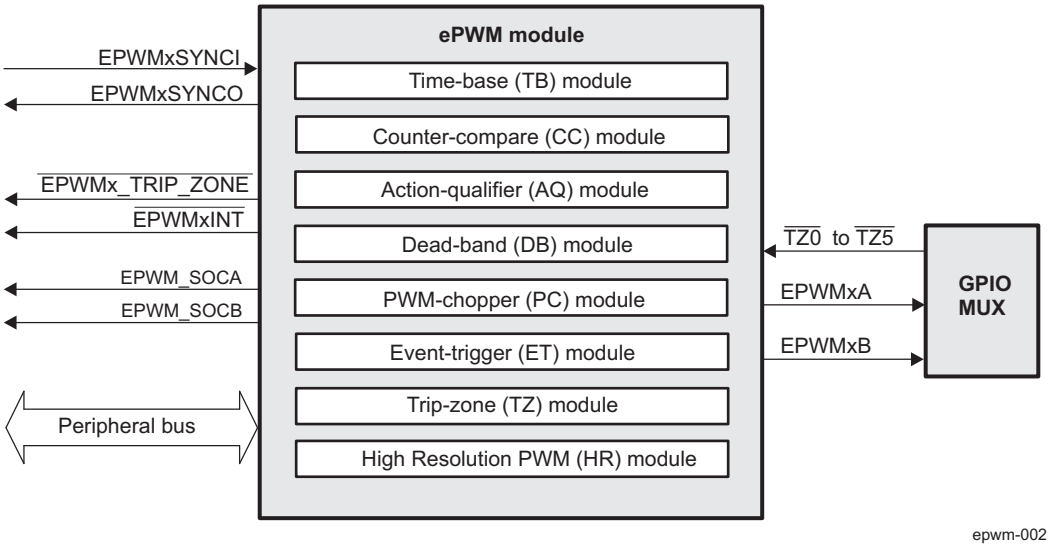
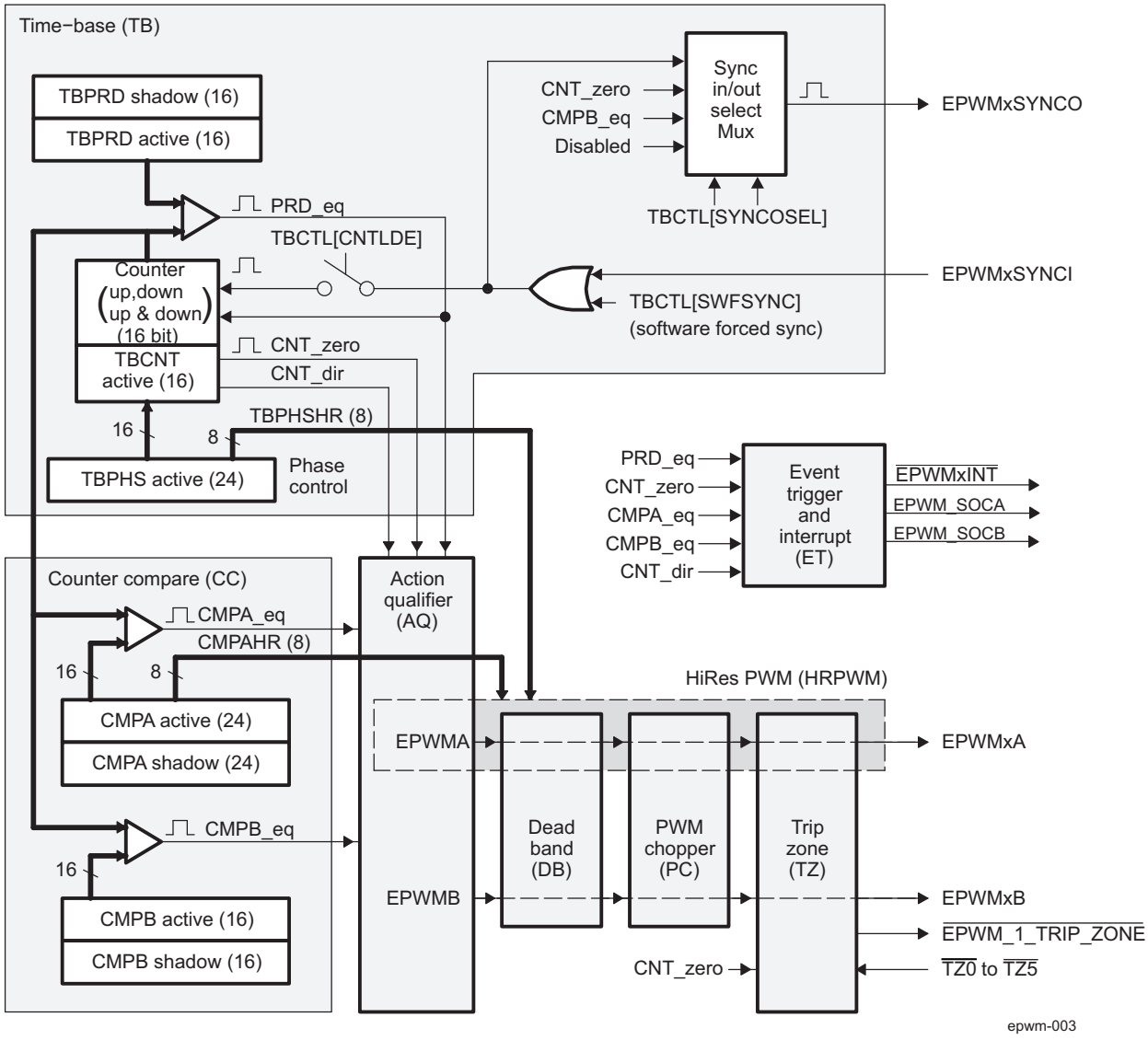


Figure 11-348 shows more internal details of a single ePWM module. The main signals used by the ePWM module are:

- **PWM output signals (EPWMxA and EPWMxB).** The PWM output signals are made available external to the device through the GPIO peripheral described in the system control and interrupts guide for the device.
- **Trip-zone signals ($\overline{TZ0}$ to $\overline{TZ5}$).** These input signals alert the ePWM module of an external fault condition. Each module on a device can be configured to either use or ignore any of the trip-zone signals. The trip-zone signal can be configured as an asynchronous input through the GPIO peripheral.
- **Time-base synchronization input (EPWMxSYNCI) and output (EPWMxSYNCO) signals.** The synchronization signals daisy chain the ePWM modules together. Each module can be configured to either use or ignore its synchronization input. The clock synchronization input and output signal are brought out to pins for ePWM_0 (ePWM module 0) and ePWM_3. The ePWM_5 synchronization output (EPWM5SYNCO) is also connected to the input SyncIn of the first enhanced capture module (eCAP_0).
- **ADC start-of-conversion signals (EPWMxSOCA and EPWMxSOCB).** Each ePWM module has two ADC start of conversion signals (one for each sequencer). Any ePWM module can trigger a start of conversion for either sequencer. Which event triggers the start of conversion is configured in the Event-Trigger submodule of the ePWM.
- **Peripheral Bus.** The peripheral bus is 32-bits wide and allows both 16-bit and 32-bit writes to the ePWM register file.

Figure 11-348 also shows the key internal submodule interconnect signals. Each submodule is described in Section 11.5.4.

Figure 11-348. ePWM Submodules and Critical Internal Signal Interconnects



11.5.2 ePWM Environment

11.5.2.1 ePWM I/O Interface

Table 11-791 shows the device integrated ePWM subsystems interface signals to external devices.

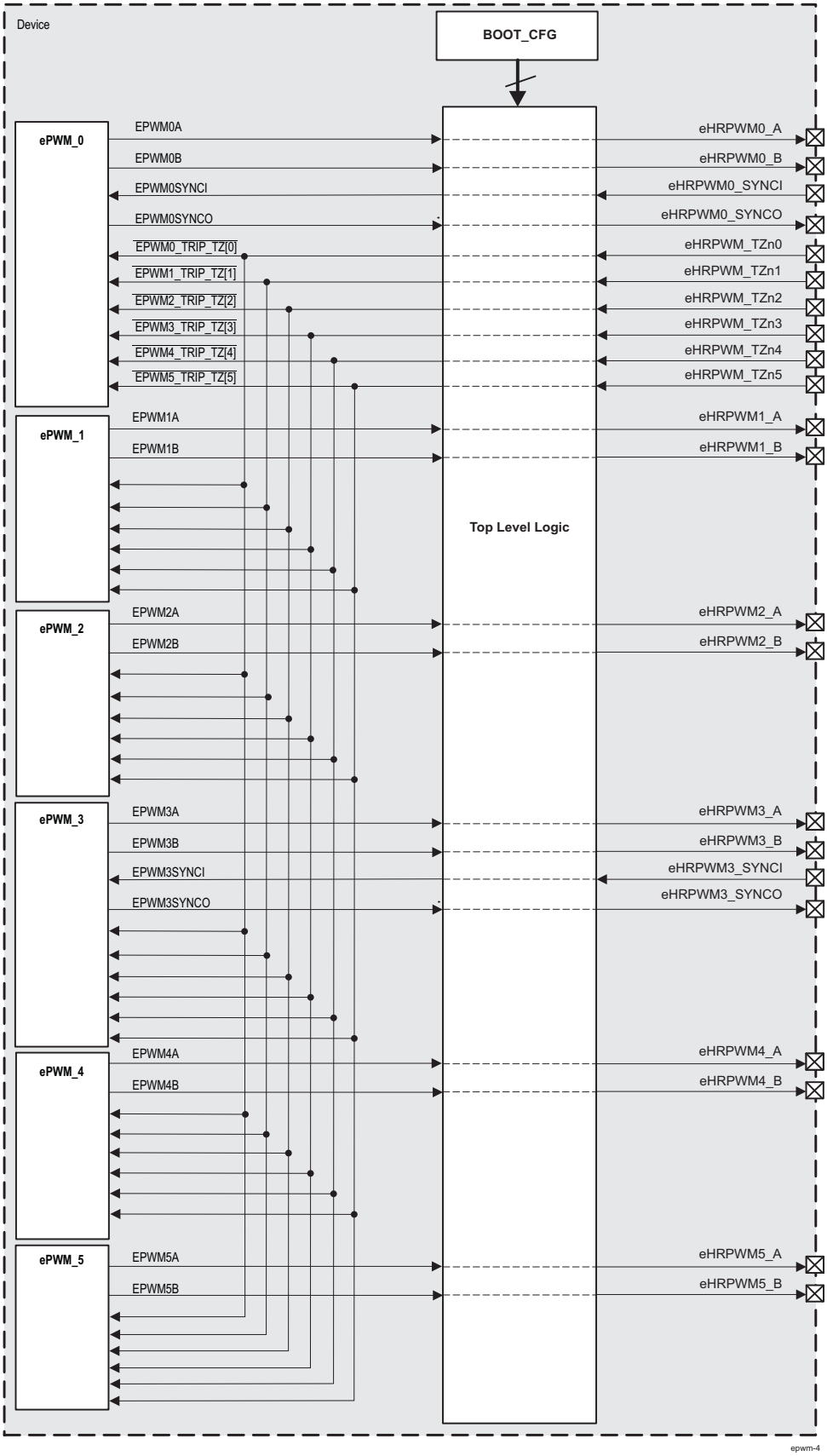
Table 11-791. ePWM Subsystems I/O Signals

ePWM Modules Level Signal Name	Device Level Signal Name	I/O Type	Description	Module Pin Reset Value
ePWM_0				
EPWM0A	eHRPWM0_A	O	ePWM_0 output A	0
EPWM0B	eHRPWM0_B	O	ePWM_0 output B	0
EPWM0SYNCI	eHRPWM0_SYNCI	I	ePWM_0 Sync input	HiZ
EPWM0SYNCO	eHRPWM0_SYNCO	O	ePWM_0 Sync output	0
$\overline{\text{EPWM0\_TRIP\_TZ[0]}}$	eHRPWM_TZn0	I	ePWM_0 TripZone input	HiZ
ePWM_1				

Table 11-791. ePWM Subsystems I/O Signals (continued)

ePWM Modules Level Signal Name	Device Level Signal Name	I/O Type	Description	Module Pin Reset Value
EPWM1A	eHRPWM1_A	O	ePWM_1 output A	0
EPWM1B	eHRPWM1_B	O	ePWM_1 output B	0
$\overline{\text{EPWM1\_TRIP\_TZ[1]}}$	eHRPWM_TZn1	I	ePWM_1 TripZone input	HiZ
ePWM_2				
EPWM2A	eHRPWM2_A	O	ePWM_2 output A	0
EPWM2B	eHRPWM2_B	O	ePWM_2 output B	0
$\overline{\text{EPWM2\_TRIP\_TZ[2]}}$	eHRPWM_TZn2	I	ePWM_2 TripZone input	HiZ
ePWM_3				
EPWM3A	eHRPWM3_A	O	ePWM_3 output A	0
EPWM3B	eHRPWM3_B	O	ePWM_3 output B	0
EPWM3SYNCI	eHRPWM3_SYNCI	I	ePWM_3 Sync input	HiZ
EPWM3SYNCO	eHRPWM3_SYNCO	O	ePWM_3 Sync output	0
$\overline{\text{EPWM3\_TRIP\_TZ[3]}}$	eHRPWM_TZn3	I	ePWM_3 TripZone input	HiZ
ePWM_4				
EPWM4A	eHRPWM4_A	O	ePWM_4 output A	0
EPWM4B	eHRPWM4_B	O	ePWM_4 output B	0
$\overline{\text{EPWM4\_TRIP\_TZ[4]}}$	eHRPWM_TZn4	I	ePWM_4 TripZone input	HiZ
ePWM_5				
EPWM5A	eHRPWM5_A	O	ePWM_5 output A	0
EPWM5B	eHRPWM5_B	O	ePWM_5 output B	0
$\overline{\text{EPWM5\_TRIP\_TZ[5]}}$	eHRPWM_TZn5	I	ePWM_5 TripZone input	HiZ
ePWM Start of Conversion				
EPWM_SOCA	eHRPWM_SOCA	O	ePWM start of conversion output A	OZ
EPWM_SOCB	eHRPWM_SOCB	O	ePWM start of conversion output B	OZ

Figure 11-349. ePWM External Interface I/Os



11.5.3 ePWM Integration

This section describes module integration in the device, including information about clocks, resets, and hardware requests.

Figure 11-350 shows the ePWM integration.

There are 6 instances of the ePWM integrated in the device. Each of the Enhanced Pulse Width Modulator (ePWM) includes an Enhanced High Resolution Modulator (eHRPWM).

At system level the ePWM_0 through ePWM_5 integration features are listed below:

- A 32-bit slave configuration port on the TeraNet_CFG interconnect.
- A single functional clock from PLL_CONTROLLER shared between the 5 ePWM modules.
- 2 hardware events per ePWM, that is a total of 12 events. From these events each ePWM:
  - generates 2 interrupts to the device ARM_GIC, CIC and ICSS_1_INTC
  - generates 2 DMA requests, mapped to the device EDMACC_1
- A synchronization input/output daisy chain-like connection exists between the ePWM_0 trough ePWM_5. For more details, refer to Section 11.5.3.2, *Daisy-Chain Connectivity between ePWM Modules*.

Figure 11-350. ePWM Integration

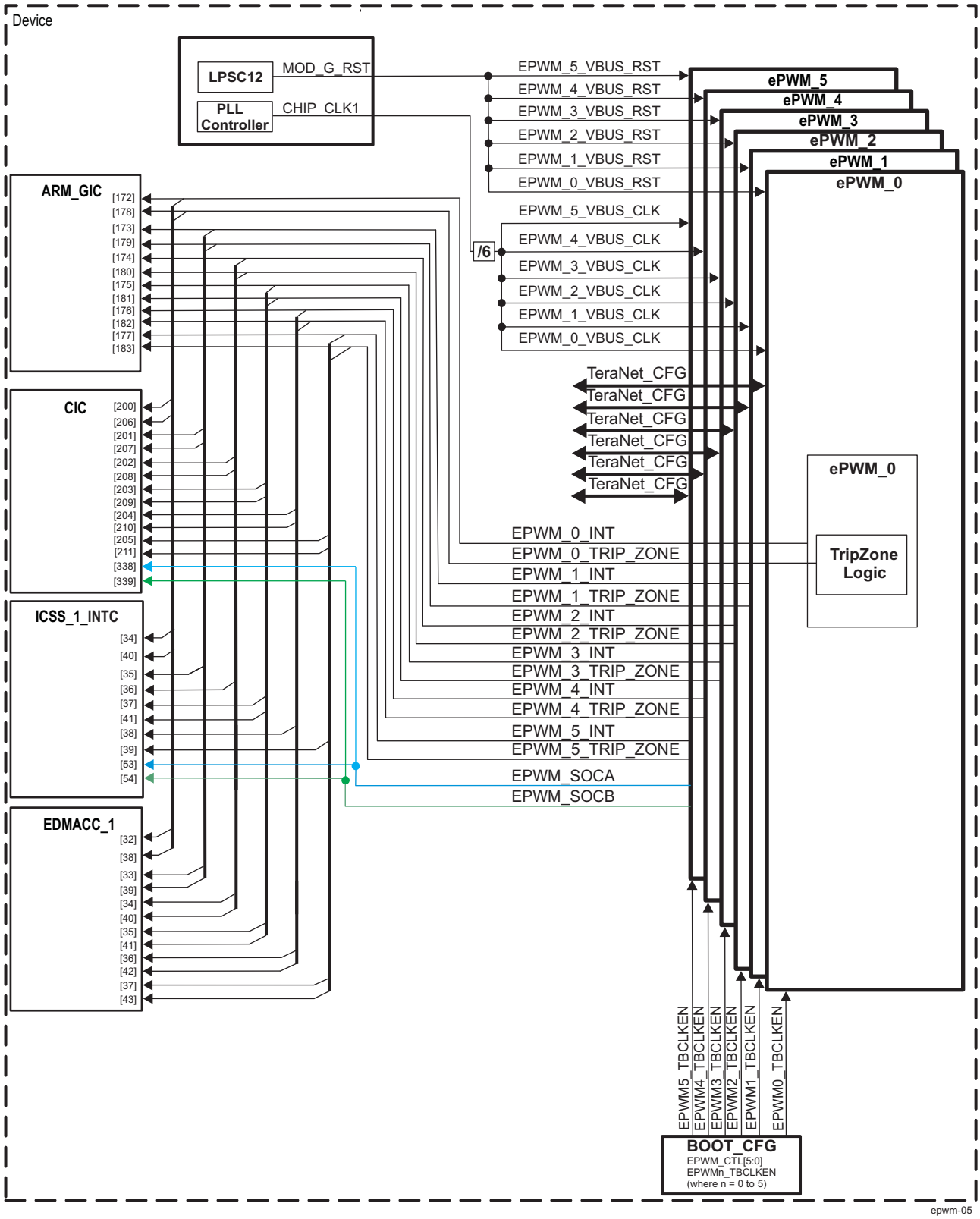


Table 11-924 through Table 11-3555 summarize the integration of the module in the device.

Table 11-792. ePWM Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
ePWM_0	PD5	LPSC12	TeraNet_CFG
ePWM_1	PD5	LPSC12	TeraNet_CFG
ePWM_2	PD5	LPSC12	TeraNet_CFG
ePWM_3	PD5	LPSC12	TeraNet_CFG
ePWM_4	PD5	LPSC12	TeraNet_CFG
ePWM_5	PD5	LPSC12	TeraNet_CFG

Table 11-793. ePWM Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
ePWM_0	EPWM_0_VBUS_CLK	CHIP_CLK1 / 6	PLL_CONTROLLER	ePWM_0 functional and interface clock
ePWM_1	EPWM_1_VBUS_CLK	CHIP_CLK1 / 6	PLL_CONTROLLER	ePWM_1 functional and interface clock
ePWM_2	EPWM_2_VBUS_CLK	CHIP_CLK1 / 6	PLL_CONTROLLER	ePWM_2 functional and interface clock
ePWM_3	EPWM_3_VBUS_CLK	CHIP_CLK1 / 6	PLL_CONTROLLER	ePWM_3 functional and interface clock
ePWM_4	EPWM_4_VBUS_CLK	CHIP_CLK1 / 6	PLL_CONTROLLER	ePWM_4 functional and interface clock
ePWM_5	EPWM_5_VBUS_CLK	CHIP_CLK1 / 6	PLL_CONTROLLER	ePWM_5 functional and interface clock
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
ePWM_0	EPWM_0_VBUS_RST	MOD_G_RST	LPSC12	Module Reset
ePWM_1	EPWM_1_VBUS_RST	MOD_G_RST	LPSC12	Module Reset
ePWM_2	EPWM_2_VBUS_RST	MOD_G_RST	LPSC12	Module Reset
ePWM_3	EPWM_3_VBUS_RST	MOD_G_RST	LPSC12	Module Reset
ePWM_4	EPWM_4_VBUS_RST	MOD_G_RST	LPSC12	Module Reset
ePWM_5	EPWM_5_VBUS_RST	MOD_G_RST	LPSC12	Module Reset

Table 11-794. ePWM Hardware Requests

Interrupt Requests					
Module Instance	**Event Name**	**Mapped To Input Event [Number]**			**Description**
		ARM_GIC	**CIC**	**ICSS_1_INTC**	
ePWM_0	EPWM_0_INT	[172]	[200]	[34]	ePWM_0 interrupt
	EPWM_0_TRIP_ZONE	[178]	[206]	[40]	ePWM_0 Tripzone interrupt
ePWM_1	EPWM_1_INT	[173]	[201]	[35]	ePWM_1 interrupt
	EPWM_1_TRIP_ZONE	[179]	[207]	–	ePWM_1 Tripzone interrupt
ePWM_2	EPWM_2_INT	[174]	[202]	[36]	ePWM_2 interrupt
	EPWM_2_TRIP_ZONE	[180]	[208]	–	ePWM_2 Tripzone interrupt
ePWM_3	EPWM_3_INT	[175]	[203]	[37]	ePWM_3 interrupt
	EPWM_3_TRIP_ZONE	[181]	[209]	[41]	ePWM_3 Tripzone interrupt
ePWM_4	EPWM_4_INT	[176]	[204]	[38]	ePWM_4 interrupt
	EPWM_4_TRIP_ZONE	[182]	[210]	–	ePWM_4 Tripzone interrupt

Table 11-794. ePWM Hardware Requests (continued)

ePWM_5	EPWM_5_INT	[177]	[205]	[39]	ePWM_5 interrupt
	EPWM_5_TRIP_ZONE	[183]	[211]	–	ePWM_5 Tripzone interrupt
ePWMx	EPWM_SOCA	–	[338]	[53]	Start of Conversion A event
ePWMx	EPWM_SOCB	–	[339]	[54]	Start of Conversion B event

DMA Requests

Module Instance	Event Name	Mapped To Input Event [Number]		Description
		EDMACC_0	EDMACC_1	
ePWM_0	EPWM_0_INT	–	[32]	ePWM_0 event to the EDMA_1
	EPWM_0_TRIP_ZONE	–	[38]	ePWM_0 Tripzone event to the EDMA_1
ePWM_1	EPWM_1_INT	–	[33]	ePWM_1 event to the EDMA_1
	EPWM_1_TRIP_ZONE	–	[39]	ePWM_1 Tripzone event to the EDMA_1
ePWM_2	EPWM_2_INT	–	[34]	ePWM_2 event to the EDMA_1
	EPWM_2_TRIP_ZONE	–	[40]	ePWM_2 Tripzone event to the EDMA_1
ePWM_3	EPWM_3_INT	–	[35]	ePWM_3 event to the EDMA_1
	EPWM_3_TRIP_ZONE	–	[41]	ePWM_3 Tripzone event to the EDMA_1
ePWM_4	EPWM_4_INT	–	[36]	ePWM_4 event to the EDMA_1
	EPWM_4_TRIP_ZONE	–	[42]	ePWM_4 Tripzone event to the EDMA_1
ePWM_5	EPWM_5_INT	–	[37]	ePWM_5 event to the EDMA_1
	EPWM_5_TRIP_ZONE	–	[43]	ePWM_5 Tripzone event to the EDMA_1

11.5.3.1 Device Specific ePWM Features

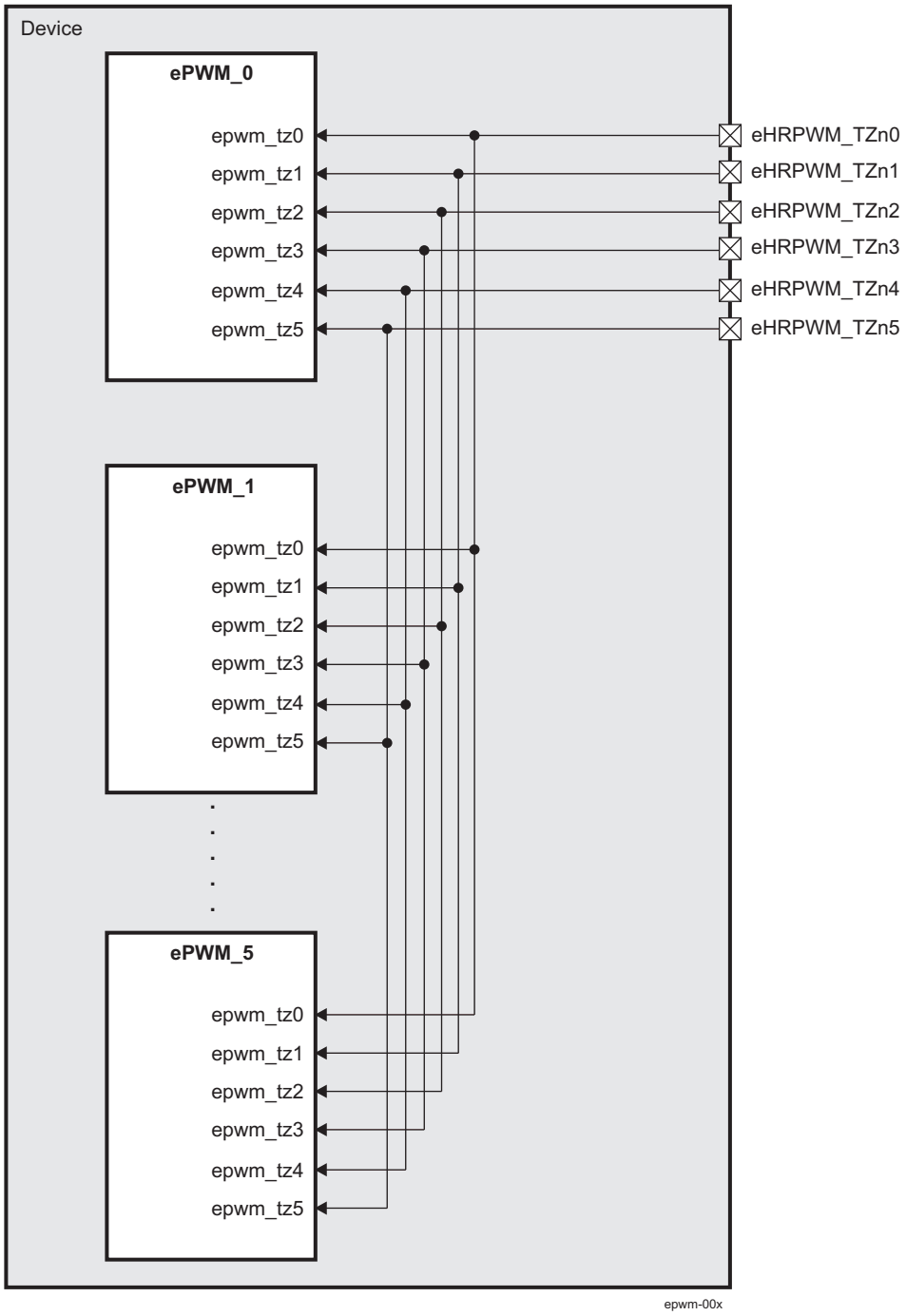
A High-Resolution PWM (HRPWM) modulator is added to each of the device ePWM modules.

NOTE: The HRPWM option applies only to the ePWMxA output channel. The ePWMxB channel has conventional capabilities. For more details on HRPWM features of ePWM modules refer to the Section 11.5.4.9.

The ePWM module interface has some restrictions in functionality as listed below:

- **For ePWM comparators — only the outputs (ehrpwmxA and ehrpwmxB, where x = 0 to 5) are available at chip level**
- **Each ePWM supports 6 tripzone event inputs and each of those inputs from all six of the ePWMs are tied to a common tripzone_input pin so that any ePWM can be triggered by any of the six tripzone inputs as shown in Figure 11-351.**

Figure 11-351. ePWM Tripzone Connectivity Detail



The ePWM functional interface signals which are NOT available to user are summarized in Table 11-927.

Table 11-795. Device Limitations for the ePWM Functional Interfaces

Module Interface	Signal	Description	Comment
ePWM_n (where n = 0 to 5 for the device)	EPWM_COMP_EPWMDCMAH	ePWM Comparator A input (HIGH)	Not available at chip boundary (can not be used)
	EPWM_COMP_EPWMDCMAL	ePWM Comparator A input (LOW)	
	EPWM_COMP_EPWMDCMBH	ePWM Comparator B input (HIGH)	
	EPWM_COMP_EPWMDCMBL	ePWM Comparator B input (LOW)	
	EPWM_TRIP_TZ_O[5:0]	ePWM Tripzone outputs	
	EPWM_TRIP_TZ_OEN[5:0]	ePWM Tripzone outputs enable	

NOTE: Only the SYNCI input and SYNCO output signals of ePWM_0 and ePWM_3 modules are available at chip-level. Respective names of the signals are: eHRPWM0_SYNCI and eHRPWM0_SYNCO (for ePWM_0) and eHRPWM3_SYNCI and eHRPWM3_SYNCO (for ePWM_3). For more information, see Section 11.5.2, *ePWM Environment*.

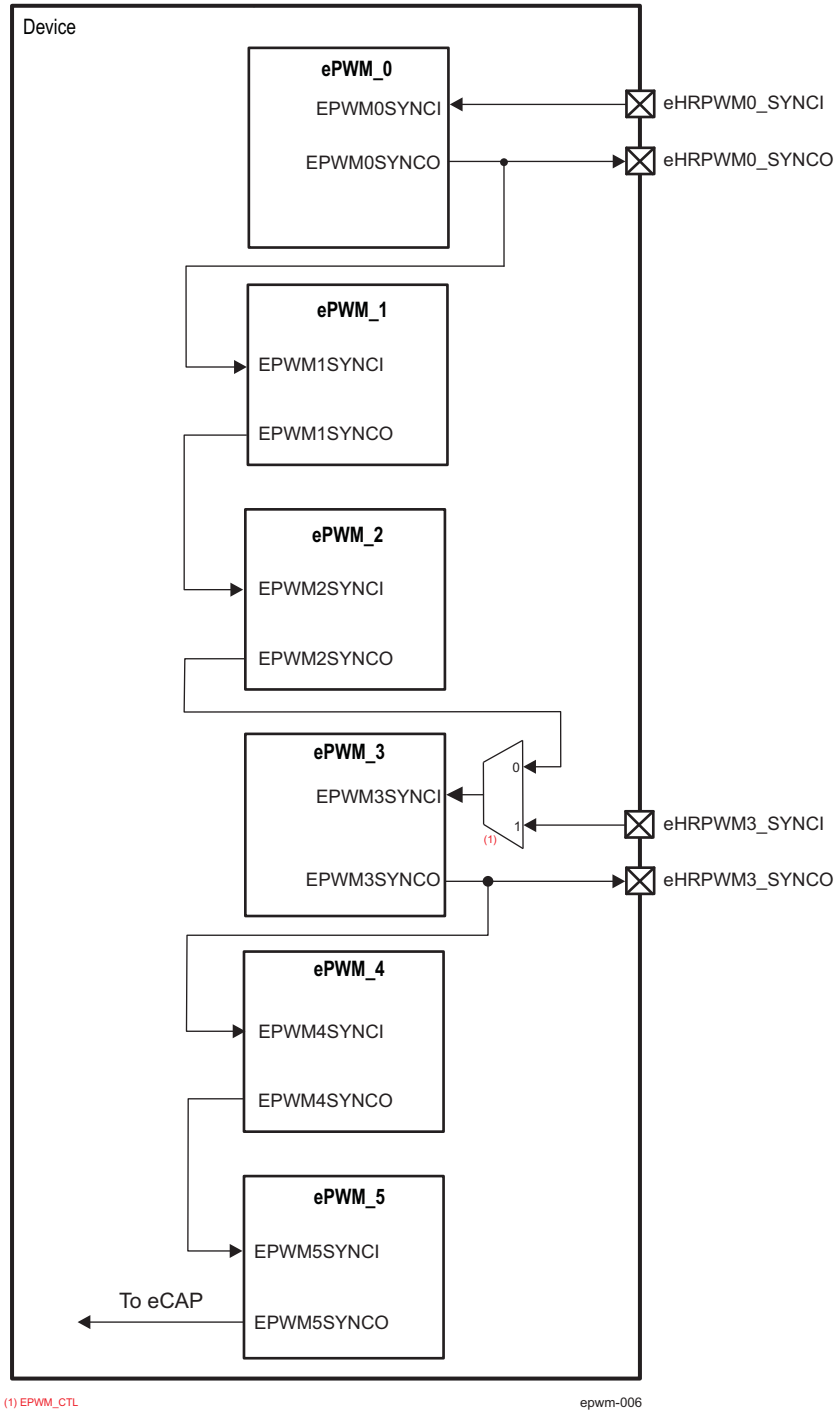
11.5.3.2 Daisy-Chain Connectivity between ePWM Modules

For synchronization to other modules or events ePWMs are provided with synchronization signals. In the device, these signals are connected in a daisy-chain fashion as shown in Figure 11-352. The ePWM_0 input synchronization signal is terminated at a device pad (eHRPWM0_SYNCI signal), such that device external sync events can be directly applied only to the ePWM_0 submodule. The ePWM_3 through ePWM_5 modules can be synchronized either to a separate external signal on the eHRPWM3_SYNCI pin (routed through the PRU-ICSS module) or to ePWM_2 output synchronization signal. The ePWM_3 input synchronization signal is selected using the corresponding EPWM3_SYNCSEL bit of the EPWM_CTL register in the BOOT_CFG module. A synchronization output is available from the ePWM_0 on the eHRPWM0_SYNCO output pin and from ePWM_3 on the eHRPWM3_SYNCO pin.

NOTE: The ePWM_0 EPWM0SYNCI signal (device eHRPWM0_SYNCI input signal) triggers the event of the ePWM_0 Phase Register being loaded into the Counter register (TBPHS -> TBCNT). This event is synchronous to the ePWM_0 **time-base clock** (TBCLK).

The ePWM_0 EPWM0SYNCO (device eHRPWM0_SYNCO output signal) is implicitly synchronous to the time-base clock, as this signal has a programmable source of event (in EPWM_TBCTL[5-4] SYNCOSEL) triggered synchronously to the ePWM_0 TBCLK.

Figure 11-352. Daisy-Chain Connectivity between ePWM Modules

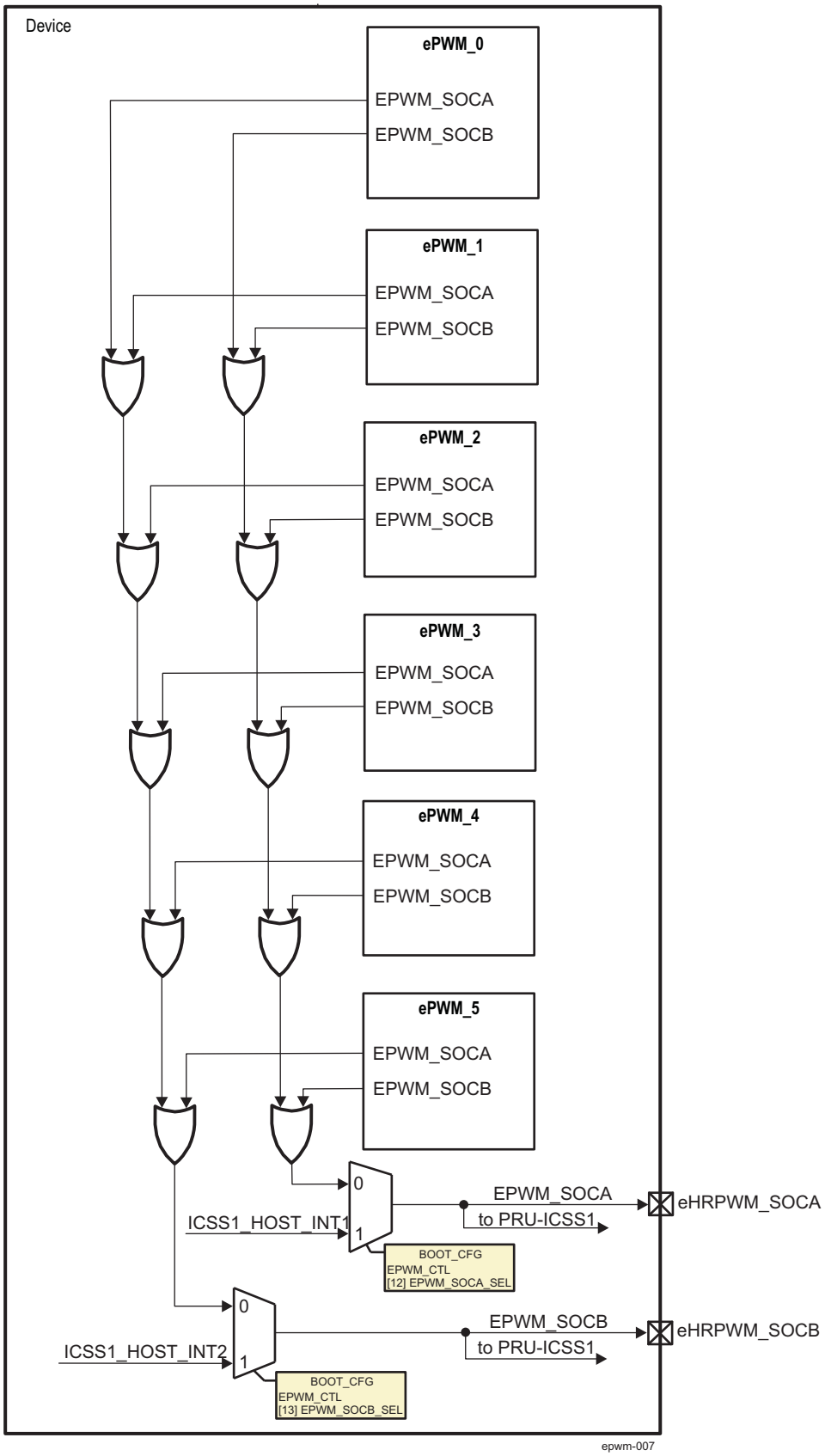


11.5.3.3 ADC start of conversion signals (EPWM_SOCA and EPWM_SOCB).

Each ePWM module provides start of conversion events (SOCA and SOCB), that can be used to start conversion of external ADC module. In this device all SOCA and all SOCB events are ORed together and hence any of the 6 ePWMs can initiate an ADC start of conversion. These events are then multiplexed with an PRU-ICSS1 host interrupts (ICSS1_HOST_INT1 or ICSS1_HOST_INT2), allowing either PRU-ICSS1 or ePWM_0 through ePWM_5 modules to trigger the SOCA/SOCB event pins (eHRPWM_SOCA and eHRPWM_SOCB). The EPWM_SOCA and EPWM_SOCB event output signals are also routed back to the PRU-ICSS1 module as interrupts as shown in Figure 11-353

NOTE: ICSS1_HOST_INT1 and ICSS1_HOST_INT2 interrupts are mapped to Host-3 and Host-4 of the PRU-ICSS Interrupt Controller.

Figure 11-353. Interconnectivity of ADC start of conversion



11.5.3.4 ePWM Modules Time Base Clock Gating

Each ePWM module has an EPWMTBCLKEN module input used to individually **enable / disable its ePWM time-base clock**. The ePWM time-base clock enable input comes from the EPWM_CTL register, as follows:

- ePWM_0 -> EPWM_CTL[0] EPWM0_TBCLKEN bit
- ePWM_1 -> EPWM_CTL[1] EPWM1_TBCLKEN bit
- ePWM_2 -> EPWM_CTL[2] EPWM2_TBCLKEN bit
- ePWM_3 -> EPWM_CTL[3] EPWM3_TBCLKEN bit
- ePWM_4 -> EPWM_CTL[4] EPWM4_TBCLKEN bit
- ePWM_5 -> EPWM_CTL[5] EPWM5_TBCLKEN bit.

This individual TBCLKEN control can be used to align the ePWM time base clock. EPWMn_TBCLKEN (where n = 0 to 5) bit set to 0b0, holds the TBCLK generation counter in its reset state. When EPWMn_TBCLKEN is set to 0b1, then the TBCLK generation counter is allowed to count.

11.5.4 ePWM Functional Description

8 submodules are included in each ePWM peripheral. Each of these submodules performs specific tasks that can be configured by software.

11.5.4.1 ePWM Submodule Features

Table 11-796 lists the eight key submodules together with a list of their main configuration parameters.

Table 11-796. Submodule Configuration Parameters

Submodule	Configuration Parameter or Option
Time-base (TB)	• Scale the time-base clock (TBCLK) relative to the system clock (VBUS_CLK). • Configure the PWM time-base counter (TBCNT) frequency or period. • Time-base counter mode selection: – count-up mode: used for asymmetric PWM – count-down mode: used for asymmetric PWM – count-up-and-down mode: used for symmetric PWM • Configure the time-base phase relative to another ePWM module. • Synchronize the time-base counter between modules through hardware or software. • Configure the direction (up or down) of the time-base counter after a synchronization event. • Configure how the time-base counter will behave when the device is halted by an emulator. • Specify the source for the synchronization output of the ePWM module: – Synchronization input signal – Time-base counter equal to zero – Time-base counter equal to counter-compare B (CMPB) – No output synchronization signal generated.
Counter-compare (CC)	• Specify the PWM duty cycle for output EPWMxA and/or output EPWMxB • Specify the time at which switching events occur on the EPWMxA or EPWMxB output
Action-qualifier (AQ)	• Specify the type of action taken when a time-base or counter-compare submodule event occurs: – No action taken – Output EPWMxA and/or EPWMxB switched high – Output EPWMxA and/or EPWMxB switched low – Output EPWMxA and/or EPWMxB toggled • Force the PWM output state through software control • Configure and control the PWM dead-band through software
Dead-band (DB)	• Control of traditional complementary dead-band relationship between upper and lower switches • Specify the output rising-edge-delay value • Specify the output falling-edge delay value • Bypass the dead-band module entirely. In this case the PWM waveform is passed through without modification.
PWM-chopper (PC)	• Create a chopping (carrier) frequency. • Pulse width of the first pulse in the chopped pulse train. • Duty cycle of the second and subsequent pulses. • Bypass the PWM-chopper module entirely. In this case the PWM waveform is passed through without modification.

Table 11-796. Submodule Configuration Parameters (continued)

Submodule	Configuration Parameter or Option
Trip-zone (TZ)	• Configure the ePWM module to react to one, all, or none of the trip-zone pins. • Specify the tripping action taken when a fault occurs: – Force EPWMxA and/or EPWMxB high – Force EPWMxA and/or EPWMxB low – Force EPWMxA and/or EPWMxB to a high-impedance state – Configure EPWMxA and/or EPWMxB to ignore any trip condition. • Configure how often the ePWM will react to the trip-zone pin: – One-shot – Cycle-by-cycle • Enable the trip-zone to initiate an interrupt. • Bypass the trip-zone module entirely.
Event-trigger (ET)	• Enable the ePWM events that will trigger an interrupt. • Specify the rate at which events cause triggers (every occurrence or every second or third occurrence) • Poll, set, or clear event flags
High-Resolution PWM (HRPWM)	• Enable extended time resolution capabilities • Configure finer time granularity control or edge positioning

Some examples on various ePWM module configurations are shown below. These examples use the constant definitions shown in Example 11-1.

Example 11-1. Constant Definitions Used in the ePWM Code Examples

```
// TBCTL (Time-Base Control)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// TBCNT MODE bits
#define         TB_COUNT_UP         0x0
#define         TB_COUNT_DOWN       0x1
#define         TB_COUNT_UPDOWN     0x2
#define         TB_FREEZE           0x3
// PHSEN bit
#define         TB_DISABLE          0x0
#define         TB_ENABLE           0x1
// PRDLD bit
#define         TB_SHADOW           0x0
#define         TB_IMMEDIATE        0x1
// SYNCOSEL bits
#define         TB_SYNC_IN          0x0
#define         TB_CTR_ZERO         0x1
#define         TB_CTR_CMPB         0x2
#define         TB_SYNC_DISABLE     0x3
// HSPCLKDIV and CLKDIV bits
#define         TB_DIV1             0x0
#define         TB_DIV2             0x1
#define         TB_DIV4             0x2
// PHSDIR bit
#define         TB_DOWN             0x0
#define         TB_UP               0x1

// CMPCTL (Compare Control)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// LOADAMODE and LOADBMODE bits
#define         CC_CTR_ZERO         0x0
#define         CC_CTR_PRD          0x1
#define         CC_CTR_ZERO_PRD     0x2
#define         CC_LD_DISABLE       0x3
// SHDWAMODE and SHDWBMODE bits
```

Example 11-1. Constant Definitions Used in the ePWM Code Examples (continued)

```
#define         CC_SHADOW          0x0
#define         CC_IMMEDIATE       0x1
// AQCTLA and AQCTLB (Action-qualifier Control)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// ZRO, PRD, CAU, CAD, CBU, CBD bits
#define         AQ_NO_ACTION       0x0
#define         AQ_CLEAR           0x1
#define         AQ_SET             0x2
#define         AQ_TOGGLE          0x3
// DBCTL (Dead-Band Control)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// MODE bits
#define         DB_DISABLE         0x0
#define         DBA_ENABLE         0x1
#define         DBB_ENABLE         0x2
#define DB_FULL_ENABLE 0x3
// POLSEL bits
#define         DB_ACTV_HI         0x0
#define         DB_ACTV_LOC        0x1
#define         DB_ACTV_HIC        0x2
#define         DB_ACTV_LO         0x3
// PCCTL (chopper control)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// CHPEN bit
#define         CHP_ENABLE         0x0
#define CHP_DISABLE 0x1
// CHPFREQ bits
#define         CHP_DIV1           0x0
#define         CHP_DIV2           0x1
#define         CHP_DIV3           0x2
#define         CHP_DIV4           0x3
#define         CHP_DIV5           0x4
#define         CHP_DIV6           0x5
#define         CHP_DIV7           0x6
#define         CHP_DIV8           0x7
// CHPDUTY bits
#define         CHP1_8TH           0x0
#define         CHP2_8TH           0x1
#define         CHP3_8TH           0x2
#define         CHP4_8TH           0x3
#define         CHP5_8TH           0x4
#define         CHP6_8TH           0x5
#define         CHP7_8TH           0x6
// TZSEL (Trip-zone Select)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// CBCn and OSHTn bits
#define         TZ_ENABLE          0x0
#define         TZ_DISABLE         0x1
// TZCTL (Trip-zone Control)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// TZA and TZB bits
#define         TZ_HIZ             0x0
#define         TZ_FORCE_HI        0x1
#define         TZ_FORCE_LO        0x2
#define         TZ_DISABLE         0x3
// ETSEL (Event-trigger Select)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// INTSEL, SOCASEL, SOCBSEL bits
#define         ET_CTR_ZERO        0x1
#define         ET_CTR_PRD         0x2
#define         ET_CTRU_CMPA       0x4
#define         ET_CTRD_CMPA       0x5
#define         ET_CTRU_CMPB       0x6
#define         ET_CTRD_CMPB       0x7
```

Example 11-1. Constant Definitions Used in the ePWM Code Examples (continued)

```
// ETPS (Event-trigger Prescale)
// = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
// INTPRD, SOCAPRD, SOCBPRD bits
#define         ET_DISABLE          0x0
#define         ET_1ST              0x1
#define         ET_2ND              0x2
#define         ET_3RD              0x3
```

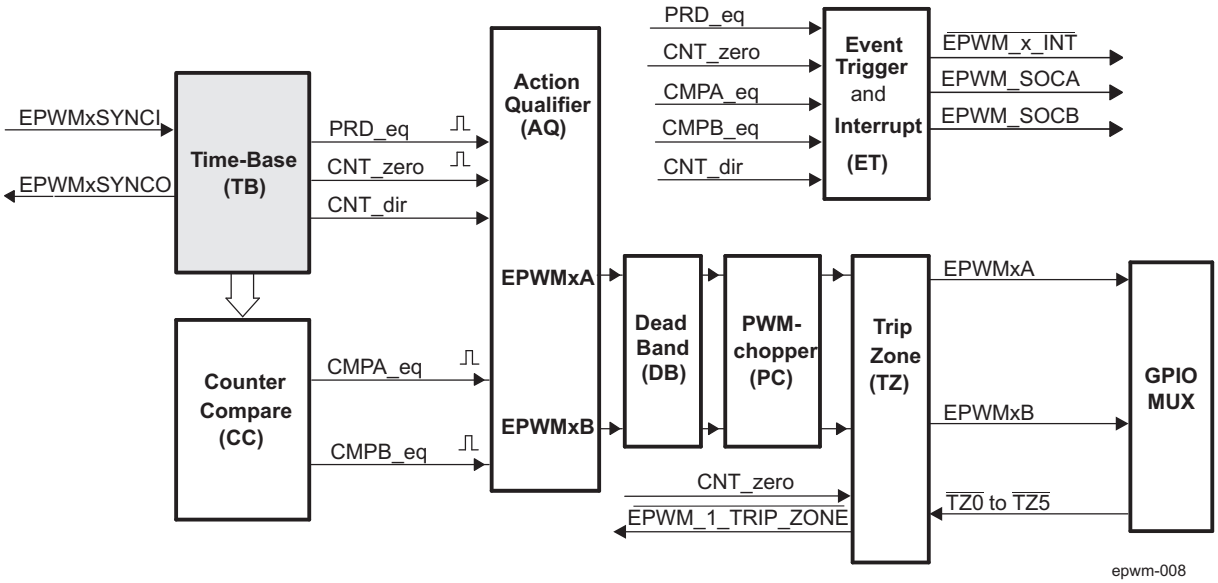
11.5.4.2 ePWM Time-Base (TB) Submodule

This section describes the Time-Base (TB) submodule in the PWM module.

11.5.4.2.1 Overview

Each ePWM module has its own time-base submodule that determines all of the timing events. Built-in synchronization logic allows the time-base of multiple ePWM modules (ePWMx) to work together as a single system. Figure 11-354 illustrates the time-base module's place within the ePWM.

Figure 11-354. ePWM Time-Base Submodule



TB module generates the period (or frequency) of the PWM output waveforms and consists of a 16-bit counter that can be configured for up, down or up and down counting. The time base for the counter is a pre-scaled version of the system clock (VBUS_CLK/n) which can be programmed for a pre-scale of 1, that is same as system clock.

TB module features:

- Specify the ePWMx time-base counter (TBCNT) frequency or period in the EPWM_TBCNT register to control how often events occur.
- Manage time-base synchronization with other ePWMx modules.
- Maintain a phase relationship with other ePWMx modules.
- Set the time-base counter to count-up, count-down, or count-up-and-down mode.
- Generate the following events:
  - PRD_eq (Time-base counter (EPWM_TBCNT register) equal to the specified period in EPWM_TBPRD register (that is TBCNT = TBPRD)).
  - CNT_zero (Time-base counter equal to zero (TBCNT = 0000h)).
- Configure the rate of the time-base clock; a prescaled version of the CPU system clock (VBUS_CLK). This allows the time-base counter to increment/decrement at a slower rate.

11.5.4.2.2 Controlling and Monitoring the ePWM Time-Base Submodule

Table 11-797 lists the registers used to control and monitor the time-base submodule.

Table 11-797. ePWM Time-Base Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
EPWM_TBCTL	Time-Base Control Register	0h	No
EPWM_TBSTS	Time-Base Status Register	2h	No
HRPWM_TBPHSHR	HRPWM extension Phase Register (1)	4h	No
EPWM_TBPHS	Time-Base Phase Register	6h	No
EPWM_TBCNT	Time-Base Counter Register	8h	No
EPWM_TBPRD	Time-Base Period Register	Ah	Yes

(1) This register is available only on ePWM instances that include the high-resolution extension (HRPWM). On ePWM modules that do not include the HRPWM, this location is reserved. See Section 11.6.3 to determine which ePWM instances include this feature.

Figure 11-355 shows the critical signals and registers of the time-base submodule. Table 11-798 provides descriptions of the key signals associated with the time-base submodule.

Figure 11-355. ePWM Time-Base Submodule Signals and Registers

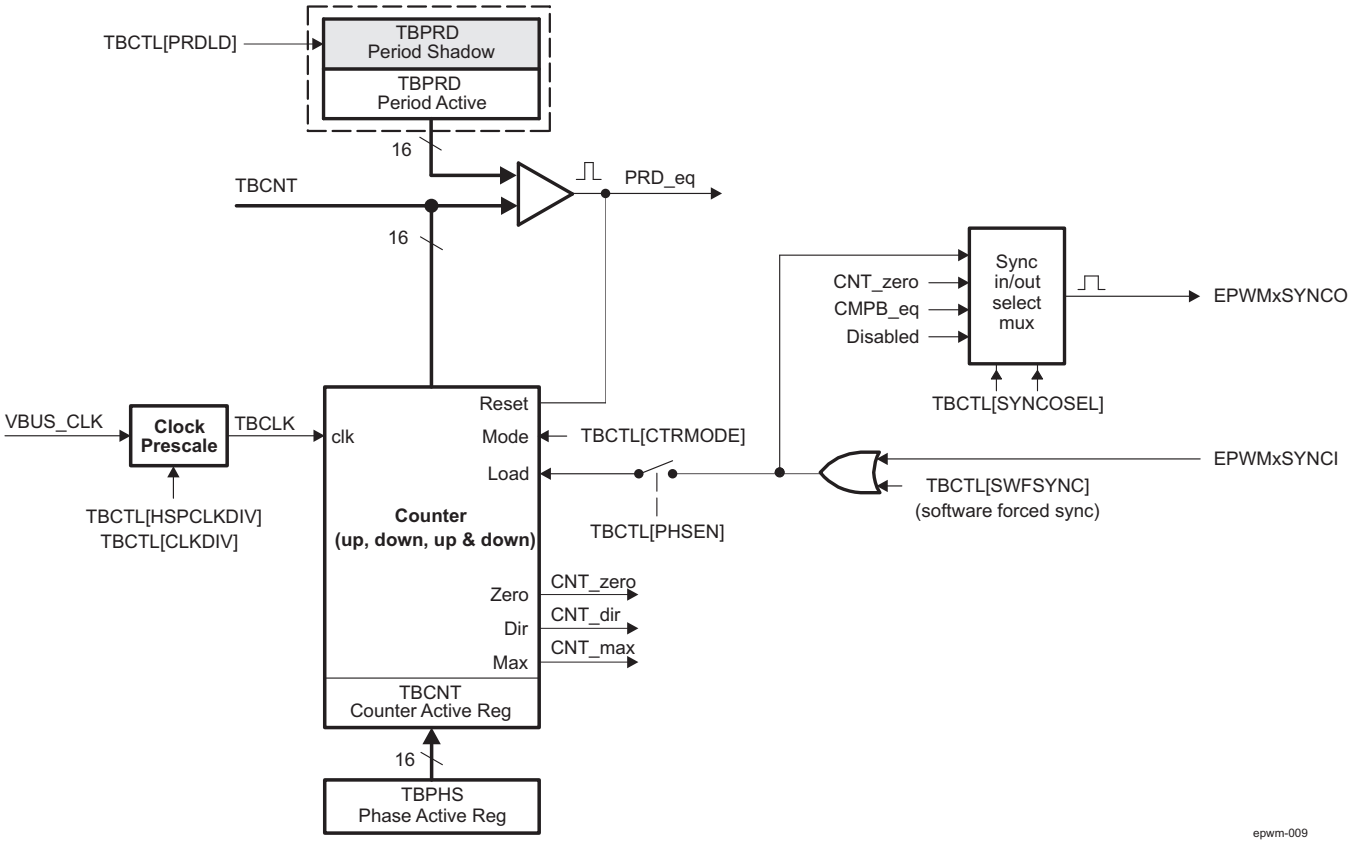


Table 11-798. ePWM Time-Base Submodule Key Signals

Signal	Description
EPWMxSYNCI	Time-base synchronization input. Input pulse used to synchronize the time-base counter with the counter of another ePWM module earlier in the synchronization chain. An ePWM peripheral can be configured to use or ignore this signal. For the first ePWM module (EPWM_0) this signal comes from a device pin. For subsequent ePWM modules this signal is passed from another ePWM peripheral. For example, EPWM2SYNCI is generated by the ePWM_1 peripheral and the EPWM3SYNCI is generated (optional through internal multiplexer) by ePWM_2 or from a device pin. See Section 11.5.4.2.3.2 for information on the synchronization order of a particular device.
EPWMxSYNCO	Time-base synchronization output. This output pulse is used to synchronize the counter of an ePWM module later in the synchronization chain. The ePWM module generates this signal from one of three event sources: 1. EPWMxSYNCI (Synchronization input pulse) 2. TBCNT = 0: The time-base counter (register EPWM_TBCNT) equal to zero (TBCNT = 0000h). 3. TBCNT = CMPB: The time-base counter (register EPWM_TBCNT) equal to the counter-compare B register — EPWM_CMPB (that is bitfield TBCNT = bitfield CMPB).
PRD_eq	Time-base counter equal to the specified period. This signal is generated whenever the counter value is equal to the active period register value. That is when TBCNT = TBPRD.
CNT_zero	Time-base counter equal to zero. This signal is generated whenever the counter value is zero. That is when TBCNT equals 0000h.
CMPB_eq	Time-base counter equal to active counter-compare B register (TBCNT = CMPB). This event is generated by the counter-compare submodule and used by the synchronization out logic.
CNT_dir	Time-base counter direction. Indicates the current direction of the ePWM's time-base counter. This signal is high when the counter is increasing and low when it is decreasing.
CNT_max	Time-base counter equal max value. (TBCNT = FFFFh) Generated event when the EPWM_TBCNT value reaches its maximum value. This signal is only used only as a status bit.
TBCLK	Time-base clock. This is a prescaled version of the system clock — VBUS_CLK and is used by all submodules within the ePWMn. This clock determines the rate at which time-base counter increments or decrements.

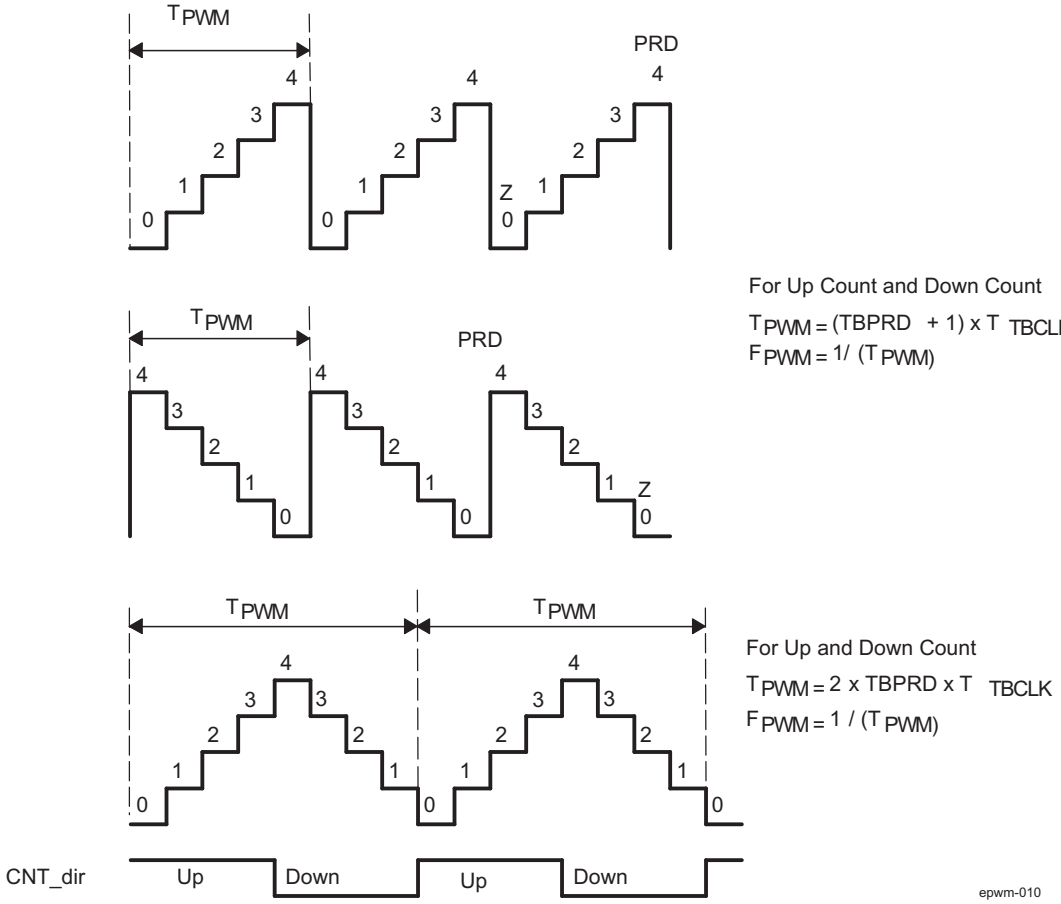
11.5.4.2.3 Calculating PWM Period and Frequency

The frequency of PWM events is controlled by the time-base period (EPWM_TBPRD) register and the mode of the time-base counter. Figure 11-356 shows the period (T_{pwm}) and frequency (F_{pwm}) relationships for the up-count, down-count, and up-down-count time-base counter modes when the period is set to 4 (EPWM_TBPRD register bitfield TBPRD = 0x4). The time increment for each step is defined by the time-base clock (TBCLK) which is a prescaled version of the system clock (VBUS_CLK).

The time-base counter has three modes of operation selected by the time-base control register (EPWM_TBCTL):

- **Up-Down-Count Mode:** In up-down-count mode, the time-base counter starts from zero and increments until the period (TBPRD) value is reached. When the period value is reached, the time-base counter then decrements until it reaches zero. At this point the counter repeats the pattern and begins to increment.
- **Up-Count Mode:** In this mode, the time-base counter starts from zero and increments until it reaches the value in the period register (TBPRD). When the period value is reached, the time-base counter resets to zero and begins to increment once again.
- **Down-Count Mode:** In down-count mode, the time-base counter starts from the period (TBPRD) value and decrements until it reaches zero. When it reaches zero, the time-base counter is reset back to the period value and it repeats this pattern.

Figure 11-356. ePWM Time-Base Frequency and Period



11.5.4.2.3.1 ePWM Time-Base Period Shadow Register

The time-base period register (EPWM_TBPRD) has a shadow register. Shadowing allows the register update to be synchronized with the hardware. The following definitions are used to describe all shadow registers in the ePWM module:

- **Active Register:** The active register controls the hardware and is responsible for actions that the hardware causes or invokes.
- **Shadow Register:** The shadow register buffers or provides a temporary holding location for the active register. It has no direct effect on any control hardware. At a strategic point in time the shadow register's content is transferred to the active register. This prevents corruption or spurious operation due to the register being asynchronously modified by software.

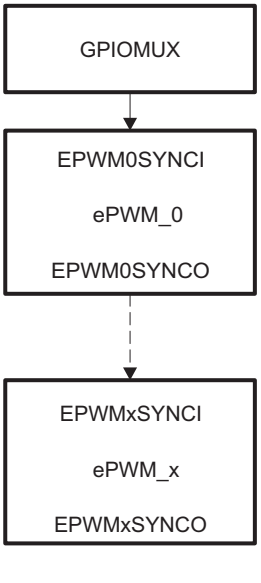
The memory address of the shadow period register is the same as the active register. Which register is written to or read from is determined by the EPWM_TBCTL[3] PRDLD bit. This bit enables and disables the EPWM_TBPRD shadow register as follows:

- **Time-Base Period Shadow Mode:** The EPWM_TBPRD shadow register is enabled when EPWM_TBCTL[3] PRDLD = 0. Reads from and writes to the EPWM_TBPRD memory address go to the shadow register. The shadow register contents are transferred to the active register (EPWM_TBPRD (Active) ← EPWM_TBPRD (shadow)) when the time-base counter (register (EPWM_TBCNT) equals zero (TBCNT = 0000h). By default the EPWM_TBPRD shadow register is enabled.
- **Time-Base Period Immediate Load Mode:** If immediate load mode is selected (EPWM_TBCTL[3] PRDLD = 1), then a read from or a write to the TBPRD memory address goes directly to the active register.

11.5.4.2.3.2 ePWM Time-Base Counter Synchronization

A time-base synchronization scheme connects all of the ePWM modules on a device. Each ePWM module has a synchronization input (EPWMxSYNCI) and a synchronization output (EPWMxSYNCO). The input synchronization for the first instance (ePWM_0) comes from an external pin. For the device ePWM environment sync pin details refer to the Section 11.5.2. The possible synchronization connections for the remaining ePWM modules is shown in Figure 11-357.

Figure 11-357. ePWM Time-Base Counter Synchronization Scheme 1



Each ePWM module can be configured to use or ignore the synchronization input. If the EPWM_TBCTL[2] PHSEN bit is set, then the time-base counter (TBCNT) of the ePWM module (register EPWM_TBCNT) will be automatically loaded with the phase register (EPWM_TBPHS) contents when one of the following conditions occur:

- **EPWMxSYNCI: Synchronization Input Pulse:** The value of the phase register is loaded into the counter register when an input synchronization pulse is detected (EPWM_TBPHS → EPWM_TBCNT). This operation occurs on the next valid time-base clock (TBCLK) edge.
- **Software Forced Synchronization Pulse:** Writing a 1 to the EPWM_TBCTL[6] SWFSYNC control bit invokes a software forced synchronization. This pulse is ORed with the synchronization input signal, and therefore has the same effect as a pulse on EPWMxSYNCI.

This feature enables the ePWM module to be automatically synchronized to the time base of another ePWM module. Lead or lag phase control can be added to the waveforms generated by different ePWM modules to synchronize them. In up-down-count mode, the EPWM_TBCTL[13] PHSDIR bit configures the direction of the time-base counter immediately after a synchronization event. The new direction is independent of the direction prior to the synchronization event. The TBPHS bit is ignored in count-up or count-down modes. See Figure 11-358 through Figure 11-361 for examples.

Clearing the EPWM_TBCTL[2] PHSEN bit configures the ePWM to ignore the synchronization input pulse. The synchronization pulse can still be allowed to flow-through to the EPWMxSYNCO and be used to synchronize other ePWM modules. In this way, a master time-base (for example, ePWM_1) and downstream modules (ePWM_2–ePWM_x) can be set and they may elect to run in synchronization with the master.

11.5.4.2.4 Phase Locking the Time-Base Clocks of Multiple ePWM Modules

As already described in the Section 11.5.3.4, EPWM0_TBCLKEN through EPWM5_TBCLKEN bits in the EPWM_CTL register of the device BOOT_CFG can be used to individually control or globally synchronize the time-base clocks of all enabled ePWM modules on a device. When all EPWMn_TBCLKEN (where n = 0 to 5) bits are set to 0b0, the time-base clocks of all ePWM_x (where x = 0 to 5) modules are stopped (default). When all EPWMn_TBCLKEN bits are simultaneously set in SW to 0b1, all ePWM_x modules time-base clocks are started with the rising edge of TBCLK aligned. For perfectly synchronized TBCLKs, the prescaler bits in the EPWM_TBCTL register of each ePWM module must be set identically. The proper procedure for enabling the ePWM clocks is as follows:

1. Enable the ePWM module clocks.
2. Set EPWMn_TBCLKEN = 0. This will stop the time-base clock within any enabled ePWM_x module.
3. Configure the prescaler values and desired ePWM modes per each involved ePWM_x.
4. Simultaneously set bits EPWMn_TBCLKEN to 0b1 (where n = 0 to 5) 1.

11.5.4.2.5 ePWM Time-Base Counter Modes and Timing Waveforms

The time-base counter operates in one of four modes:

- Up-count mode which is asymmetrical.
- Down-count mode which is asymmetrical.
- Up-down-count which is symmetrical.
- Frozen where the time-base counter is held constant at the current value.

To illustrate the operation of the first three modes, Figure 11-358 to Figure 11-361 show when events are generated and how the time-base responds to an EPWMxSYNCI signal.

Figure 11-358. ePWM Time-Base Up-Count Mode Waveforms

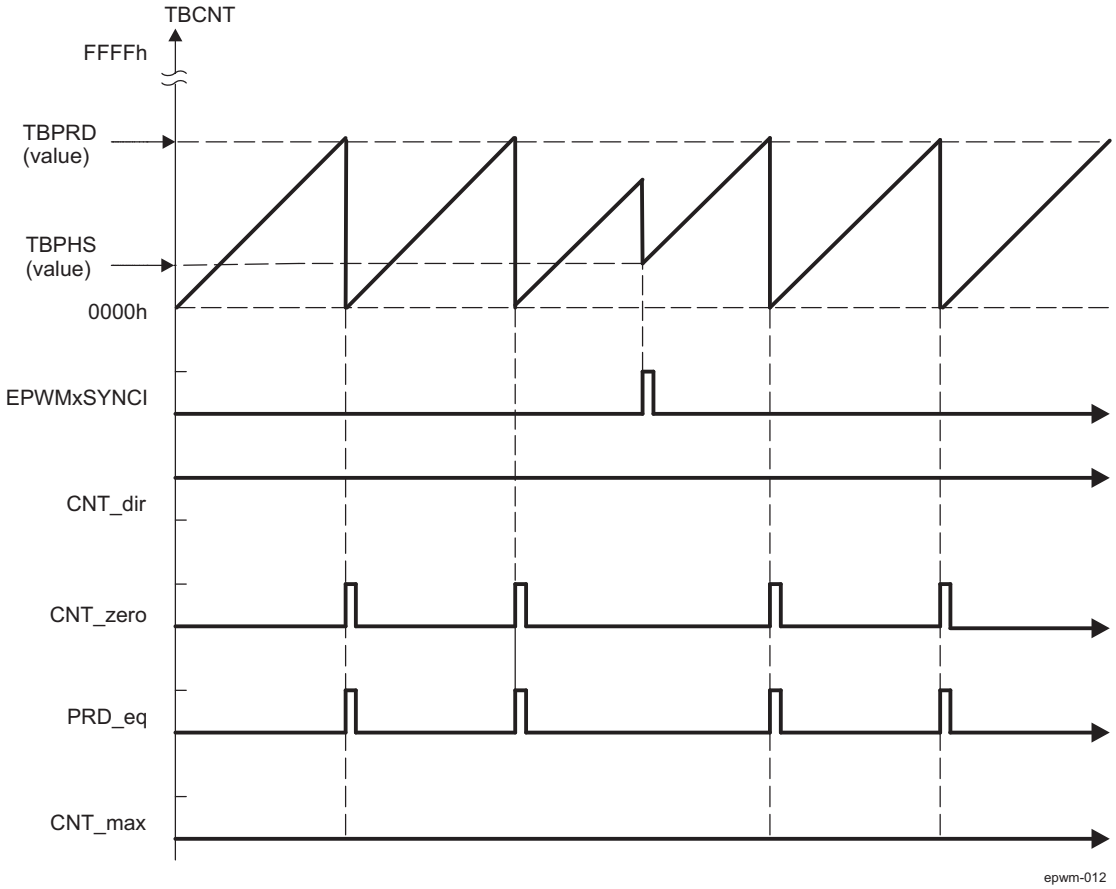


Figure 11-359. ePWM Time-Base Down-Count Mode Waveforms

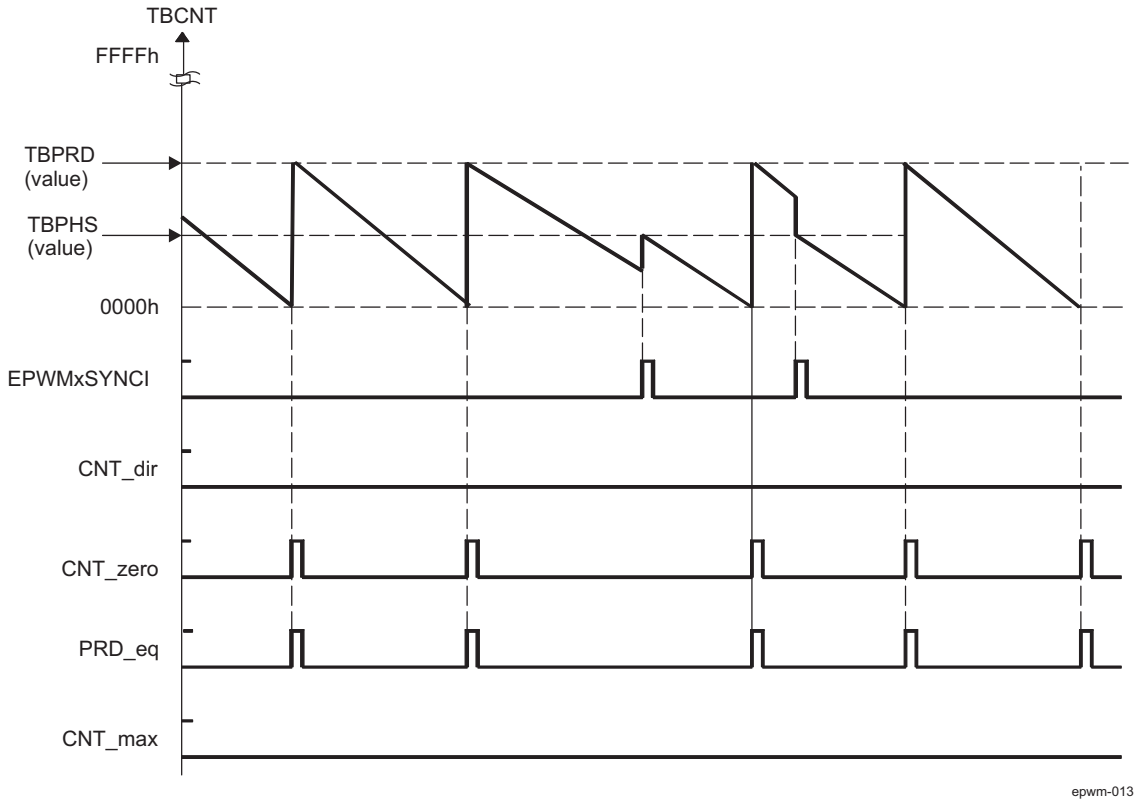


Figure 11-360. ePWM Time-Base Up-Down-Count Waveforms, EPWM_TBCTL[13] PHSDIR = 0 Count Down on Synchronization Event

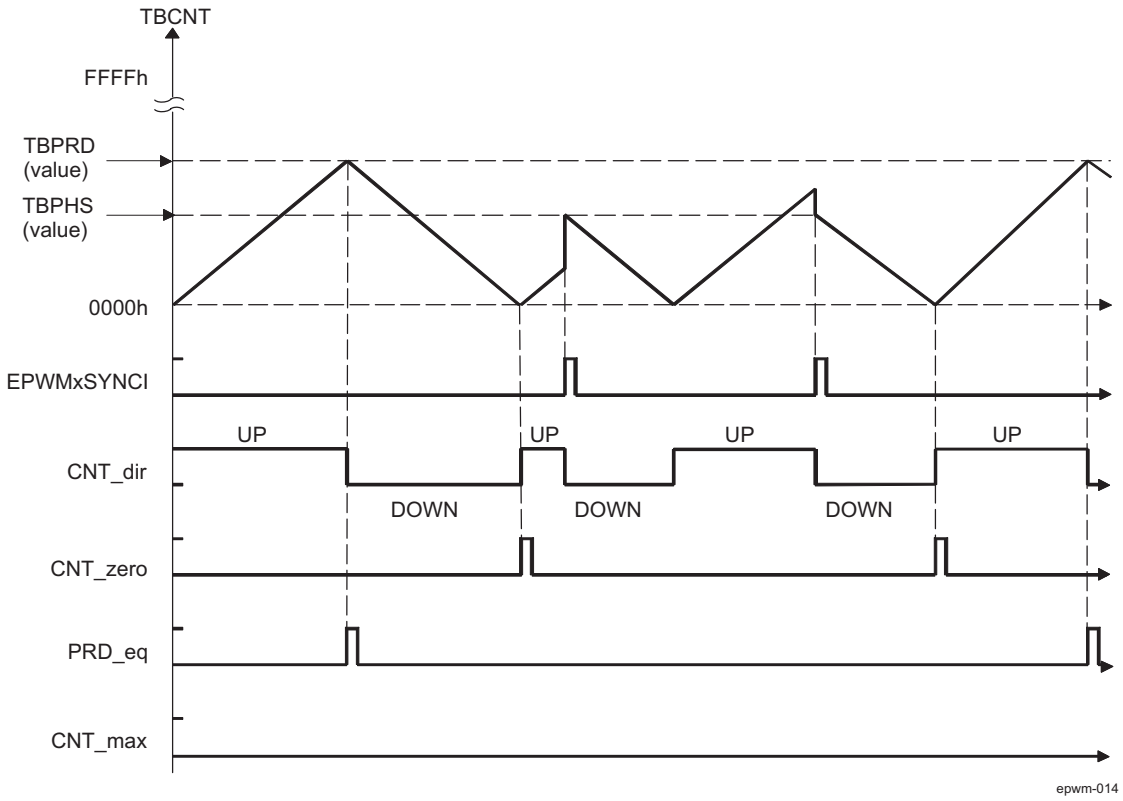
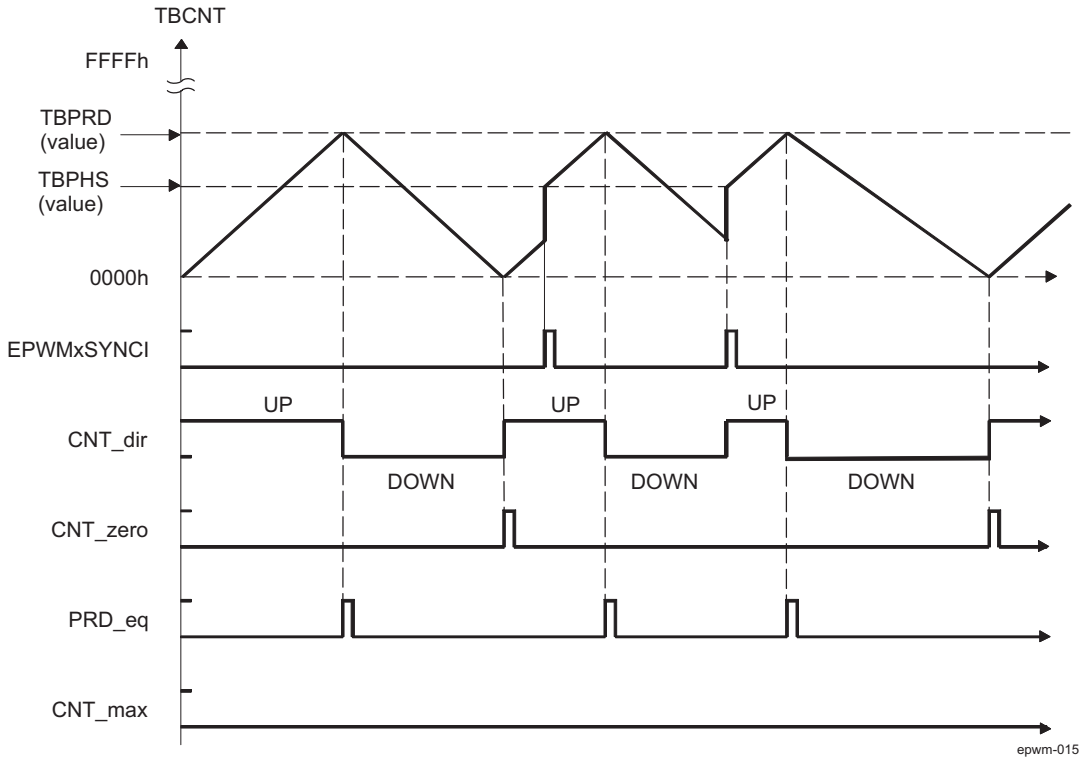


Figure 11-361. ePWM Time-Base Up-Down Count Waveforms, EPWM_TBCTL[13] PHSDIR = 1 Count Up on Synchronization Event



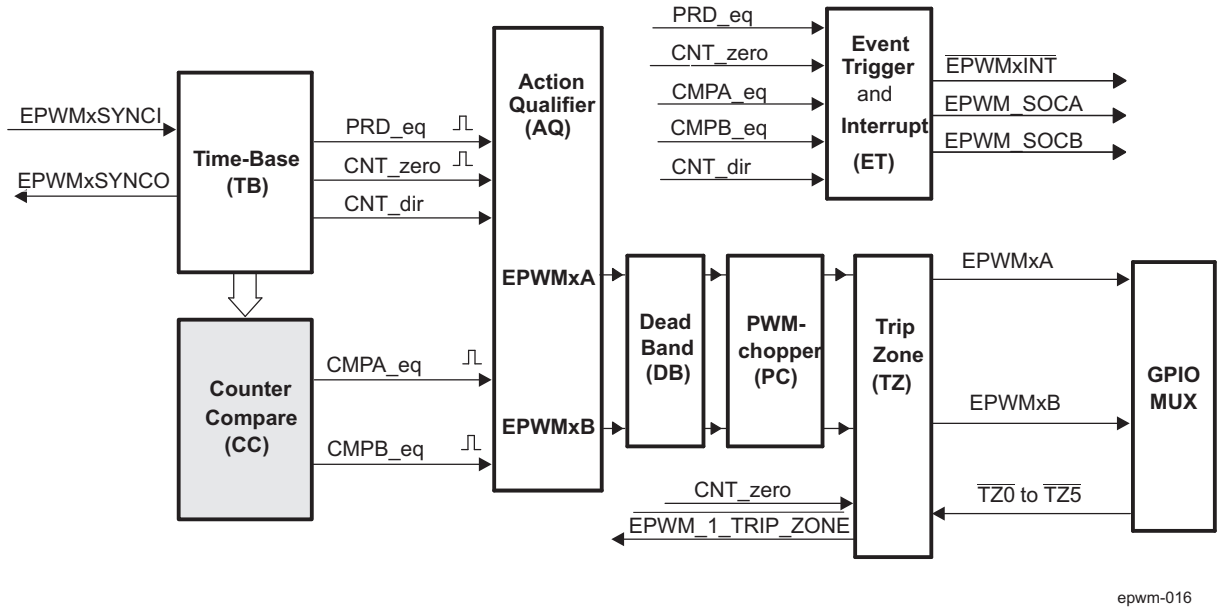
11.5.4.3 ePWM Counter-Compare (CC) Submodule

This section describes the Counter-Compare (CC) submodule in the PWM module.

11.5.4.3.1 Overview

Figure 11-362 illustrates the counter-compare submodule within the ePWM. Figure 11-363 shows the basic structure of the counter-compare submodule.

Figure 11-362. ePWM Counter-Compare Submodule



CC module features:

- Generates events based on programmable time stamps using the EPWM_CMPA and EPWM_CMPB registers
  - CMPA_eq (Time-base counter equals counter-compare A register (TBCNT = CMPA)).
  - CMPB_eq (Time-base counter equals counter-compare B register (TBCNT = CMPB)).
- Controls the PWM duty cycle if the action-qualifier submodule is configured appropriately
- Shadows new compare values to prevent corruption or glitches during the active PWM cycle.

The counter-compare submodule takes as input the time-base counter value. This value is continuously compared to the counter-compare A (EPWM_CMPA) and counter-compare B (EPWM_CMPB) registers. When the time-base counter is equal to one of the compare registers, the counter-compare unit generates an appropriate event.

11.5.4.3.2 Controlling and Monitoring the ePWM Counter-Compare Submodule

Table 11-799 lists the registers used to control and monitor the counter-compare submodule. Table 11-800 lists the key signals associated with the counter-compare submodule.

Table 11-799. ePWM Counter-Compare Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
EPWM_CMPCTL	Counter-Compare Control Register.	Eh	No
HRPWM_CMPAHR	HRPWM Counter-Compare A Extension Register (1)	10h	Yes
EPWM_CMPA	Counter-Compare A Register	12h	Yes
EPWM_CMPB	Counter-Compare B Register	14h	Yes

(1) This register is available only on ePWM modules with the high-resolution extension (HRPWM). On ePWM modules that do not include the HRPWM, this location is reserved. See Section 11.6.3 to determine which ePWM instances include this feature.

Figure 11-363. ePWM Counter-Compare Submodule Signals and Registers

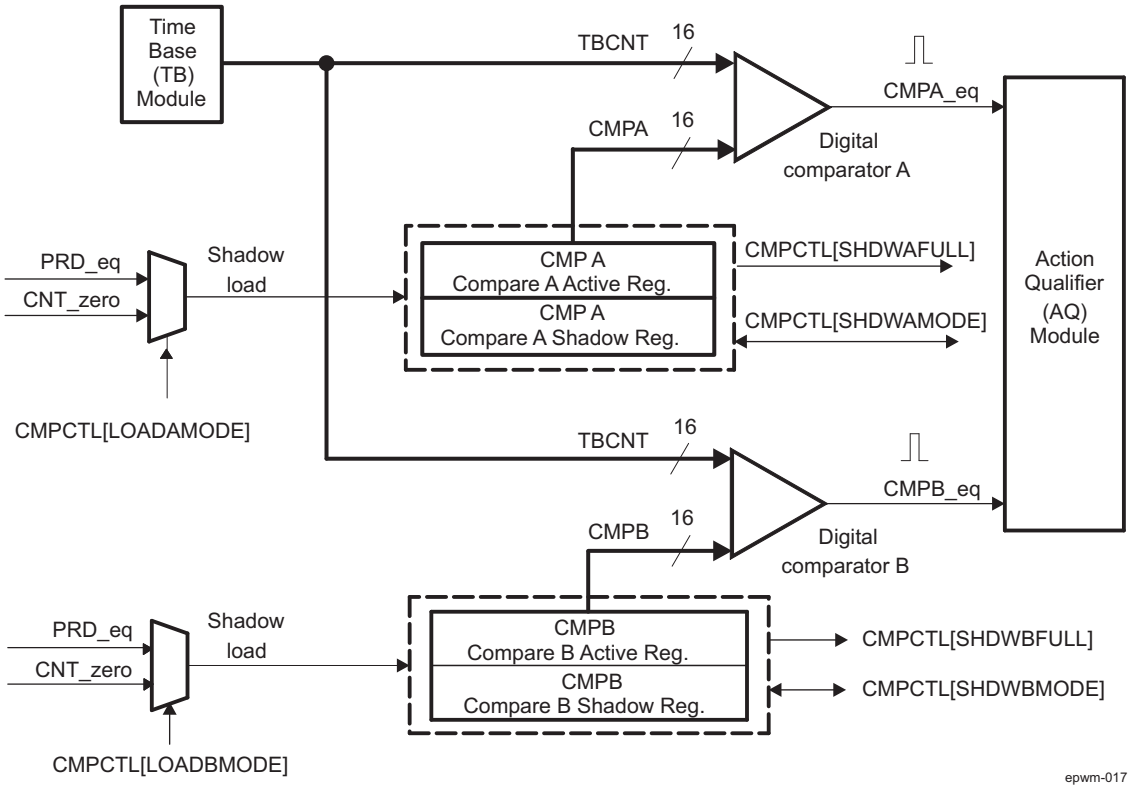


Table 11-800. ePWM Counter-Compare Submodule Key Signals

Signal	Description of Event	Register Bitfields Compared
CMPA_eq	Time-base counter equal to the active counter-compare A value	TBCNT = CMPA
CMPB_eq	Time-base counter equal to the active counter-compare B value	TBCNT = CMPB
PRD_eq	Time-base counter equal to the active period. Used to load active counter-compare A and B registers from the shadow register	TBCNT = TBPRD
CNT_zero	Time-base counter equal to zero. Used to load active counter-compare A and B registers from the shadow register	TBCNT = 0000h

11.5.4.3.3 Operational Highlights for the ePWM Counter-Compare Submodule

The counter-compare submodule is responsible for generating two independent compare events based on two compare registers:

1. CMPA: Time-base counter equal to counter-compare A register (EPWM_TBCNT = EPWM_CMPA).
2. CMPB: Time-base counter equal to counter-compare B register (EPWM_TBCNT = EPWM_CMPB).

For up-count or down-count mode, each event occurs only once per cycle. For up-down-count mode each event occurs twice per cycle, if the compare value is between 0000h and TBPRD; and occurs once per cycle, if the compare value is equal to 0000h or equal to TBPRD. These events are fed into the action-qualifier submodule where they are qualified by the counter direction and converted into actions if enabled. Refer to for more details.

The counter-compare registers EPWM_CMPA and EPWM_CMPB each have an associated shadow register. Shadowing provides a way to keep updates to the registers synchronized with the hardware. When shadowing is used, updates to the active registers only occurs at strategic points. This prevents corruption or spurious operation due to the register being asynchronously modified by software. The memory address of the active register and the shadow register is identical. Which register is written to or read from is determined by the EPWM_CMPCTL[4] SHDWAMODE and EPWM_CMPCTL[6] SHDWBMODE bits. These bits enable and disable the EPWM_CMPA shadow register and EPWM_CMPB shadow register respectively. The behavior of the two load modes is described below:

- **Shadow Load Mode:**
  The shadow mode for the EPWM_CMPA is enabled by clearing the EPWM_CMPCTL[4] SHDWAMODE bit and the shadow register for CMPB is enabled by clearing the EPWM_CMPCTL[6] SHDWBMODE bit. Shadow mode is enabled by default for both EPWM_CMPA and EPWM_CMPB.

  If the shadow register is enabled then the content of the shadow register is transferred to the active register on one of the following events:

  – PRD_eq: Time-base counter equal to the period (TBCNT = TBPRD).
  – CNT_zero: Time-base counter equal to zero (TBCNT = 0000h)
  – Both PRD_eq and CNT_zero events occurrence

  Which of these three events will drive the counter compare module is specified by the EPWM_CMPCTL[1-0] LOADAMODE and EPWM_CMPCTL[3-2] LOADBMODE register bits. Only the active register contents are used by the counter-compare submodule to generate events to be sent to the action-qualifier.
- **Immediate Load Mode:**
  If immediate load mode is selected (EPWM_CMPCTL[4] SHDWAMODE = 1 or EPWM_CMPCTL[6] SHDWBMODE = 1), then a read from or a write to the register will go directly to the active register.

Figure 11-364 and Figure 11-365 show Compare A and B Dual Shadow registers.

Figure 11-364. Compare A Dual Shadow register

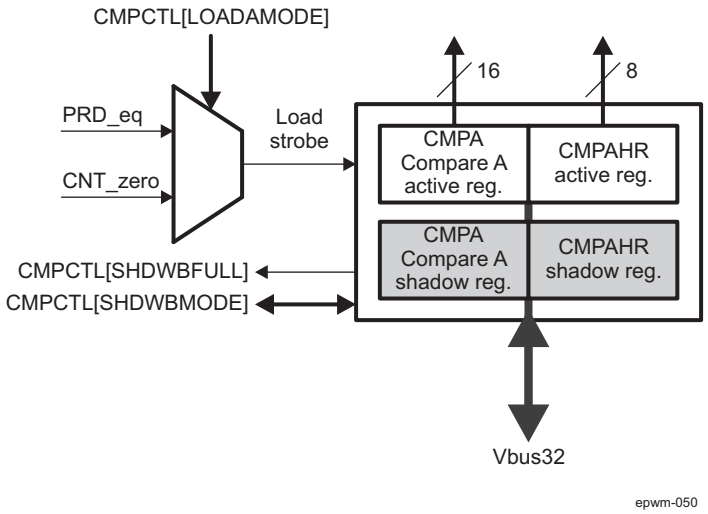
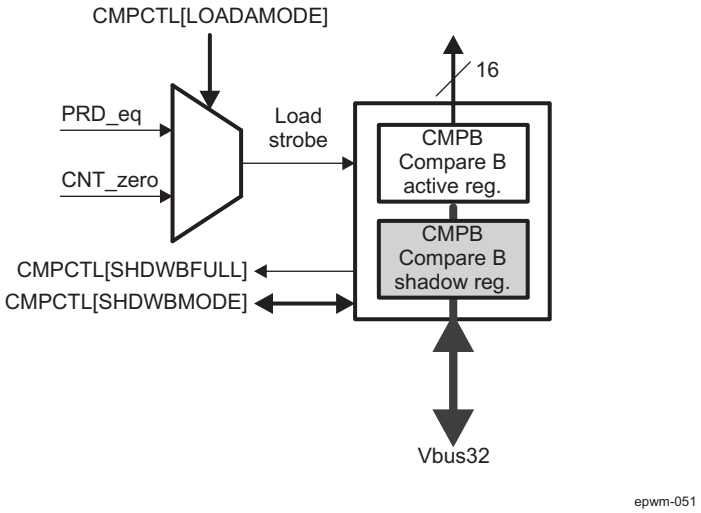


Figure 11-365. Compare B Dual Shadow register



NOTE: In HRPWM mode the user must perform a single 32-bit write access to both the HRPWM_CMPAHR and EPWM_CMPA registers. Two 16-bit accesses are not allowed.

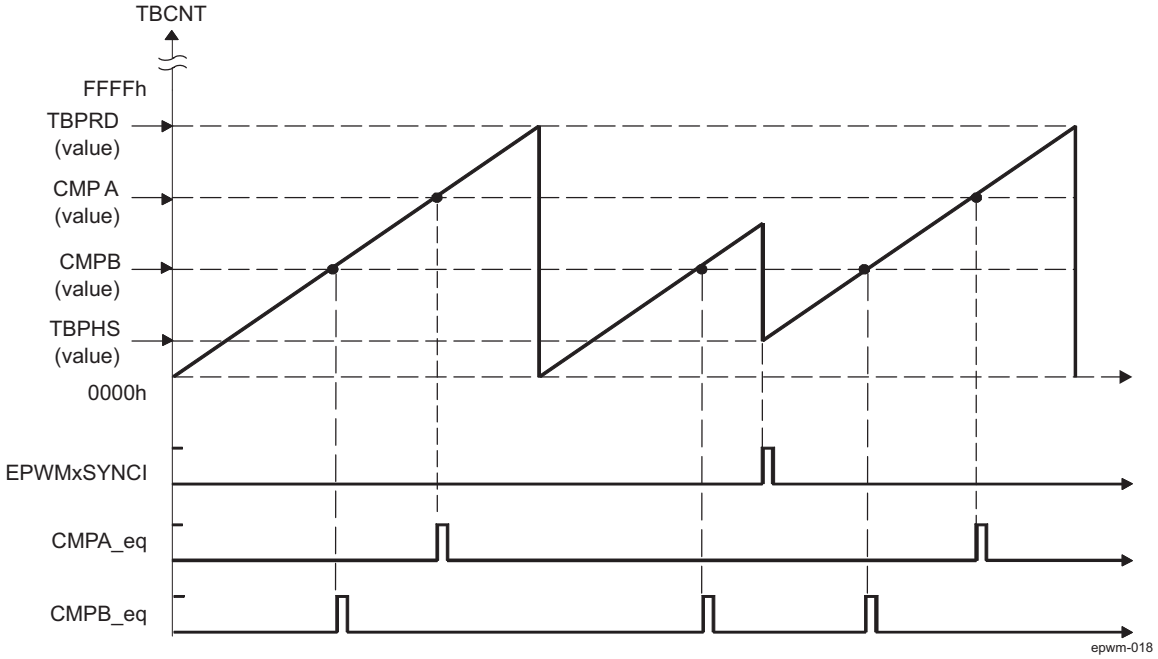
11.5.4.3.4 ePWM Counter-Compare Submodule Timing Waveforms

As described in Section 11.5.4.2, the Time Base (TB) module can be configured to operate in 3 distinct count modes:

- Up-count mode: used to generate an asymmetrical PWM waveform.
- Down-count mode: used to generate an asymmetrical PWM waveform.
- Up-down-count mode: used to generate a symmetrical PWM waveform.

The timing diagrams in Figure 11-366 to Figure 11-369 show how CMPA and CMPB events are generated in each of the 3 count modes and how the EPWMxSYNCI signal interacts.

Figure 11-366. ePWM Counter-Compare Event Waveforms in Up-Count Mode



NOTE: An EPWMxSYNCI external synchronization event can cause a discontinuity in the TBCNT count sequence. This can lead to a compare event being skipped. This skipping is considered normal operation and must be taken into account.

Figure 11-367. ePWM Counter-Compare Events in Down-Count Mode

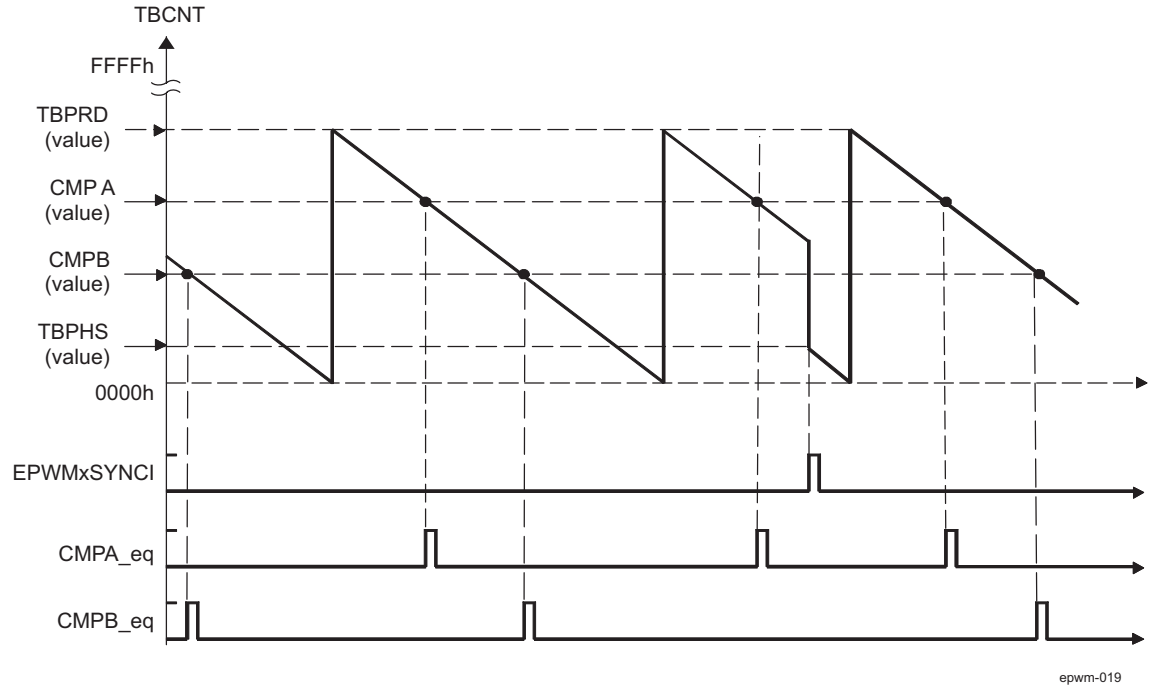


Figure 11-368. ePWM Counter-Compare Events in Up-Down-Count Mode, EPWM_TBCTL[13] PHSDIR = 0 Count Down on Synchronization Event

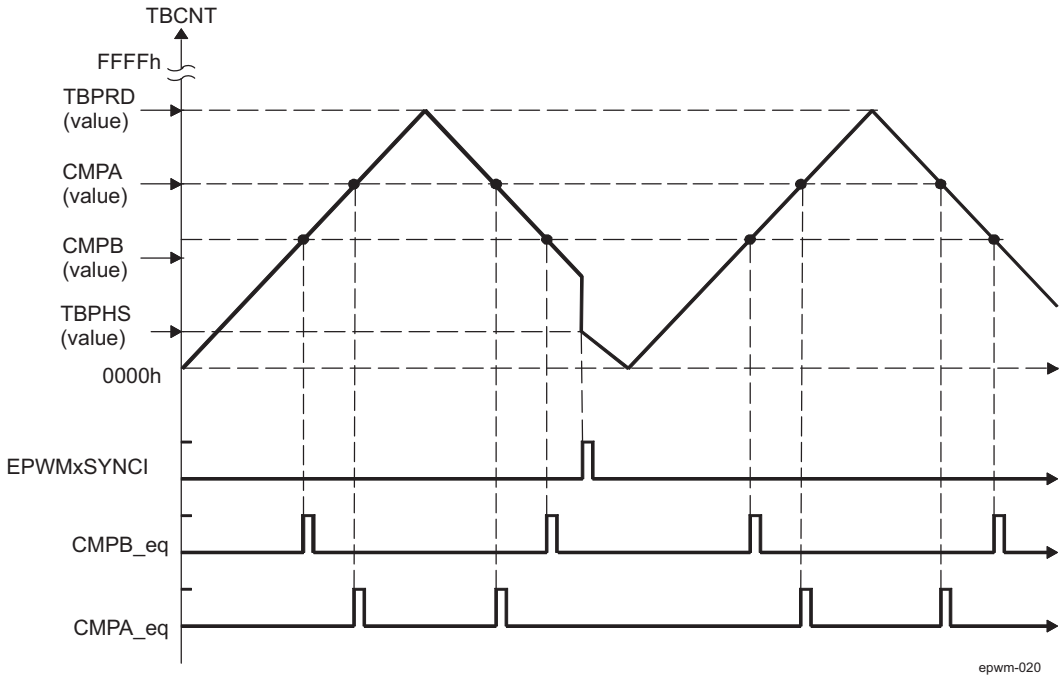
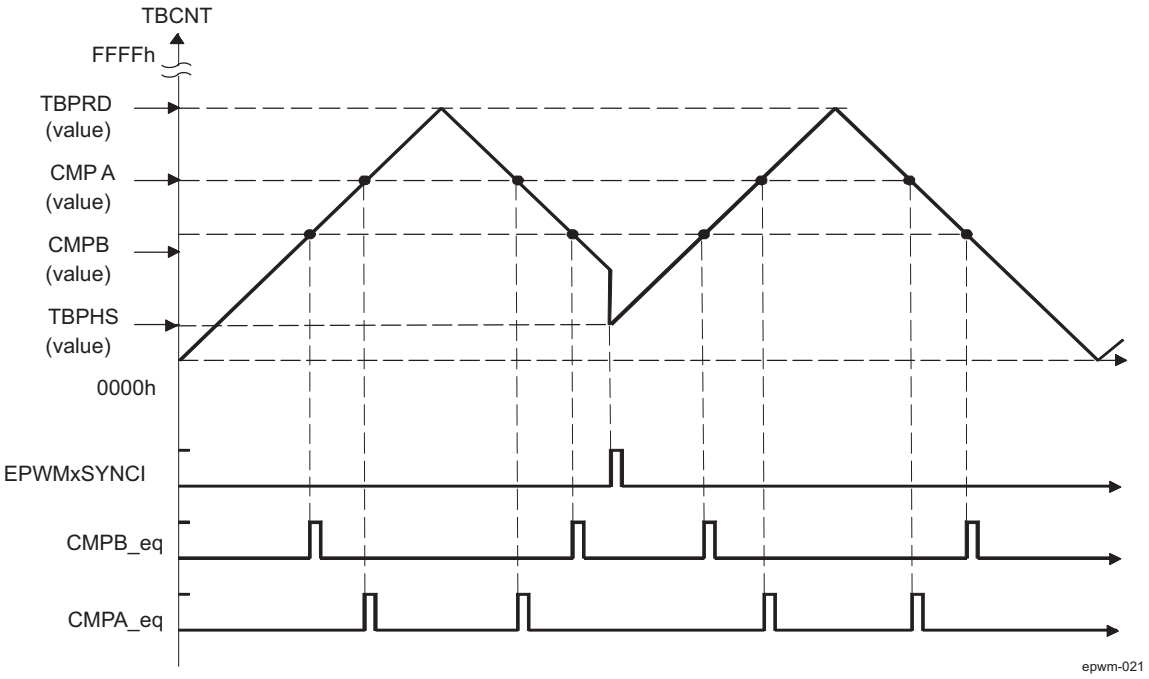


Figure 11-369. ePWM Counter-Compare Events in Up-Down-Count Mode, EPWM_TBCTL[13] PHSDIR = 1 Count Up on Synchronization Event



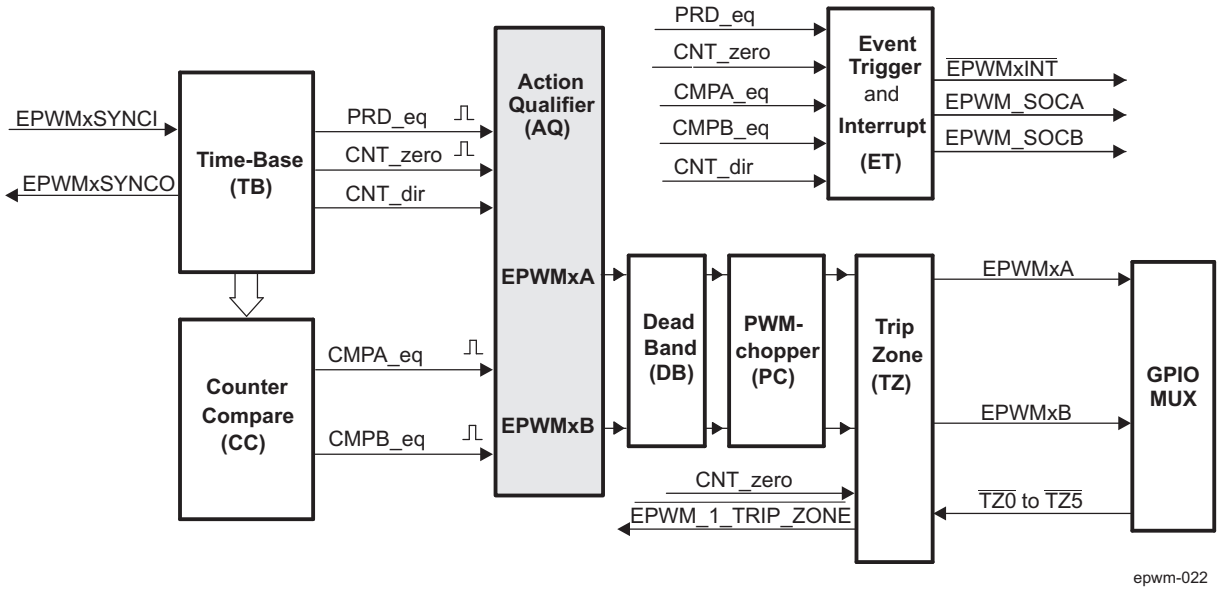
11.5.4.4 ePWM Action-Qualifier (AQ) Submodule

This section describes the Action-Qualifier (AQ) submodule in the PWM module.

11.5.4.4.1 Overview

Figure 11-370 shows the action-qualifier (AQ) submodule in the ePWM system. This submodule has the most important role in waveform construction and PWM generation. It decides which events are converted into various action types, thereby producing the required switched waveforms at the EPWMxA and EPWMxB outputs.

Figure 11-370. ePWM Action-Qualifier Submodule



AQ module features:

- Qualifying and generating actions (set, clear, toggle) based on the following events:
  - PRD_eq: Time-base counter equal to the period (TBCNT = TBPRD)
  - CNT_zero: Time-base counter equal to zero (TBCNT = 0000h)
  - CMPA_eq: Time-base counter equal to the counter-compare A register (TBCNT = CMPA)
  - CMPB_eq: Time-base counter equal to the counter-compare B register (TBCNT = CMPB)
- Managing priority when these events occur concurrently
- Providing independent control of events when the time-base counter is increasing and when it is decreasing.

11.5.4.4.2 Controlling and Monitoring the ePWM Action-Qualifier Submodule

Table 11-801 lists the registers used to control and monitor the action-qualifier submodule.

Table 11-801. ePWM Action-Qualifier Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
EPWM_AQCTLA	Action-Qualifier Control Register For Output A (EPWMxA)	16h	No
EPWM_AQCTLB	Action-Qualifier Control Register For Output B (EPWMxB)	18h	No
EPWM_AQSFRC	Action-Qualifier Software Force Register	1Ah	No
EPWM_AQCSFRC	Action-Qualifier Continuous Software Force	1Ch	Yes

The action-qualifier submodule is based on event-driven logic. It can be thought of as a programmable cross switch with events at the input and actions at the output, all of which are software controlled via the set of registers as shown in Figure 11-371. The possible input events are summarized again in Table 11-802.

Figure 11-371. ePWM Action-Qualifier Submodule Inputs and Outputs

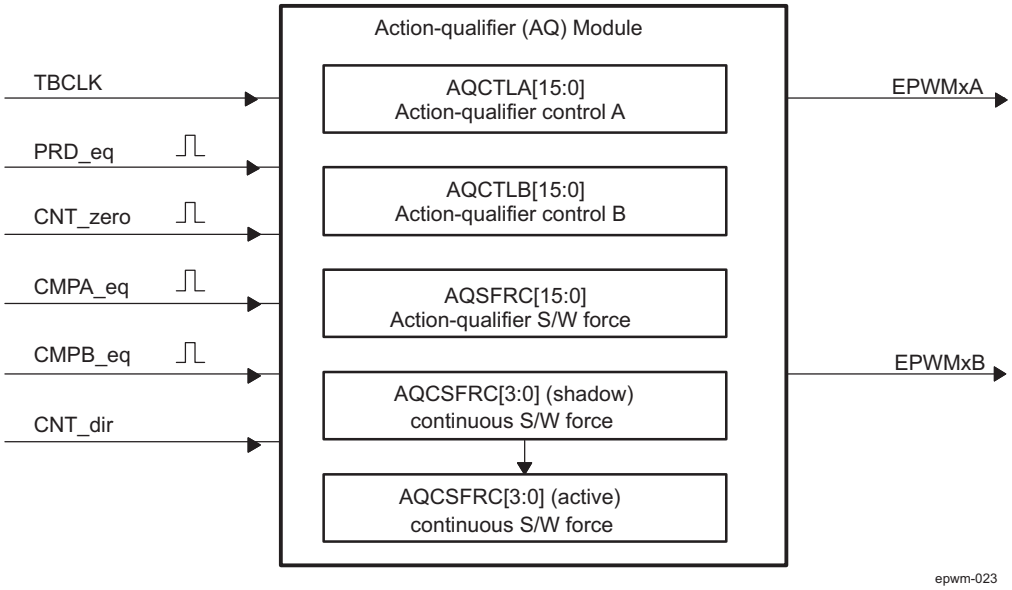


Table 11-802. ePWM Action-Qualifier Submodule Possible Input Events

Signal	Description	Register Bitfield Compared
PRD_eq	Time-base counter equal to the period value	TBCNT = TBPRD
CNT_zero	Time-base counter equal to zero	TBCNT = 0000h
CMPA_eq	Time-base counter equal to the counter-compare A	TBCNT = CMPA
CMPB_eq	Time-base counter equal to the counter-compare B	TBCNT = CMPB
Software forced event	Asynchronous event initiated by software	

The software forced action is a useful asynchronous event. This control is handled by registers EPWM_AQSFRC and EPWM_AQCSFRC.

The action-qualifier submodule controls how the two outputs EPWMxA and EPWMxB behave when a particular event occurs. The event inputs to the action-qualifier submodule are further qualified by the counter direction (up or down). This allows for independent action on outputs on both the count-up and count-down phases.

The possible actions imposed on outputs EPWMxA and EPWMxB are:

- **Set High:** Set output EPWMxA or EPWMxB to a high level.
- **Clear Low:** Set output EPWMxA or EPWMxB to a low level.
- **Toggle:** If EPWMxA or EPWMxB is currently pulled high, then pull the output low. If EPWMxA or EPWMxB is currently pulled low, then pull the output high.
- **Do Nothing:** Keep outputs EPWMxA and EPWMxB at same level as currently set. Although the "Do Nothing" option prevents an event from causing an action on the EPWMxA and EPWMxB outputs, this event can still trigger interrupts. See the Event_Trigger (ET) submodule description in Section 11.5.4.8 for details.

Actions are specified independently for either output (EPWMxA or EPWMxB). Any or all events can be configured to generate actions on a given output. All qualifier actions are configured via the control registers found at the end of this section.

For clarity, the drawings in this chapter use a set of symbolic actions. These symbols are summarized in Figure 11-372. Each symbol represents an action as a marker in time. Some actions are fixed in time (zero and period) while the CMPA and CMPB actions are moveable and their time positions are programmed via the counter-compare A and B registers, respectively. To turn off or disable an action, use the "Do Nothing option"; it is the default at reset.

Figure 11-372. Possible Action-Qualifier Actions for EPWMxA and EPWMxB Outputs

S/W force	TB Counter equals:				Actions
	Zero	Comp A	Comp B	Period	
SW ×	Z ×	CA ×	CB ×	P ×	Do Nothing
SW ↓	Z ↓	CA ↓	CB ↓	P ↓	Clear Low
SW ↑	Z ↑	CA ↑	CB ↑	P ↑	Set High
SW T	Z T	CA T	CB T	P T	Toggle

epwm-024

11.5.4.4.3 ePWM Action-Qualifier Event Priority

It is possible for the ePWM action qualifier to receive more than one event at the same time. In this case events are assigned a priority by the hardware. The general rule is: events occurring later in time have a higher priority and software forced events always have the highest priority. The event priority levels for up-down-count mode are shown in Table 11-803. A priority level 1 is the highest priority and level 7 is the lowest. The priority changes slightly depending on the direction of TBCNT.

Table 11-803. ePWM Action-Qualifier Event Priority for Up-Down-Count Mode

Priority Level	Event if TBCNT is Incrementing TBCNT = 0 up to TBCNT = TBPRD	Event if TBCNT is Decrementing TBCNT = TBPRD down to TBCNT = 1
1 (Highest)	Software forced event	Software forced event
2	Counter equals CMPB on up-count (CBU)	Counter equals CMPB on down-count (CBD)
3	Counter equals CMPA on up-count (CAU)	Counter equals CMPA on down-count (CAD)
4	Counter equals zero	Counter equals period (TBPRD in EPWM_TBPRD active register)
5	Counter equals CMPB on down-count (CBD) (1)	Counter equals CMPB on up-count (CBU) (1)
6 (Lowest)	Counter equals CMPA on down-count (CAD) (1)	Counter equals CMPA on up-count (CBU) (1)

(1) To maintain symmetry for up-down-count mode, both up-events (CAU/CBU) and down-events (CAD/CBD) can be generated for TBPRD. Otherwise, up-events can occur only when the counter is incrementing and down-events can occur only when the counter is decrementing.

Table 11-804 shows the action-qualifier priority for up-count mode. In this case, the counter direction is always defined as up and thus down-count events will never be taken.

Table 11-804. ePWM Action-Qualifier Event Priority for Up-Count Mode

Priority Level	Event
1 (Highest)	Software forced event
2	Counter equal to period (TBPRD)
3	Counter equal to CMPB on up-count (CBU)
4	Counter equal to CMPA on up-count (CAU)
5 (Lowest)	Counter equal to Zero

Table 11-805 shows the action-qualifier priority for down-count mode. In this case, the counter direction is always defined as down and thus up-count events will never be taken.

Table 11-805. ePWM Action-Qualifier Event Priority for Down-Count Mode

Priority Level	Event
1 (Highest)	Software forced event
2	Counter equal to Zero
3	Counter equal to CMPB on down-count (CBD)
4	Counter equal to CMPA on down-count (CAD)
5 (Lowest)	Counter equal to period (TBPRD)

It is possible to set the compare value greater than the period. In this case the action will take place as shown in Table 11-806.

Table 11-806. Behavior if CMPA/CMPB is Greater than the Period

Counter Mode	Compare on Up-Count Event CAU/CBU	Compare on Down-Count Event CAU/CBU
Up-Count Mode	If CMPA/CMPB ≤ TBPRD period, then the event occurs on a compare match (TBCNT = CMPA or CMPB). If CMPA/CMPB > TBPRD, then the event will not occur.	Never occurs.
Down-Count Mode	Never occurs.	If CMPA/CMPB < TBPRD, the event will occur on a compare match (TBCNT = CMPA or CMPB). If CMPA/CMPB ≥ TBPRD, the event will occur on a period match (TBCNT = TBPRD).
Up-Down-Count Mode	If CMPA/CMPB < TBPRD and the counter is incrementing, the event occurs on a compare match (TBCNT = CMPA or CMPB). If CMPA/CMPB is ≥ TBPRD, the event will occur on a period match (TBCNT = TBPRD).	If CMPA/CMPB < TBPRD and the counter is decrementing, the event occurs on a compare match (TBCNT = CMPA or CMPB). If CMPA/CMPB ≥ TBPRD, the event occurs on a period match (TBCNT = TBPRD).

11.5.4.4.4 Waveforms for Common ePWM Configurations

NOTE: The waveforms in this chapter show the ePWMs behavior for a static compare register value. In a running system, the active compare registers (EPWM_CMPA and EPWM_CMPB) are typically updated from their respective shadow registers once every period. The user specifies when the update will take place — either when the time-base counter reaches zero or when the time-base counter reaches period. There are some cases when the action based on the new value can be delayed by one period or the action based on the old value can take effect for an extra period. Some PWM configurations avoid this situation. These include, but are not limited to, the following:

Use up-down-count mode to generate a symmetric PWM:

- If EPWM_CMPA / EPWM_CMPB is loaded on zero, then use EPWM_CMPA / EPWM_CMPB values greater than or equal to 1.
- If EPWM_CMPA / EPWM_CMPB is loaded on period, then use EPWM_CMPA / EPWM_CMPB values less than or equal to TBPRD - 1.

  This means there will always be a pulse of at least one TBCLK cycle in a PWM period which, when very short, tend to be ignored by the system.

Use up-down-count mode to generate an asymmetric PWM:

- To achieve 50-0% asymmetric PWM use the following configuration: Load EPWM_CMPA / EPWM_CMPB on period and use the period action to clear the PWM and a compare-up action to set the PWM. Modulate the compare value from 0 to TBPRD to achieve 50-0% PWM duty.

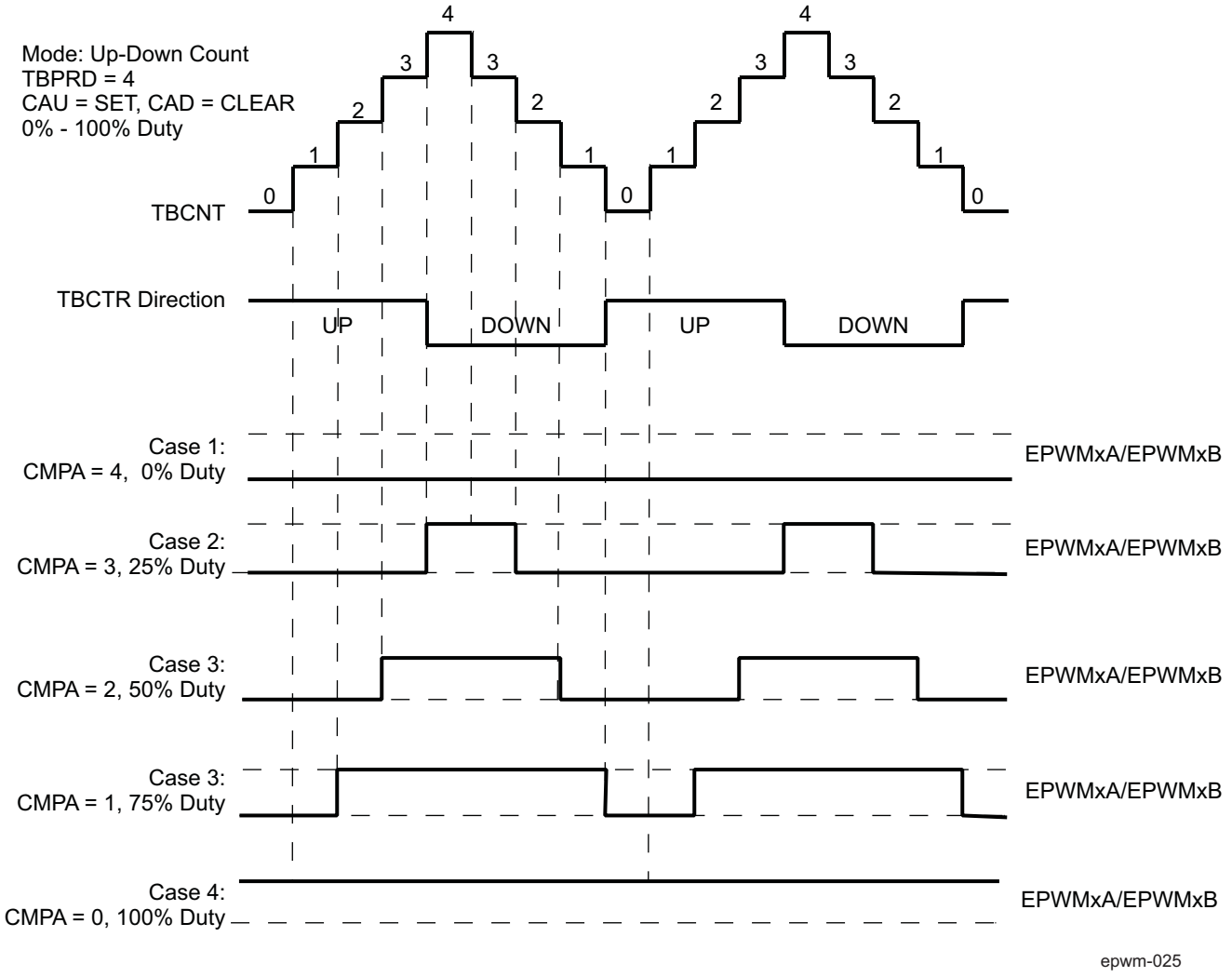
When using up-count mode to generate an asymmetric PWM:

- To achieve 0-100% asymmetric PWM use the following configuration: Load EPWM_CMPA / EPWM_CMPB on TBPRD. Use the Zero action to set the PWM and a compare-up action to clear the PWM. Modulate the compare value from 0 to TBPRD+1 to achieve 0-100% PWM duty.

Figure 11-373 shows how a symmetric PWM waveform can be generated using the up-down-count mode of the TBCNT. In this mode 0-100% DC modulation is achieved by using equal compare matches on the up count and down count portions of the waveform. In the example shown, CMPA is used to make the comparison. When the counter is incrementing the CMPA match will pull the PWM output high. Likewise, when the counter is decrementing the compare match will pull the PWM signal low. When CMPA = 0, the PWM signal is low for the entire period giving the 0% duty waveform. When EPWM_CMPA = EPWM_TBPRD, the PWM signal is high achieving 100% duty.

When using this configuration in practice, if CMPA/CMPB is loaded on zero, then use CMPA/CMPB values greater than or equal to 1. If CMPA/CMPB is loaded on period, then use CMPA/CMPB values less than or equal to TBPRD-1. This means there will always be a pulse of at least one TBCLK cycle in a PWM period which, when very short, tend to be ignored by the system.

Figure 11-373. ePWM Up-Down-Count Mode Symmetrical Waveform

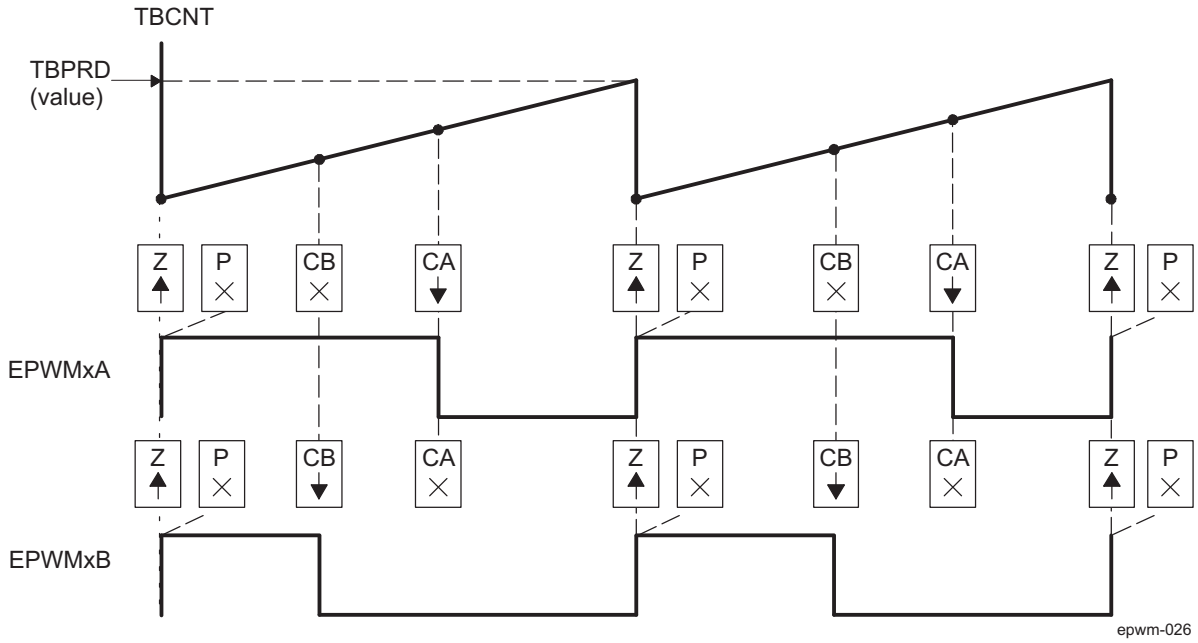


The PWM waveforms in Figure 11-374 through Figure 11-379 show some common action-qualifier configurations. Some conventions used in the figures are as follows:

- TBPRD, CMPA, and CMPB refer to the value written in their respective registers (EPWM_TBPRD, EPWM_CMPA, and EPWM_CMPB). The active register, not the shadow register, is used by the hardware.
- CMPx, refers to either CMPA or CMPB.
- EPWMxA and EPWMxB refer to the output signals from ePWMx
- Up-Down means Count-up-and-down mode, Up means up-count mode and Dwn means down-count mode
- Sym = Symmetric, Asym = Asymmetric

Table 11-807 and Table 11-808 contains initialization and runtime register configurations for the waveforms in Figure 11-374.

Figure 11-374. Up, Single Edge Asymmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB — Active High



(1) PWM period = (TBPRD + 1) × T_{TBCLK}

(2) Duty modulation for EPWMxA is set by CMPA, and is active high (that is, high time duty proportional to CMPA).

(3) Duty modulation for EPWMxB is set by CMPB and is active high (that is, high time duty proportional to CMPB).

(4) The "Do Nothing" actions (X) are shown for completeness, but will not be shown on subsequent diagrams.

(5) Actions at zero and period, although appearing to occur concurrently, are actually separated by one TBCLK period. TBCNT wraps from period to 0000h.

Table 11-807. EPWMx Initialization for Figure 11-374

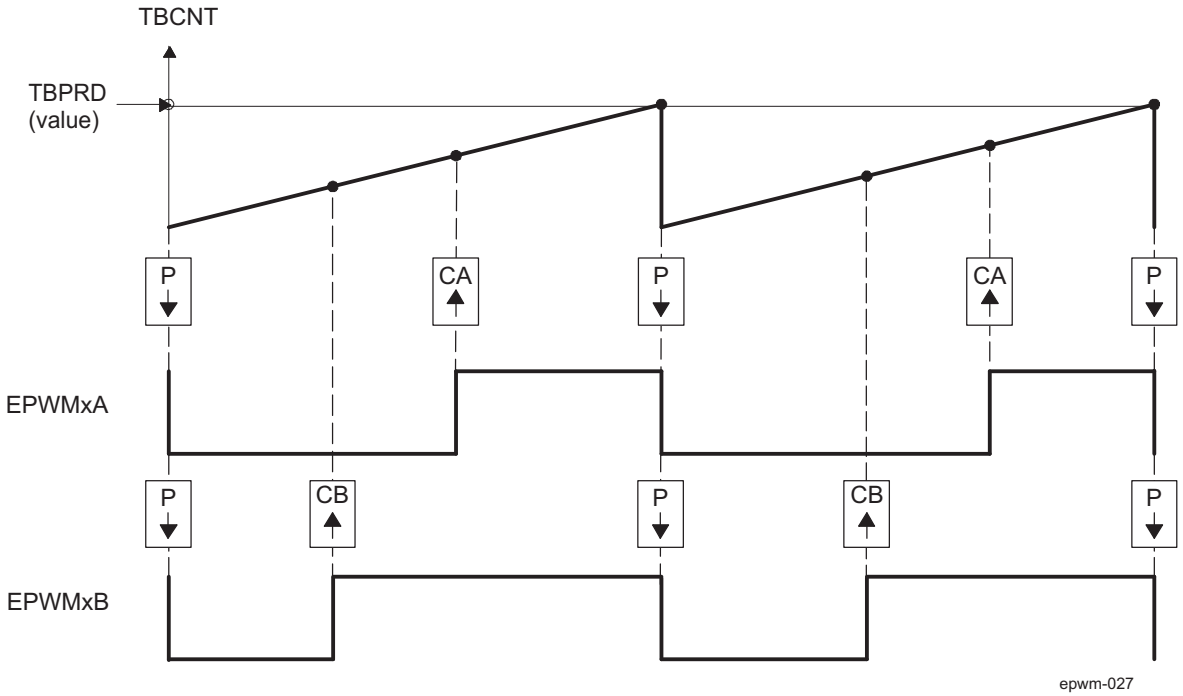
Register	Bitfield	Value	Comments
EPWM_TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
EPWM_TBPHS	TBPHS	0	Clear Phase Register to 0
EPWM_TBCNT	TBCNT	0	Clear TB counter
EPWM_TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = VBUS_CLK
	CLKDIV	TB_DIV1	
EPWM_CMPA	CMPA	350 (15Eh)	Compare A = 350 TBCLK counts
EPWM_CMPB	CMPB	200 (C8h)	Compare B = 200 TBCLK counts
EPWM_CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on TBCNT = 0
	LOADBMODE	CC_CTR_ZERO	Load on TBCNT = 0
EPWM_AQCTLA	ZRO	AQ_SET	
	CAU	AQ_CLEAR	
EPWM_AQCTLB	ZRO	AQ_SET	
	CBU	AQ_CLEAR	

Table 11-808. EPWMx Run Time Changes for Figure 11-374

Register	Bitfield	Value	Comments
EPWM_CMPA	CMPA	Duty1A	Adjust duty for output EPWM1A
EPWM_CMPB	CMPB	Duty1B	Adjust duty for output EPWM1B

Table 11-809 and Table 11-810 contains initialization and runtime register configurations for the waveforms in Figure 11-375.

Figure 11-375. Up, Single Edge Asymmetric Waveform With Independent Modulation on EPWMxA and EPWMxB — Active Low



(1) PWM period = (TBPRD + 1) × T_{TBCLK}

(2) Duty modulation for EPWMxA is set by CMPA, and is active low (that is, the low time duty is proportional to CMPA).

(3) Duty modulation for EPWMxB is set by CMPB and is active low (that is, the low time duty is proportional to CMPB).

(4) The Do Nothing actions (X) are shown for completeness here, but will not be shown on subsequent diagrams.

(5) Actions at zero and period, although appearing to occur concurrently, are actually separated by one TBCLK period. TBCNT wraps from period to 0000h.

Table 11-809. EPWMx Initialization for Figure 11-375

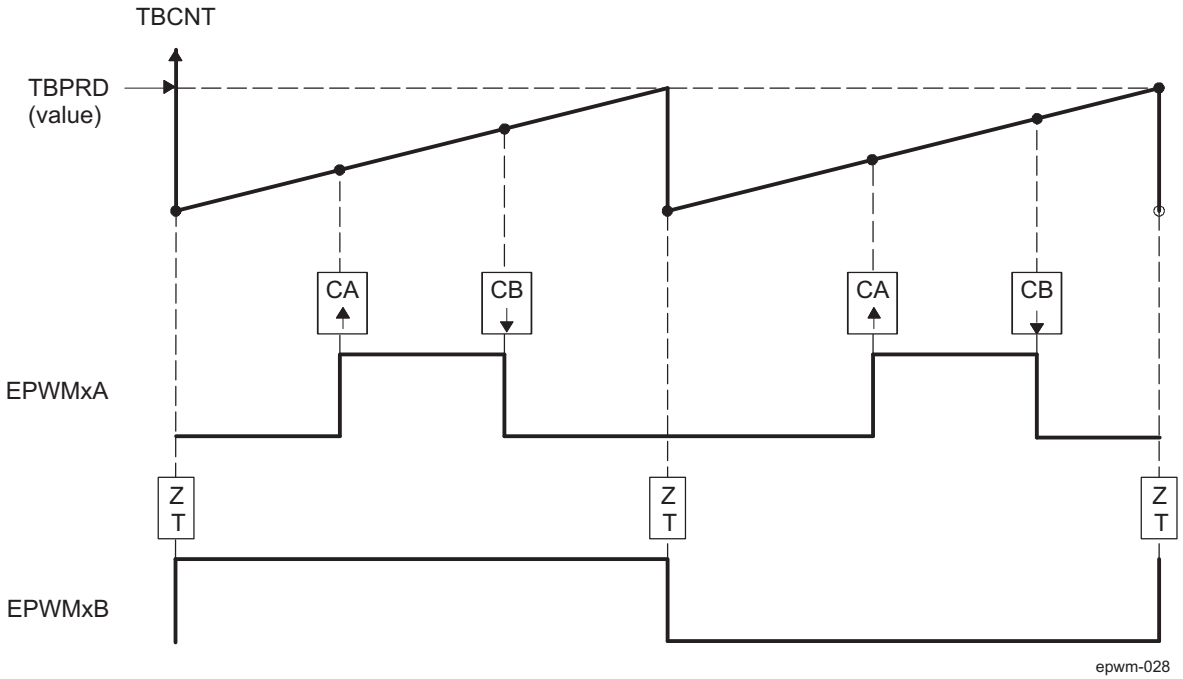
Register	Bitfiled	Value	Comments
EPWM_TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
EPWM_TBPHS	TBPHS	0	Clear Phase Register to 0
EPWM_TBCNT	TBCNT	0	Clear TB counter
EPWM_TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = VBUS_CLK
	CLKDIV	TB_DIV1	
EPWM_CMPA	CMPA	350 (15Eh)	Compare A = 350 TBCLK counts
EPWM_CMPB	CMPB	200 (C8h)	Compare B = 200 TBCLK counts
EPWM_CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on TBCNT = 0
	LOADBMODE	CC_CTR_ZERO	Load on TBCNT = 0
EPWM_AQCTLA	PRD	AQ_CLEAR	
	CAU	AQ_SET	
EPWM_AQCTLB	PRD	AQ_CLEAR	
	CBU	AQ_SET	

Table 11-810. EPWMx Run Time Changes for Figure 11-375

Register	Bit	Value	Comments
EPWM_CMPA	CMPA	Duty1A	Adjust duty for output EPWM1A
EPWM_CMPB	CMPB	Duty1B	Adjust duty for output EPWM1B

Table 11-811 and Table 11-812 contains initialization and runtime register configurations for the waveforms Figure 11-376. Use the code in Example 11-1 to define the headers.

Figure 11-376. Up-Count, Pulse Placement Asymmetric Waveform With Independent Modulation on EPWMxA



(1) PWM frequency = 1/((TBPRD + 1) × T_{TBCLK})

(2) Pulse can be placed anywhere within the PWM cycle (0000h - TBPRD)

(3) High time duty proportional to (CMPB - CMPA)

(4) EPWMxB can be used to generate a 50% duty square wave with frequency = 1/2 × ((TBPRD + 1) × TBCLK)

Table 11-811. EPWMx Initialization for Figure 11-376

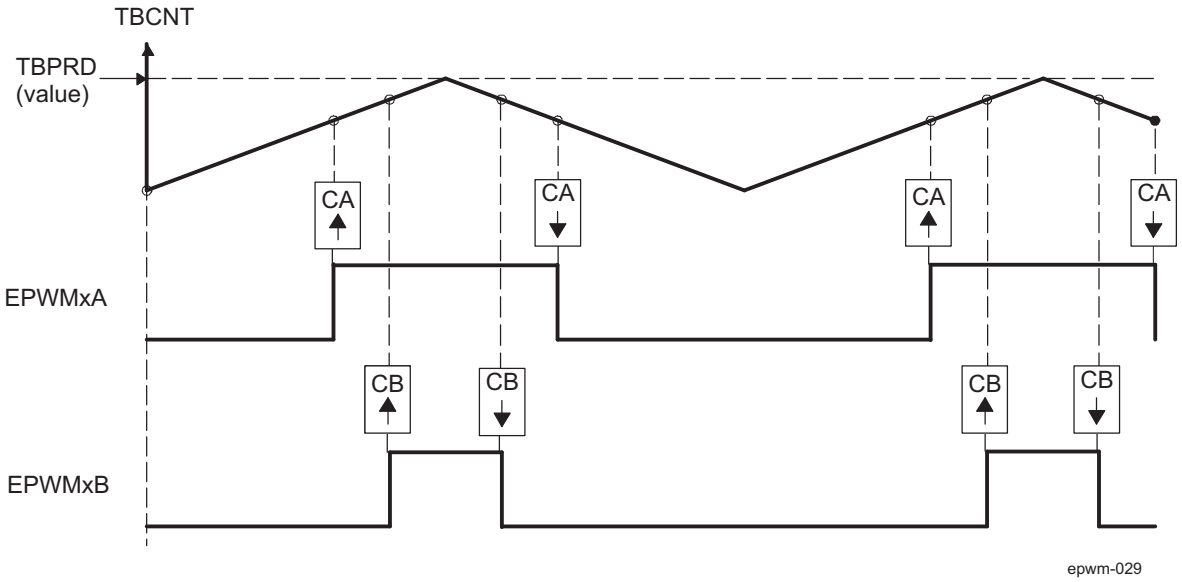
Register	Bitfield	Value	Comments
EPWM_TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
EPWM_TBPHS	TBPHS	0	Clear Phase Register to 0
EPWM_TBCNT	TBCNT	0	Clear TB counter
EPWM_TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = VBUS_CLK
	CLKDIV	TB_DIV1	
EPWM_CMPA	CMPA	200 (C8h)	Compare A = 200 TBCLK counts
EPWM_CMPB	CMPB	400 (190h)	Compare B = 400 TBCLK counts
EPWM_CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on TBCNT = 0
	LOADBMODE	CC_CTR_ZERO	Load on TBCNT = 0
EPWM_AQCTLA	CAU	AQ_SET	
	CBU	AQ_CLEAR	
EPWM_AQCTLB	ZRO	AQ_TOGGLE	

Table 11-812. EPWMx Run Time Changes for Figure 11-376

Register	Bitfield	Value	Comments
EPWM_CMPA	CMPA	EdgePosA	Adjust duty for output EPWM1A
EPWM_CMPB	CMPB	EdgePosB	

Table 11-813 and Table 11-814 contains initialization and runtime register configurations for the waveforms in Figure 11-377. Use the code in Example 11-1 to define the headers.

Figure 11-377. Up-Down-Count, Dual Edge Symmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB — Active Low



(1) PWM period = 2 x TBPRD × T_{TBCLK}

(2) Duty modulation for EPWMxA is set by CMPA, and is active low (that is, the low time duty is proportional to CMPA).

(3) Duty modulation for EPWMxB is set by CMPB and is active low (that is, the low time duty is proportional to CMPB).

(4) Outputs EPWMxA and EPWMxB can drive independent power switches

Table 11-813. EPWMx Initialization for Figure 11-377

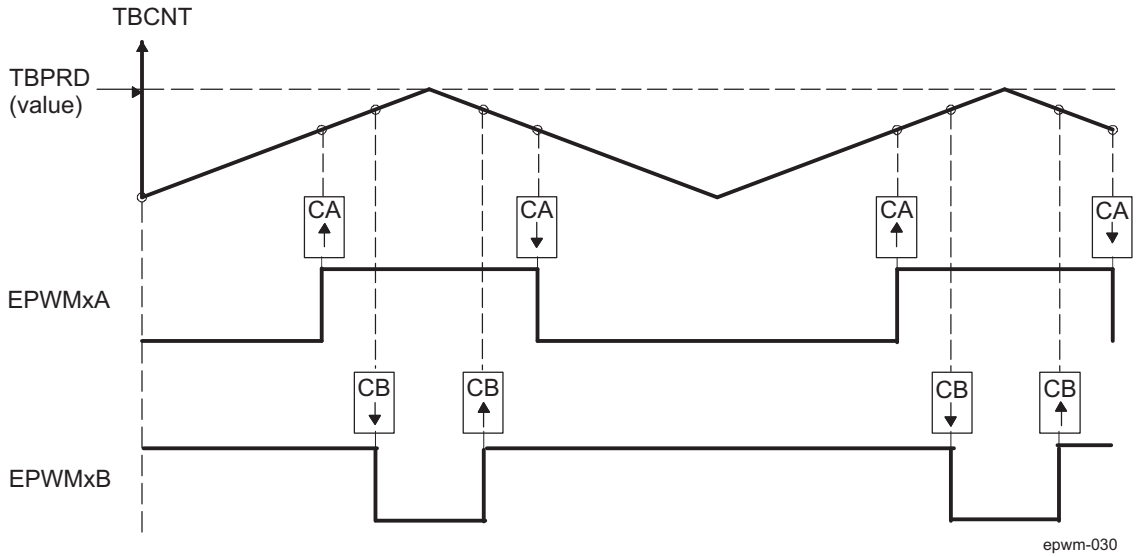
Register	Bitfield	Value	Comments
EPWM_TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
EPWM_TBPHS	TBPHS	0	Clear Phase Register to 0
EPWM_TBCNT	TBCNT	0	Clear TB counter
EPWM_TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = VBUS_CLK
	CLKDIV	TB_DIV1	
EPWM_CMPA	CMPA	400 (190h)	Compare A = 400 TBCLK counts
EPWM_CMPB	CMPB	500 (1F4h)	Compare B = 500 TBCLK counts
EPWM_CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on TBCNT = 0
	LOADBMODE	CC_CTR_ZERO	Load on TBCNT = 0
EPWM_AQCTLA	CAU	AQ_SET	
	CAD	AQ_CLEAR	
EPWM_AQCTLB	CBU	AQ_SET	
	CBD	AQ_CLEAR	

Table 11-814. EPWMx Run Time Changes for Figure 11-377

Register	Bitfield	Value	Comments
EPWM_CMPA	CMPA	Duty1A	Adjust duty for output EPWM1A
EPWM_CMPB	CMPB	Duty1B	Adjust duty for output EPWM1B

Table 11-815 and Table 11-816 contains initialization and runtime register configurations for the waveforms in Figure 11-378. Use the code in Example 11-1 to define the headers.

Figure 11-378. Up-Down-Count, Dual Edge Symmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB — Complementary



(1) PWM period = 2 × TBPRD × T_{TBCLK}

(2) Duty modulation for EPWMxA is set by CMPA, and is active low, that is, low time duty proportional to CMPA

(3) Duty modulation for EPWMxB is set by CMPB and is active high, that is, high time duty proportional to CMPB

(4) Outputs EPWMx can drive upper/lower (complementary) power switches

(5) Dead-band = CMPB - CMPA (fully programmable edge placement by software). Note the dead-band module is also available if the more classical edge delay method is required.

Table 11-815. EPWMx Initialization for Figure 11-378

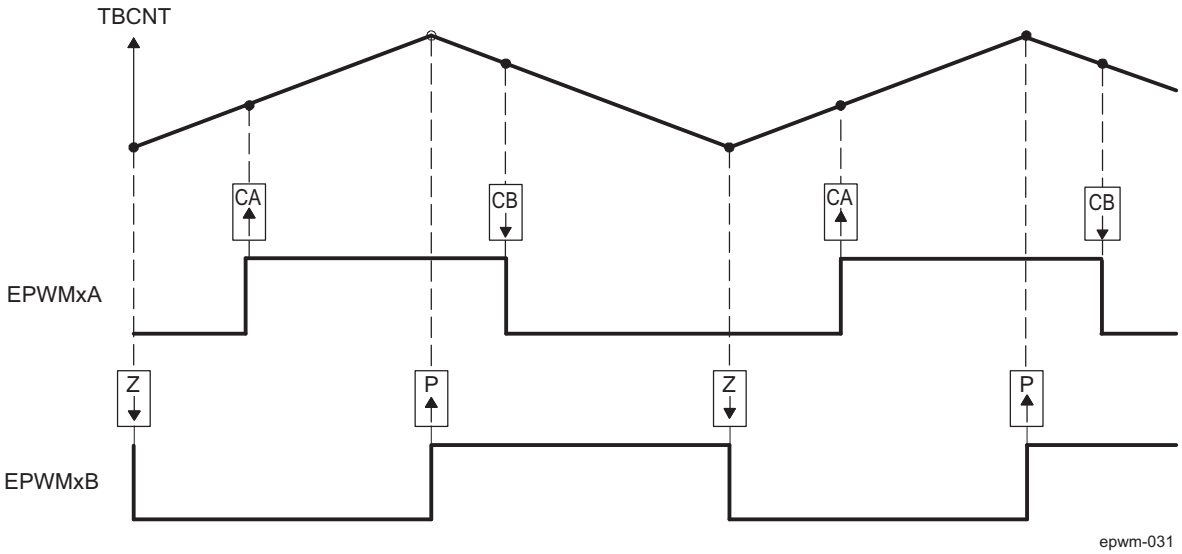
Register	Bitfield	Value	Comments
EPWM_TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
EPWM_TBPHS	TBPHS	0	Clear Phase Register to 0
EPWM_TBCNT	TBCNT	0	Clear TB counter
EPWM_TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = VBUS_CLK
	CLKDIV	TB_DIV1	
EPWM_CMPA	CMPA	350 (15Eh)	Compare A = 350 TBCLK counts
EPWM_CMPB	CMPB	400 (190h)	Compare B = 400 TBCLK counts
EPWM_CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on TBCNT = 0
	LOADBMODE	CC_CTR_ZERO	Load on TBCNT = 0
EPWM_AQCTLA	CAU	AQ_SET	
	CAD	AQ_CLEAR	
EPWM_AQCTLB	CBU	AQ_CLEAR	
	CBD	AQ_SET	

Table 11-816. EPWMx Run Time Changes for Figure 11-378

Register	Bitfield	Value	Comments
EPWM_CMPA	CMPA	Duty1A	Adjust duty for output EPWM1A
EPWM_CMPB	CMPB	Duty1B	Adjust duty for output EPWM1B

Table 11-817 and Table 11-818 contains initialization and runtime register configurations for the waveforms in Figure 11-379. Use the code in Example 11-1 to define the headers.

Figure 11-379. Up-Down-Count, Dual Edge Asymmetric Waveform, With Independent Modulation on EPWMxA — Active Low



(1) PWM period = 2 × TBPRD × TBCLK

(2) Rising edge and falling edge can be asymmetrically positioned within a PWM cycle. This allows for pulse placement techniques.

(3) Duty modulation for EPWMxA is set by CMPA and CMPB.

(4) Low time duty for EPWMxA is proportional to (CMPA + CMPB).

(5) To change this example to active high, CMPA and CMPB actions need to be inverted (that is, Set ! Clear and Clear Set).

(6) Duty modulation for EPWMxB is fixed at 50% (utilizes spare action resources for EPWMxB)

Table 11-817. EPWMx Initialization for Figure 11-379

Register	Bitfield	Value	Comments
EPWM_TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
EPWM_TBPHS	TBPHS	0	Clear Phase Register to 0
EPWM_TBCNT	TBCNT	0	Clear TB counter
EPWM_TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDLD	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = VBUS_CLK
	CLKDIV	TB_DIV1	
EPWM_CMPA	CMPA	250 (FAh)	Compare A = 250 TBCLK counts
EPWM_CMPB	CMPB	450 (1C2h)	Compare B = 450 TBCLK counts
EPWM_CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on TBCNT = 0
	LOADBMODE	CC_CTR_ZERO	Load on TBCNT = 0
EPWM_AQCTLA	CAU	AQ_SET	
	CBD	AQ_CLEAR	
EPWM_AQCTLB	ZRO	AQ_CLEAR	
	PRD	AQ_SET	

Table 11-818. EPWMx Run Time Changes for Figure 11-379

Register	Bitfield	Value	Comments
EPWM_CMPA	CMPA	EdgePosA	Adjust duty for output EPWM1A
EPWM_CMPB	CMPB	EdgePosB	

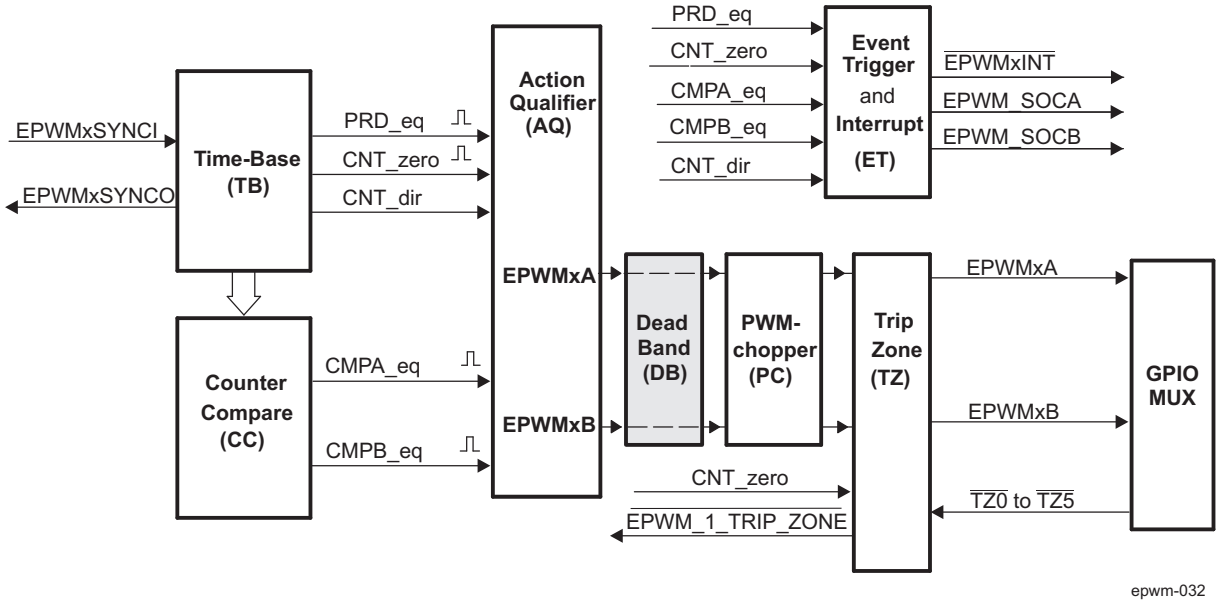
11.5.4.5 ePWM Dead-Band Generator (DB) Submodule

This section describes the Dead-Band Generator (DB) submodule in the PWM module.

11.5.4.5.1 Overview

Figure 11-380 illustrates the dead-band generator submodule within the ePWM module.

Figure 11-380. Dead-Band Generator Submodule



The "Action-qualifier (AQ) Module" section discussed how it is possible to generate the required dead-band by having full control over edge placement using both the CMPA and CMPB resources of the ePWM module. However, if the more classical edge delay-based dead-band with polarity control is required, then the dead-band generator submodule should be used.

The key functions of the dead-band generator submodule are:

- Generating appropriate signal pairs (EPWMxA and EPWMxB) with dead-band relationship from a single EPWMxA input
- Programming signal pairs for:
  - Active high (AH)
  - Active low (AL)
  - Active high complementary (AHC)
  - Active low complementary (ALC)
- Adding programmable delay to rising edges (RED)
- Adding programmable delay to falling edges (FED)
- Can be totally bypassed from the signal path (note dotted lines in Figure 11-380)

11.5.4.5.2 Controlling and Monitoring the ePWM Dead-Band Submodule

The dead-band generator submodule operation is controlled and monitored via the following registers:

Table 11-819. Dead-Band Generator Submodule Registers

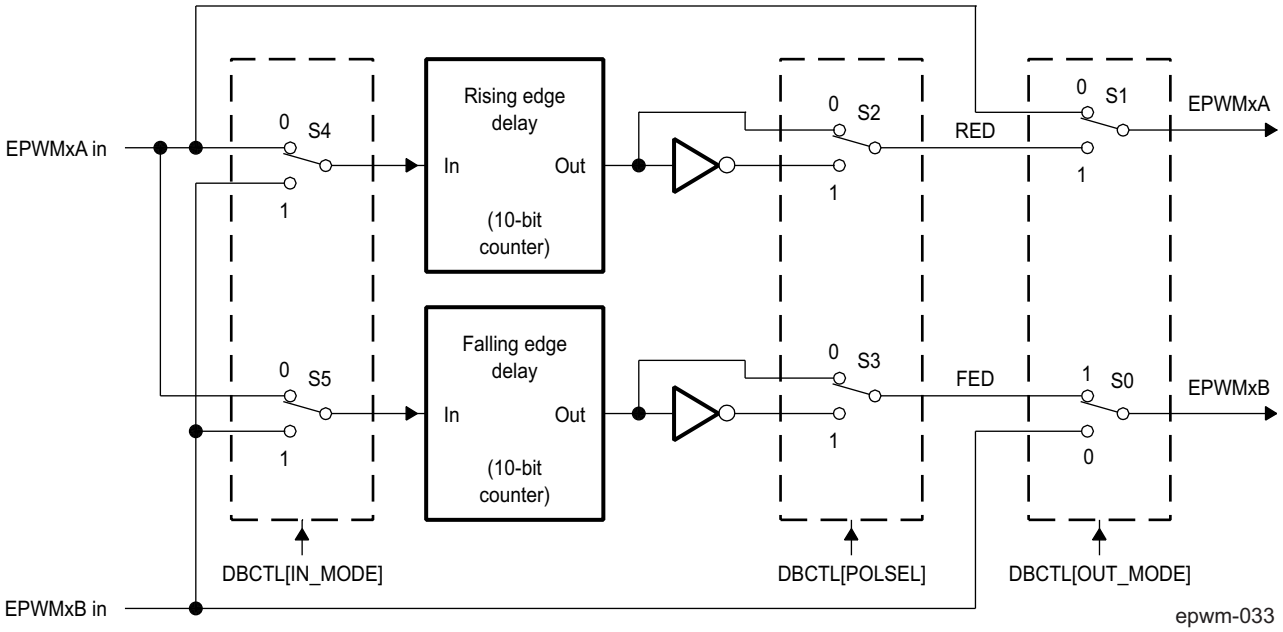
Acronym	Register Description	Address Offset	Shadowed
EPWM_DBCTL	Dead-Band Control Register	1Eh	No
EPWM_DBRED	Dead-Band Rising Edge Delay Count Register	20h	No
EPWM_DBFED	Dead-Band Falling Edge Delay Count Register	22h	No

11.5.4.5.3 Operational Highlights for the ePWM Dead-Band Generator Submodule

The dead-band submodule has two groups of independent selection options as shown in Figure 11-381.

- **Input Source Selection:** The input signals to the dead-band module are the EPWMxA and EPWMxB output signals from the action-qualifier. In this section they will be referred to as EPWMxA In and EPWMxB In. Using the EPWM_DBCTL[5-4] IN_MODE control bits, the signal source for each delay, falling-edge or rising-edge, can be selected:
  - EPWMxA In is the source for both falling-edge and rising-edge delay. This is the default mode.
  - EPWMxA In is the source for falling-edge delay, EPWMxB In is the source for rising-edge delay.
  - EPWMxA In is the source for rising edge delay, EPWMxB In is the source for falling-edge delay.
  - EPWMxB In is the source for both falling-edge and rising-edge delay.
- **Output Mode Control:** The output mode is configured by way of the EPWM_DBCTL[1-0] OUT_MODE bits. These bits determine if the falling-edge delay, rising-edge delay, neither, or both are applied to the input signals.
- **Polarity Control:** The polarity control (EPWM_DBCTL[3-2] POLSEL) allows to be specified whether the rising-edge delayed signal and/or the falling-edge delayed signal is to be inverted before being sent out of the dead-band submodule.

Figure 11-381. Configuration Options for the ePWM Dead-Band Generator Submodule



Although all combinations are supported, not all are typical usage modes. Table 11-820 lists some classical dead-band configurations. These modes assume that the EPWM_DBCTL[5-4] IN_MODE is configured such that EPWMxA In is the source for both falling-edge and rising-edge delay. Enhanced, or non-traditional modes can be achieved by changing the input signal source. The modes shown in Table 11-820 fall into the following categories:

- **Mode 1: Bypass both falling-edge delay (FED) and rising-edge delay (RED)** Allows to be fully disabled the dead-band submodule from the PWM signal path.
- **Mode 2-5: Classical Dead-Band Polarity Settings** These represent typical polarity configurations that should address all the active high/low modes required by available industry power switch gate drivers. The waveforms for these typical cases are shown in Figure 11-382. Note that to generate equivalent waveforms to Figure 11-382, configure the action-qualifier submodule to generate the signal as shown for EPWMxA.
- **Mode 6: Bypass rising-edge-delay and Mode 7: Bypass falling-edge-delay** Finally the last two entries in Table 11-820 show combinations where either the falling-edge-delay (FED) or rising-edge-delay (RED) blocks are bypassed.

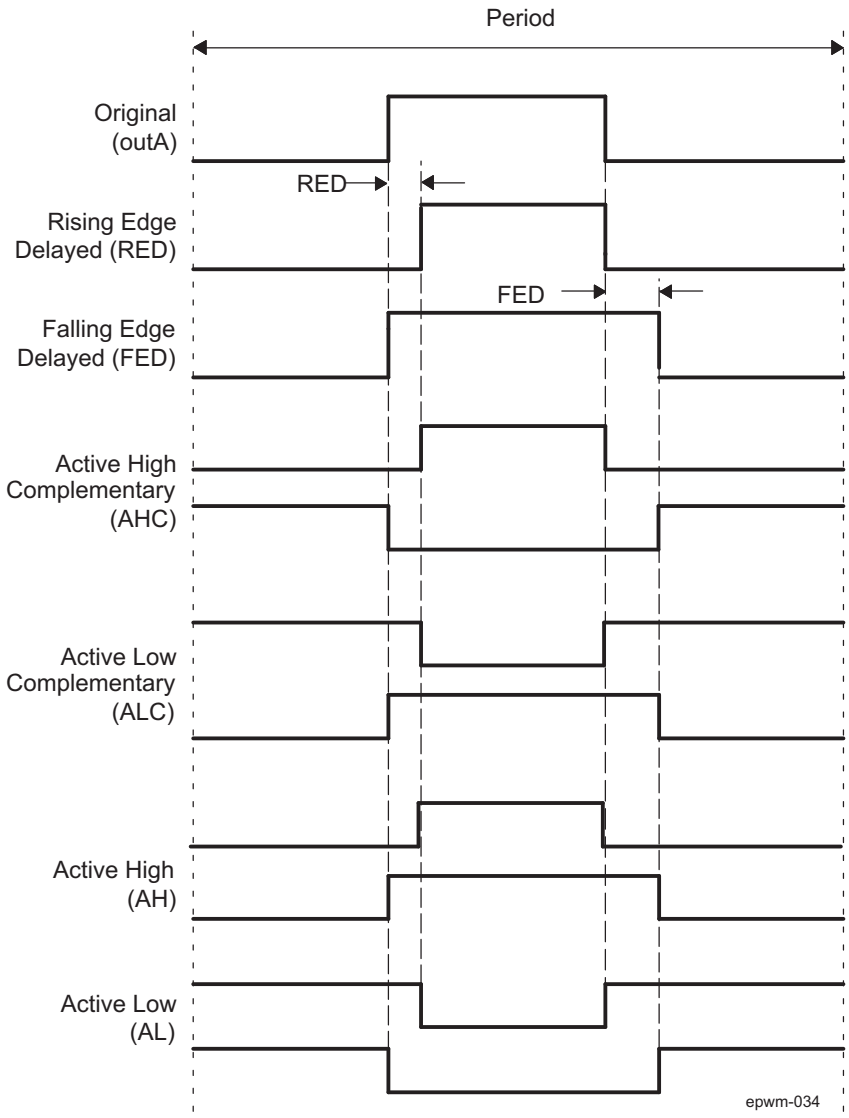
Table 11-820. Classical Dead-Band Operating Modes

		EPWM_DBCTL[3-2] POLSEL		EPWM_DBCTL[1-0] OUT_MODE	
Mode	Mode Description [1]	S3	S2	S1	S0
1	EPWMxA and EPWMxB Passed Through (No Delay)	x	x	0	0
2	Active High Complementary (AHC)	1	0	1	1
3	Active Low Complementary (ALC)	0	1	1	1
4	Active High (AH)	0	0	1	1
5	Active Low (AL)	1	1	1	1
6	EPWMxA Out = EPWMxA In (No Delay) EPWMxB Out = EPWMxA In with Falling Edge Delay	0 or 1	0 or 1	0	1
7	EPWMxA Out = EPWMxA In with Rising Edge Delay EPWMxB Out = EPWMxB In with No Delay	0 or 1	0 or 1	1	0

(1) These are classical dead-band modes and assume that EPWM_DBCTL[5-4] IN_MODE = 0b00. That is, EPWMxA in is the source for both the falling-edge and rising-edge delays. Enhanced, non-traditional modes can be achieved by changing the IN_MODE configuration.

Figure 11-382 shows waveforms for typical cases where 0% < duty < 100%.

Figure 11-382. Dead-Band Waveforms for Typical Cases (0% < Duty < 100%)



The dead-band submodule supports independent values for rising-edge (RED) and falling-edge (FED) delays. The amount of delay is programmed using the EPWM_DBRED and EPWM_DBFED registers. These are 10-bit registers and their value represents the number of time-base clock, TBCLK, periods a signal edge is delayed by. For example, the formulas to calculate falling-edge-delay and rising-edge-delay are:

FED = EPWM_DBFED × T_{TBCLK}

RED = EPWM_DBRED × T_{TBCLK}

Where T_{TBCLK} is the period of TBCLK, the prescaled version of VBUS_CLK.

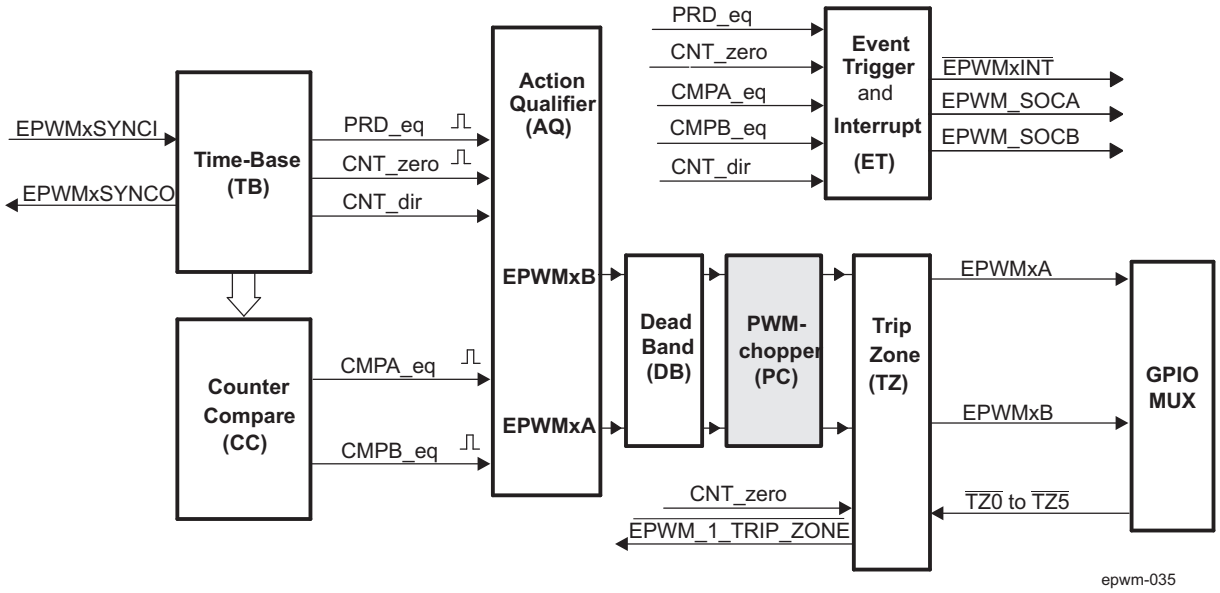
11.5.4.6 ePWM-Chopper (PC) Submodule

This section describes the PWM-Chopper (PC) submodule in the PWM module.

11.5.4.6.1 Overview

Figure 11-383 illustrates the PWM-chopper (PC) submodule within the ePWM module. The PWM-chopper submodule allows a high-frequency carrier signal to modulate the PWM waveform generated by the action-qualifier and dead-band submodules. This capability is important if a pulse transformer-based gate drivers to control the power switching elements is needed.

Figure 11-383. PWM-Chopper Submodule



PC module key features:

- Programmable chopping (carrier) frequency
- Programmable pulse width of first pulse
- Programmable duty cycle of second and subsequent pulses
- Can be fully bypassed if not required

11.5.4.6.2 Controlling the ePWM-Chopper Submodule

The ePWM-chopper submodule operation is controlled via the register in Table 11-821.

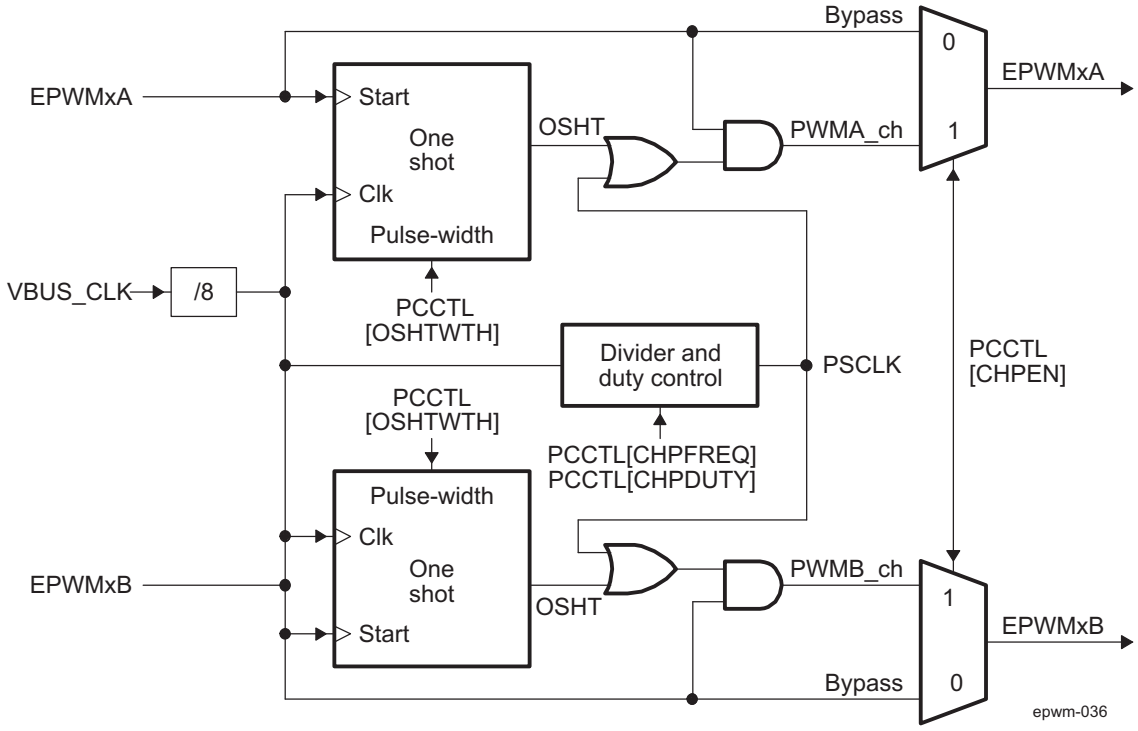
Table 11-821. ePWM-Chopper Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
EPWM_PCCTL	PWM-chopper Control Register	3Ch	No

11.5.4.6.3 Operational Highlights for the ePWM-Chopper Submodule

Figure 11-384 shows the operational details of the ePWM-chopper submodule. The carrier clock is derived from VBUS_CLK. Its frequency and duty cycle are controlled via the CHPFREQ and CHPDUTY bits in the EPWM_PCCTL register. The one-shot block is a feature that provides a high energy first pulse to ensure hard and fast power switch turn on, while the subsequent pulses sustain pulses, ensuring the power switch remains on. The one-shot width is programmed via the OSHTWTH bits. The PWM-chopper submodule can be fully disabled (bypassed) via the CHPEN bit.

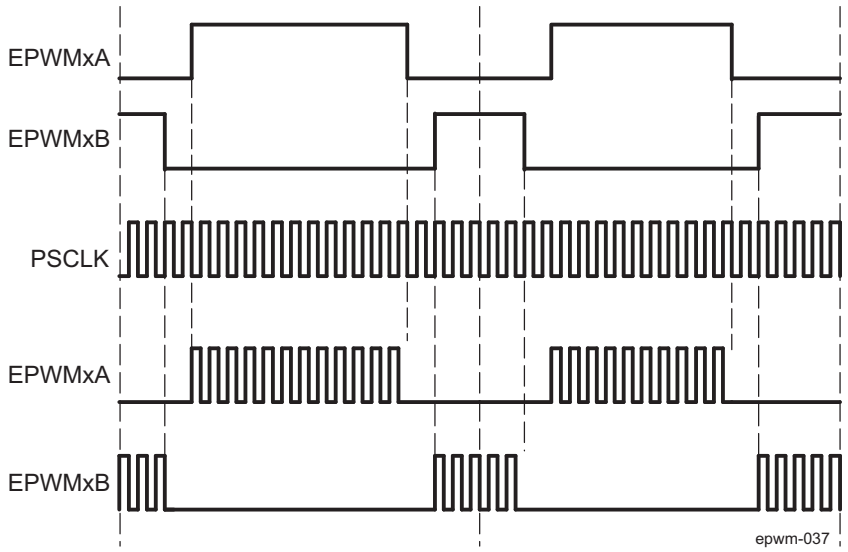
Figure 11-384. PWM-Chopper Submodule Signals and Registers



11.5.4.6.4 ePWM-Chopper Waveforms

Figure 11-385 shows simplified waveforms of the chopping action only; one-shot and duty-cycle control are not shown. Details of the one-shot and duty-cycle control are discussed in the following sections.

Figure 11-385. Simple ePWM-Chopper Submodule Waveforms Showing Chopping Action Only



11.5.4.6.4.1 ePWM-Chopper One-Shot Pulse

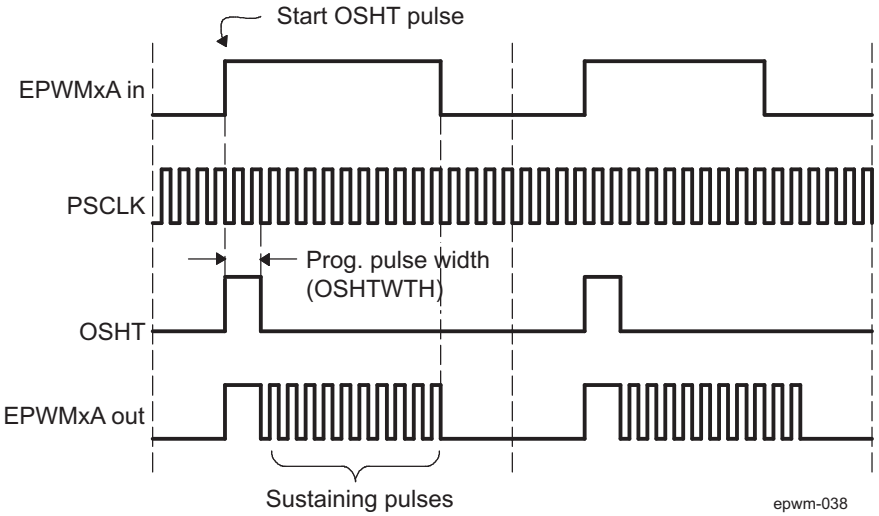
The width of the first pulse can be programmed to any of 16 possible pulse width values. The width or period of the first pulse is given by:

$T_{1stpulse} = T_{VBUS\_CLK} \times 8 \times OSHTWTH$

Where T_{VBUS_CLK} is the period of the system clock (VBUS_CLK) and OSHTWTH is the four control bits (value from 1 to 16)

Figure 11-386 shows the first and subsequent sustaining pulses.

Figure 11-386. ePWM-Chopper Submodule Waveforms Showing the First Pulse and Subsequent Sustaining Pulses

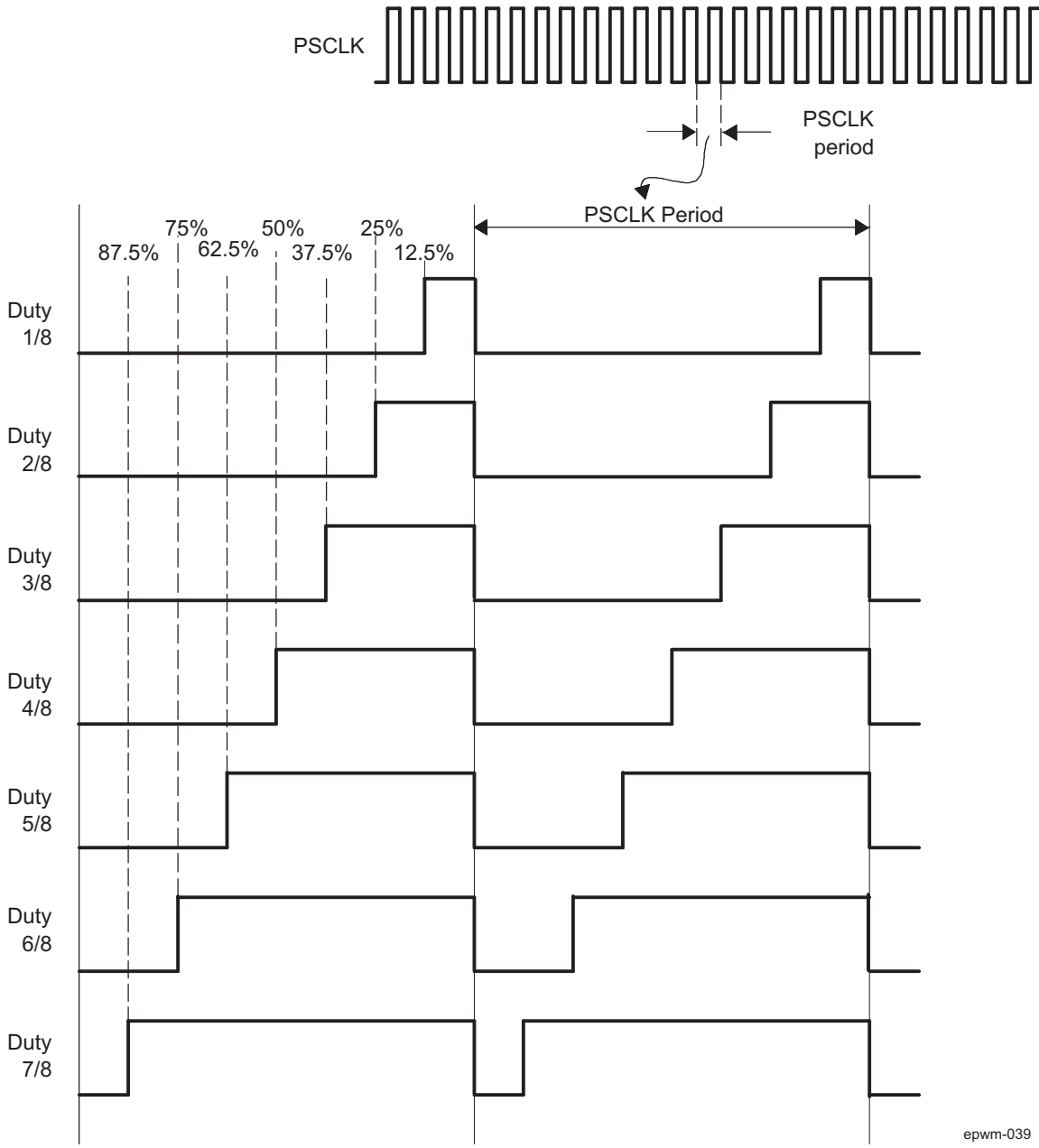


11.5.4.6.4.2 ePWM-Chopper Duty Cycle Control

Pulse transformer-based gate drive designs need to comprehend the magnetic properties or characteristics of the transformer and associated circuitry. Saturation is one such consideration. To assist the gate drive designer, the duty cycles of the second and subsequent pulses have been made programmable. These sustaining pulses ensure the correct drive strength and polarity is maintained on the power switch gate during the on period, and hence a programmable duty cycle allows a design to be tuned or optimized via software control.

Figure 11-387 shows the duty cycle control that is possible by programming the CHPDUTY bits. One of seven possible duty ratios can be selected ranging from 12.5 to 87.5%.

Figure 11-387. ePWM-Chopper Submodule Waveforms Showing the Pulse Width (Duty Cycle) Control of Sustaining Pulses



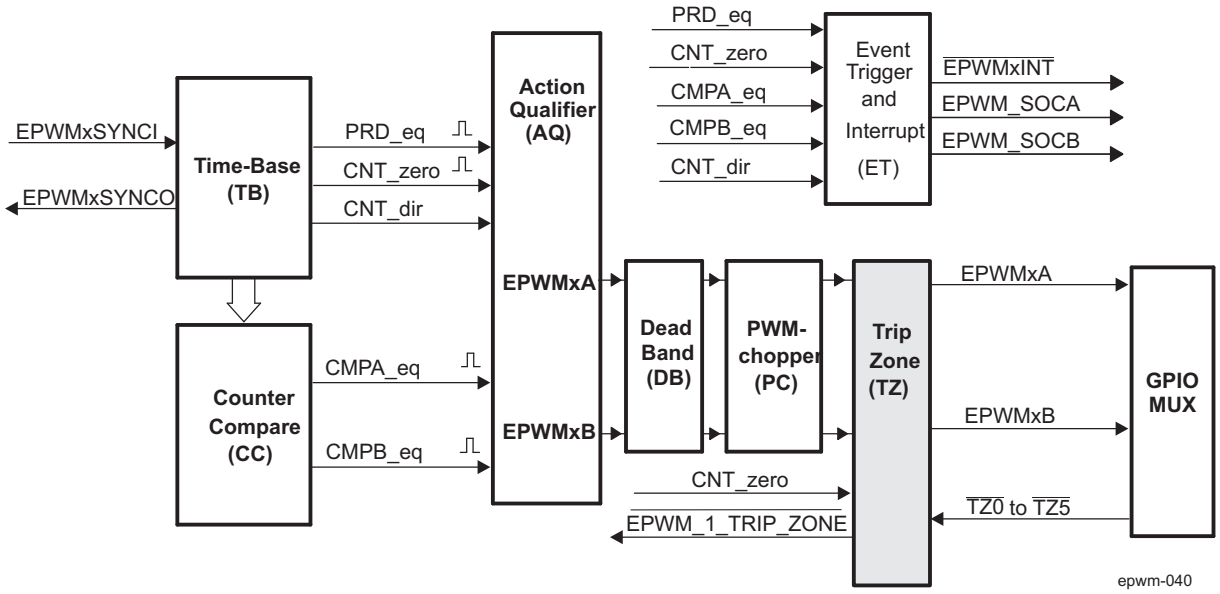
11.5.4.7 ePWM Trip-Zone (TZ) Submodule

This section describes the Trip-Zone (TZ) submodule in the PWM module.

11.5.4.7.1 Overview

Figure 11-388 shows how the trip-zone (TZ) submodule fits within the ePWM module. Each ePWM module is connected to six $\overline{TZ}$ signals ($\overline{TZ0}$ to $\overline{TZ5}$) that are sourced from the GPIO MUX. These signals indicates external fault or trip conditions, and the ePWM outputs can be programmed to respond accordingly when faults occur. See Section 11.6.3 to determine the number of trip-zone pins available for the device.

Figure 11-388. ePWM Trip-Zone Submodule



The key functions of the trip-zone submodule are:

- Trip inputs $\overline{TZ0}$ to $\overline{TZ5}$ can be flexibly mapped to any ePWM module.
- Upon a fault condition, outputs EPWMxA and EPWMxB can be forced to one of the following:
  - High
  - Low
  - High-impedance
  - No action taken
- Support for one-shot trip (OSHT) for major short circuits or over-current conditions.
- Support for cycle-by-cycle tripping (CBC) for current limiting operation.
- Each trip-zone input pin can be allocated to either one-shot or cycle-by-cycle operation.
- Interrupt generation is possible on any trip-zone pin.
- Software-forced tripping is also supported.
- The trip-zone submodule can be fully bypassed if it is not required.

NOTE: For each ePWM_x module, 6 tripzone input events are supported ($\overline{EPWMx\_TRIP\_TZ[5:0]}$). Each tripzone input for all ePWM_x modules (where x = 0 to 5) are tied to a common tripzone input pin, so that any ePWM_x can be triggered by any of the six tripzone inputs.

11.5.4.7.2 Controlling and Monitoring the ePWM Trip-Zone Submodule

The trip-zone submodule operation is controlled and monitored through the following registers:

Table 11-822. ePWM Trip-Zone Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
EPWM_TZSEL	Trip-Zone Select Register(1)	24h	No
EPWM_TZCTL	Trip-Zone Control Register(1)	28h	No
EPWM_TZEINT	Trip-Zone Enable Interrupt Register(1)	2Ah	No
EPWM_TZFLG	Trip-Zone Flag Register(1)	2Ch	No
EPWM_TZCLR	Trip-Zone Clear Register(1)	2Eh	No
EPWM_TZFRC	Trip-Zone Force Register(1)	30h	No

(1) All trip-zone registers are EALLOW protected and can be modified only after executing the EALLOW instruction.

11.5.4.7.3 Operational Highlights for the ePWM Trip-Zone Submodule

The trip-zone signals at pin $\overline{TZ0}$ to $\overline{TZ5}$ is an active-low input signal. When the pin goes low, it indicates that a trip event has occurred. Each ePWM module can be individually configured to ignore or use each of the trip-zone pins. Which trip-zone pins are used by a particular ePWM module is determined by the EPWM_TZSEL register for that specific ePWM module. The trip-zone signal may or may not be synchronized to the system clock (VBUS_CLK). A minimum of 1 VBUS_CLK low pulse on the $\overline{TZ0}$ to $\overline{TZ5}$ inputs is sufficient to trigger a fault condition in the ePWM module. The asynchronous trip makes sure that if clocks are missing for any reason, the outputs can still be tripped by a valid event present on the $\overline{TZ0}$ to $\overline{TZ5}$ inputs.

The $\overline{TZ0}$ to $\overline{TZ5}$ inputs can be individually configured to provide either a cycle-by-cycle or one-shot trip event for a ePWM module. The configuration is determined by the EPWM_TZSEL[5-0] CBCk and EPWM_TZSEL[13-8] OSHTk bits (where k = 0 to 5 corresponds to the trip pin), respectively.

- **Cycle-by-Cycle (CBC):** When a cycle-by-cycle trip event occurs, the action specified in the EPWM_TZCTL register is carried out immediately on the EPWMxA and/or EPWMxB output. Table 11-823 lists the possible actions. In addition, the cycle-by-cycle trip event flag (EPWM_TZFLG[1] CBC) is set and a EPWMxTZINT interrupt is generated if it is enabled in the EPWM_TZEINT register.

  The specified condition on the pins is automatically cleared when the ePWM time-base counter reaches zero (EPWM_TBCNT bitfield TBCNT = 0000h) if the trip event is no longer present. Therefore, in this mode, the trip event is cleared or reset every PWM cycle. The EPWM_TZFLG[1] CBC flag bit will remain set until it is manually cleared by writing to the EPWM_TZCLR[1] CBC bit. If the cycle-by-cycle trip event is still present when the EPWM_TZFLG[1] CBC bit is cleared, then it will again be immediately set.
- **One-Shot (OSHT):** When a one-shot trip event occurs, the action specified in the EPWM_TZCTL register is carried out immediately on the EPWMxA and/or EPWMxB output. Table 11-823 lists the possible actions. In addition, the one-shot trip event flag (EPWM_TZFLG[2] OST) is set and a EPWMxTZINT interrupt is generated if it is enabled in the EPWM_TZEINT register. The one-shot trip condition must be cleared manually by writing to the EPWM_TZCLR[2] OST bit.

The action taken when a trip event occurs can be configured individually for each of the ePWM output pins by way of the EPWM_TZCTL[1-0] TZA and EPWM_TZCTL[3-2] TZB register bits. One of four possible actions, shown in Table 11-823, can be taken on a trip event.

Table 11-823. Possible Actions On an ePWM Trip Event

EPWM_TZCTL[1-0] TZA and/or EPWM_TZCTL[3-2] TZB	EPWMxA and/or EPWMxB	Comment
0	High-Impedance	Tripped
1h	Force to High State	Tripped
2h	Force to Low State	Tripped
3h	No Change	Do Nothing. No change is made to the output.

Example of ePWM Trip-Zone Configurations

A one-shot trip event on $\overline{TZ0}$ pulls both EPWM1A, EPWM1B low and also forces EPWM2A and EPWM2B high.

- Configure the ePWM_1 registers as follows:
  - EPWM_TZSEL[8] OSHT1 = 1: enables $\overline{TZ}$ as a one-shot event source for ePWM_1
  - EPWM_TZCTL[1-0] TZA = 2: EPWM1A will be forced low on a trip event.
  - EPWM_TZCTL[3-2] TZB = 2: EPWM1B will be forced low on a trip event.
- Configure the ePWM_2 registers as follows:
  - EPWM_TZSEL[8] OSHT1 = 1: enables $\overline{TZ}$ as a one-shot event source for ePWM_2
  - EPWM_TZCTL[1-0] TZA = 1: EPWM2A will be forced high on a trip event.
  - EPWM_TZCTL[3-2] TZB = 1: EPWM2B will be forced high on a trip event.

11.5.4.7.4 Generating ePWM Trip-Event Interrupts

Figure 11-389 and Figure 11-390 illustrate the trip-zone submodule control and interrupt logic, respectively.

Figure 11-389. ePWM Trip-Zone Submodule Mode Control Logic

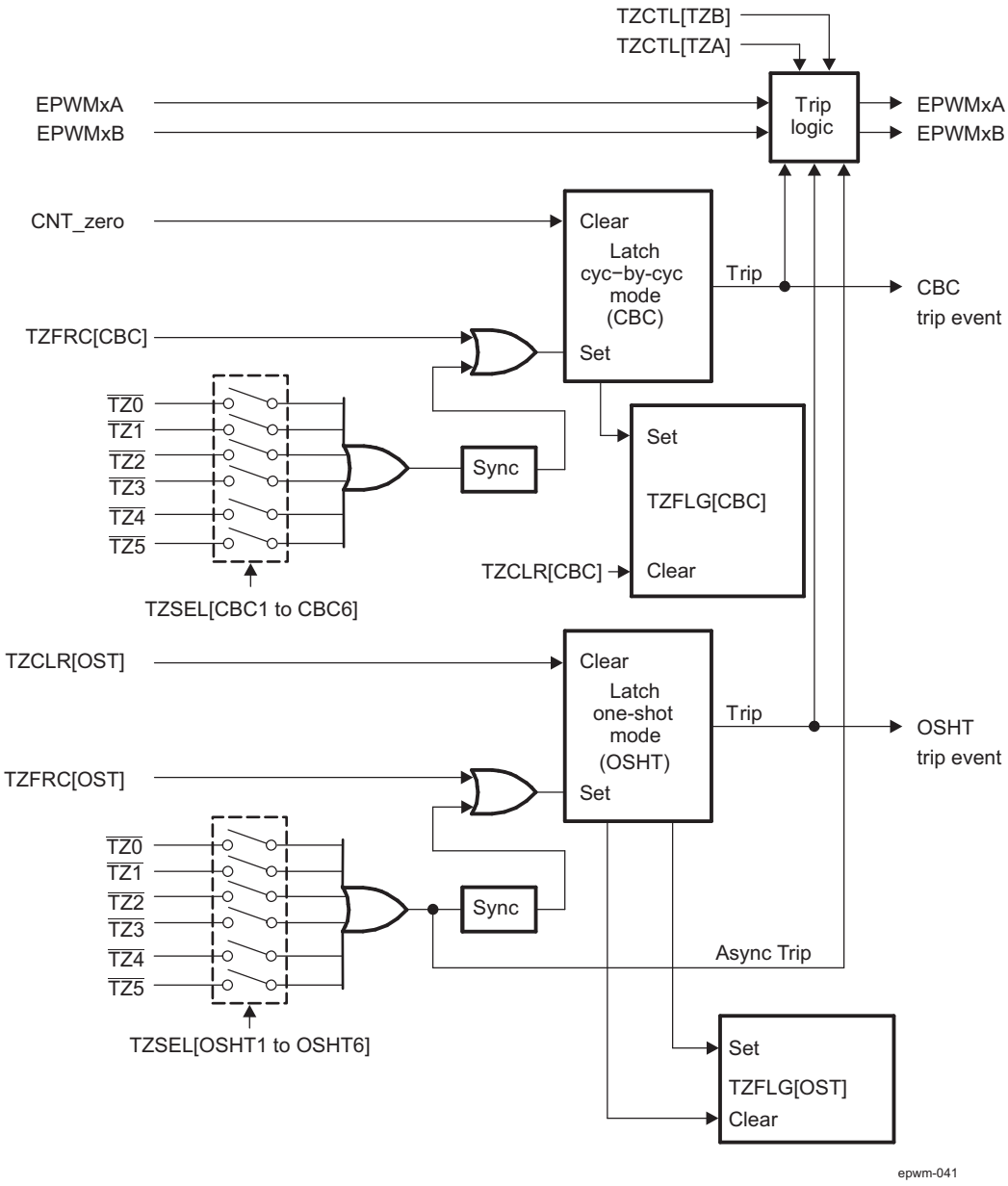
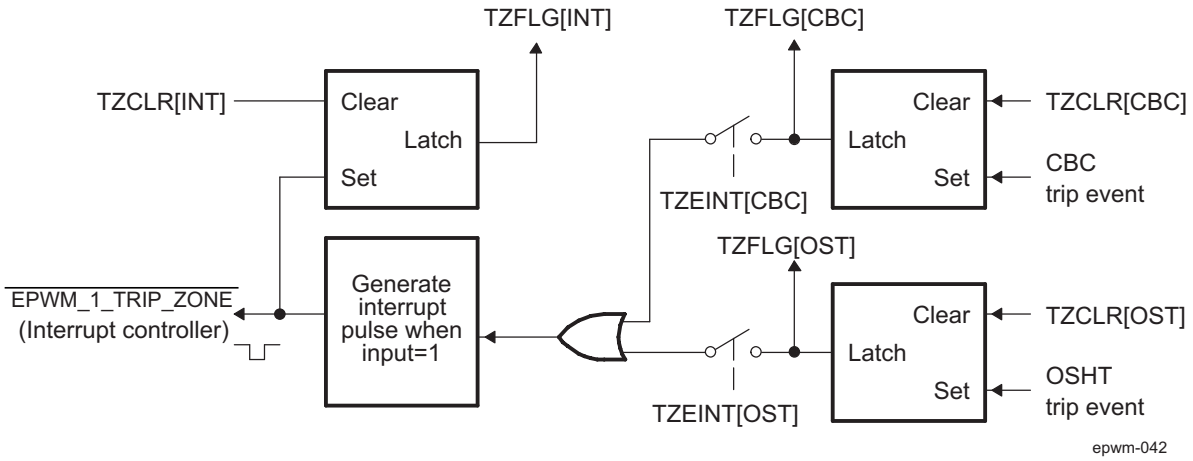


Figure 11-390. ePWM Trip-Zone Submodule Interrupt Logic



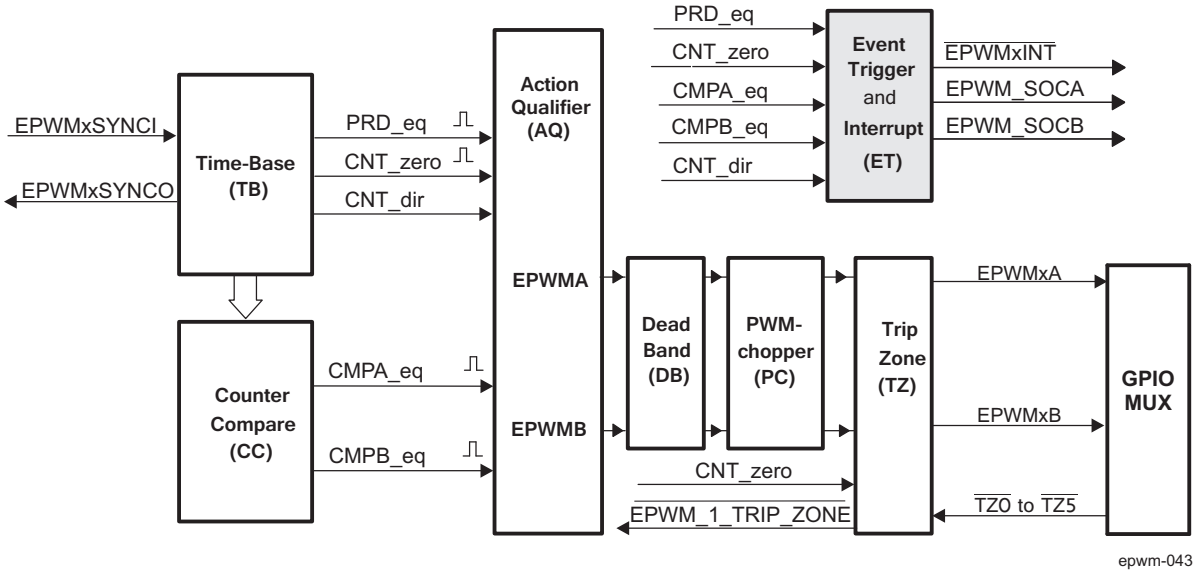
11.5.4.8 ePWM Event-Trigger (ET) Submodule

This section describes the Event-Trigger (ET) submodule in the PWM module.

11.5.4.8.1 Overview

Figure 11-391 shows the event-trigger (ET) submodule in the ePWM system. The event-trigger submodule manages the events generated by the time-base submodule and the counter-compare submodule to generate an aggregated interrupt request.

Figure 11-391. ePWM Event-Trigger Submodule



The key functions of the event-trigger submodule are:

- Receives event inputs generated by the time-base and counter-compare submodules
- Uses the time-base direction information for up/down event qualification
- Uses prescaling logic to issue interrupt requests at:
  - Every event
  - Every second event
  - Every third event.
- Provides full visibility of event generation via event counters and flags

11.5.4.8.2 Controlling and Monitoring the ePWM Event-Trigger Submodule

The key registers used to configure the event-trigger submodule are shown in Table 11-824:

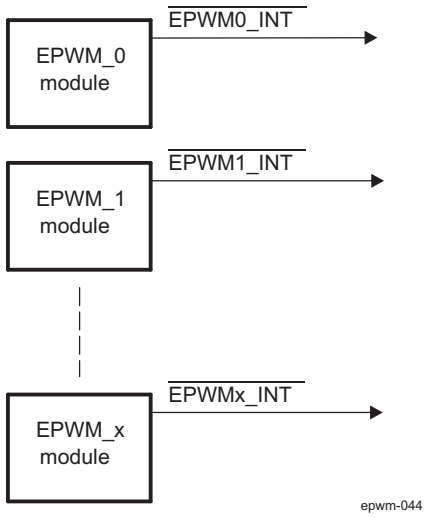
Table 11-824. ePWM Event-Trigger Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
EPWM_ETSEL	Event-Trigger Selection Register	32h	No
EPWM_ETPS	Event-Trigger Prescale Register	34h	No
EPWM_ETFLG	Event-Trigger Flag Register	36h	No
EPWM_ETCLR	Event-Trigger Clear Register	38h	No
EPWM_ETFRC	Event-Trigger Force Register	3Ah	No

11.5.4.8.3 Operational Overview of the ePWM Event-Trigger Submodule

Each ePWM module has one interrupt request line as shown in Figure 11-392. Mapping interrupt lines to device host interrupt controllers is device specific and is covered in the Section 11.6.3.

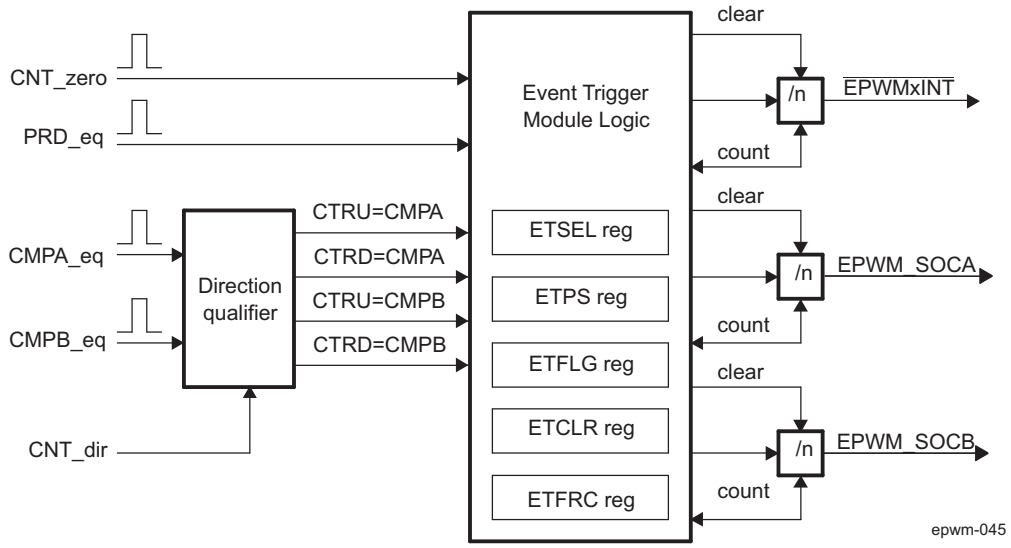
Figure 11-392. ePWM Event-Trigger Submodule Inter-Connectivity to Interrupt Controller



The event-trigger submodule monitors various event conditions (the left side inputs to event-trigger submodule shown in Figure 11-393) and can be configured to prescale these events before issuing an Interrupt request. The event-trigger prescaling logic can issue Interrupt requests at:

- Every event
- Every second event
- Every third event

Figure 11-393. ePWM Event-Trigger Submodule Showing Event Inputs and Prescaled Outputs



- ETSEL — This selects which of the possible events will trigger an interrupt.
- ETPS — This programs the event prescaling options previously mentioned.
- ETFLG — These are flag bits indicating status of the selected and prescaled events.
- ETCLR — These bits allow to clear the flag bits in the EPWM_ETFLG register via software.
- ETFRC — These bits allow software forcing of an event. Useful for debugging or software intervention.

A more detailed look at how the various register bits interact with the Interrupt is shown in Figure 11-394.

Figure 11-394 shows the event-trigger's interrupt generation logic. The interrupt-period (EPWM_ETPS[1-0] INTPRD) bits specify the number of events required to cause an interrupt pulse to be generated. The choices available are:

- Do not generate an interrupt
- Generate an interrupt on every event
- Generate an interrupt on every second event
- Generate an interrupt on every third event

An interrupt cannot be generated on every fourth or more events.

Which event can cause an interrupt is configured by the interrupt selection (EPWM_ETSEL[2-0] INTSEL) bits. The event can be one of the following:

- Time-base counter equal to zero (EPWM_TBCNT bitfield TBCNT = 0000h).
- Time-base counter equal to period (EPWM_TBCNT bitfield TBCNT = TBPRD value in EPWM_TBPRD active register).
- Time-base counter equal to the compare A register (EPWM_CMPA) when the timer is incrementing.
- Time-base counter equal to the compare A register (EPWM_CMPA) when the timer is decrementing.
- Time-base counter equal to the compare B register (EPWM_CMPB) when the timer is incrementing.
- Time-base counter equal to the compare B register (EPWM_CMPB) when the timer is decrementing.

The number of events that have occurred can be read from the interrupt event counter (EPWM_ETPS[3-2] INTCNT) register bits. That is, when the specified event occurs the EPWM_ETPS[3-2] INTCNT bits are incremented until they reach the value specified by EPWM_ETPS[1-0] INTPRD. When EPWM_ETPS[3-2] INTCNT = EPWM_ETPS[1-0] INTPRD the counter stops counting and its output is set. The counter is only cleared when an interrupt is sent to the interrupt controller.

When EPWM_ETPS[3-2] INTCNT reaches EPWM_ETPS[1-0] INTPRD, one of the following behaviors will occur:

- If interrupts are enabled, EPWM_ETSEL[3] INTEN = 1 and the interrupt flag is clear, EPWM_ETFLG[0] INT = 0, then an interrupt pulse is generated and the interrupt flag is set, EPWM_ETFLG[0] INT = 1, and the event counter is cleared EPWM_ETPS[3-2] INTCNT = 0. The counter will begin counting events again.
- If interrupts are disabled, EPWM_ETSEL[3] INTEN = 0, or the interrupt flag is set, EPWM_ETFLG[0] INT = 1, the counter stops counting events when it reaches the period value EPWM_ETPS[3-2] INTCNT = EPWM_ETPS[1-0] INTPRD.
- If interrupts are enabled, but the interrupt flag is already set, then the counter will hold its output high until the ENTFLG[0] INT flag is cleared. This allows for one interrupt to be pending while one is serviced.

Writing to the INTPRD bits will automatically clear the counter INTCNT = 0 and the counter output will be reset (so no interrupts are generated). Writing a 1 to the EPWM_ETFRC[0] INT bit will increment the event counter INTCNT. The counter will behave as described above when INTCNT = INTPRD. When INTPRD = 0, the counter is disabled and hence no events will be detected and the EPWM_ETFRC[0] INT bit is also ignored.

Figure 11-394. ePWM Event-Trigger Interrupt Generator

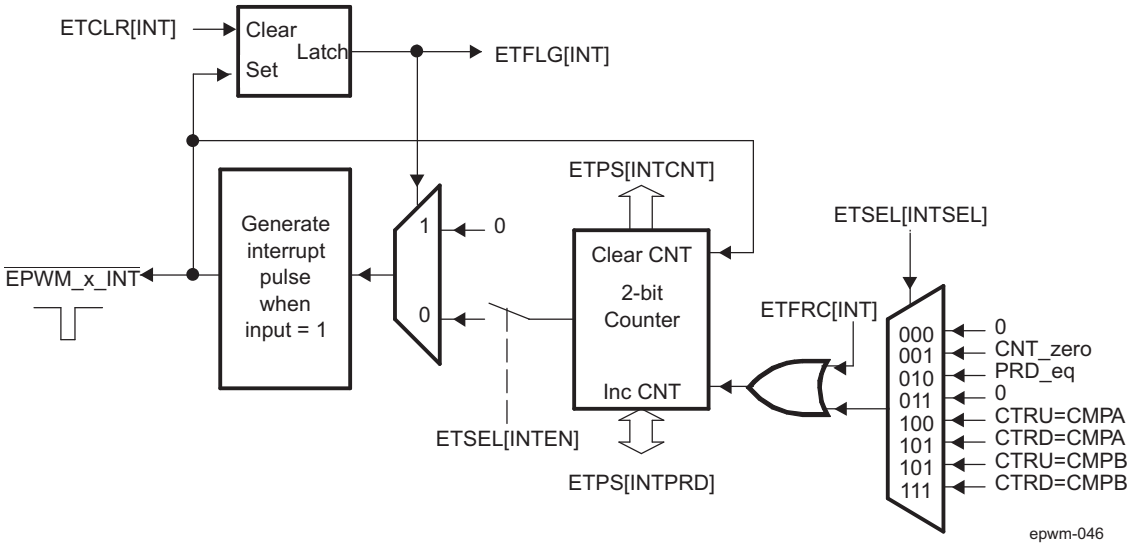


Figure 11-395 shows the operation of the event-trigger's start-of-conversion-A (SOCA) pulse generator. The EPWM_ETPS[11-10] SOCACNT counter and EPWM_ETPS[9-8] SOCAPRD period values behave similarly to the interrupt generator except that the pulses are continuously generated. That is, the pulse flag EPWM_ETFLG[2] SOCA is latched when a pulse is generated, but it does not stop further pulse generation. The enable/disable bit EPWM_ETSEL[11] SOCAEN stops pulse generation, but input events can still be counted until the period value is reached as with the interrupt generation logic. The event that will trigger an SOCA and SOCB pulse can be configured separately in the EPWM_ETSEL[10-8] SOCASEL and EPWM_ETSEL[14-12] SOCBSEL bits. The possible events are the same events that can be specified for the interrupt generation logic.

Figure 11-395. ePWM Event-Trigger SOCA Pulse Generator

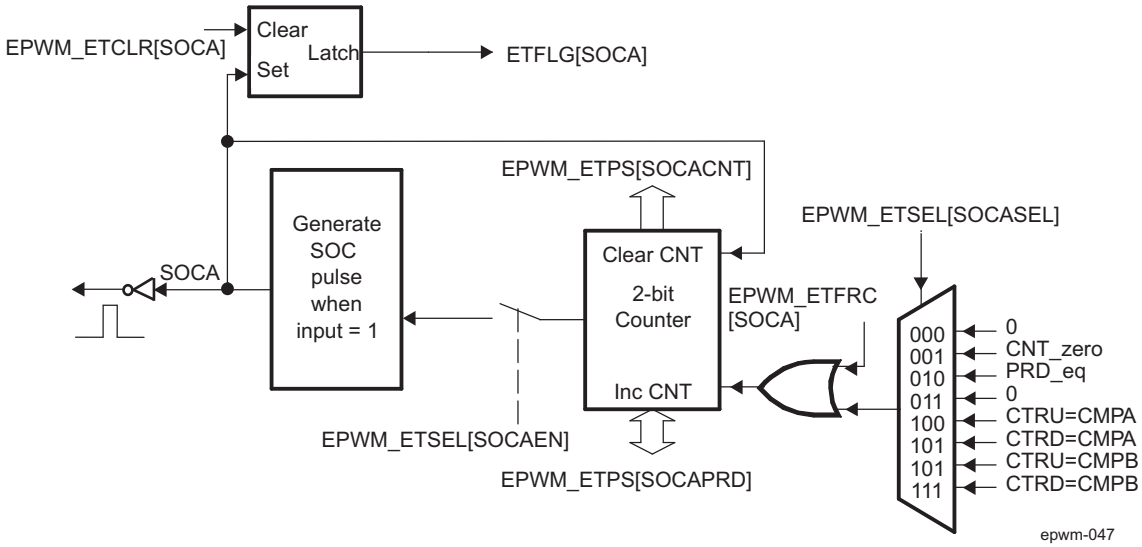
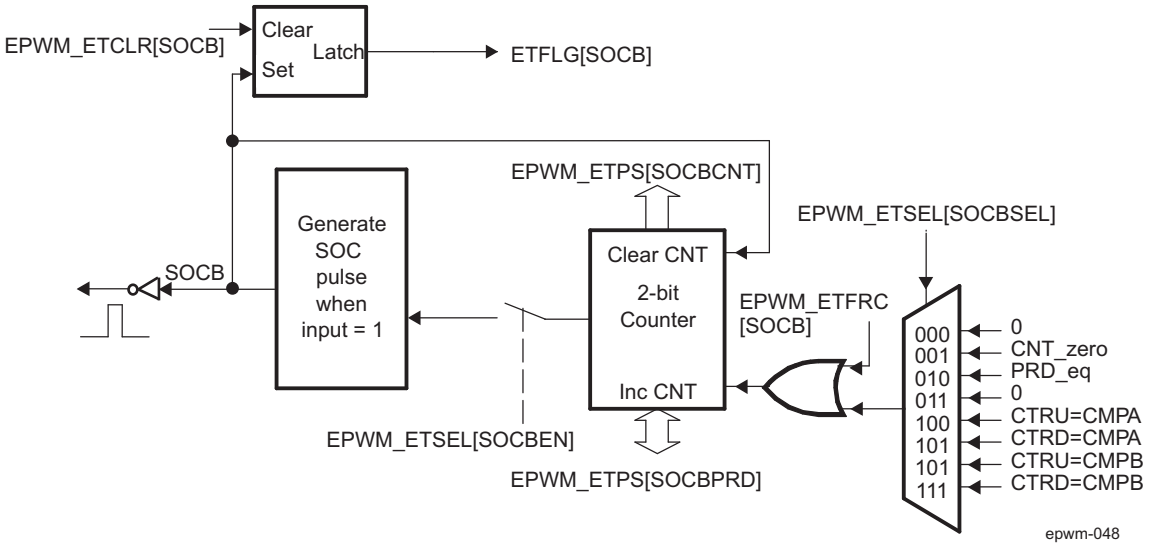


Figure 11-396 shows the operation of the event-trigger's start-of-conversion-B (SOCB) pulse generator. The event-trigger's SOCB pulse generator operates the same way as the SOCA.

Figure 11-396. ePWM Event-Trigger SOCB Pulse Generator



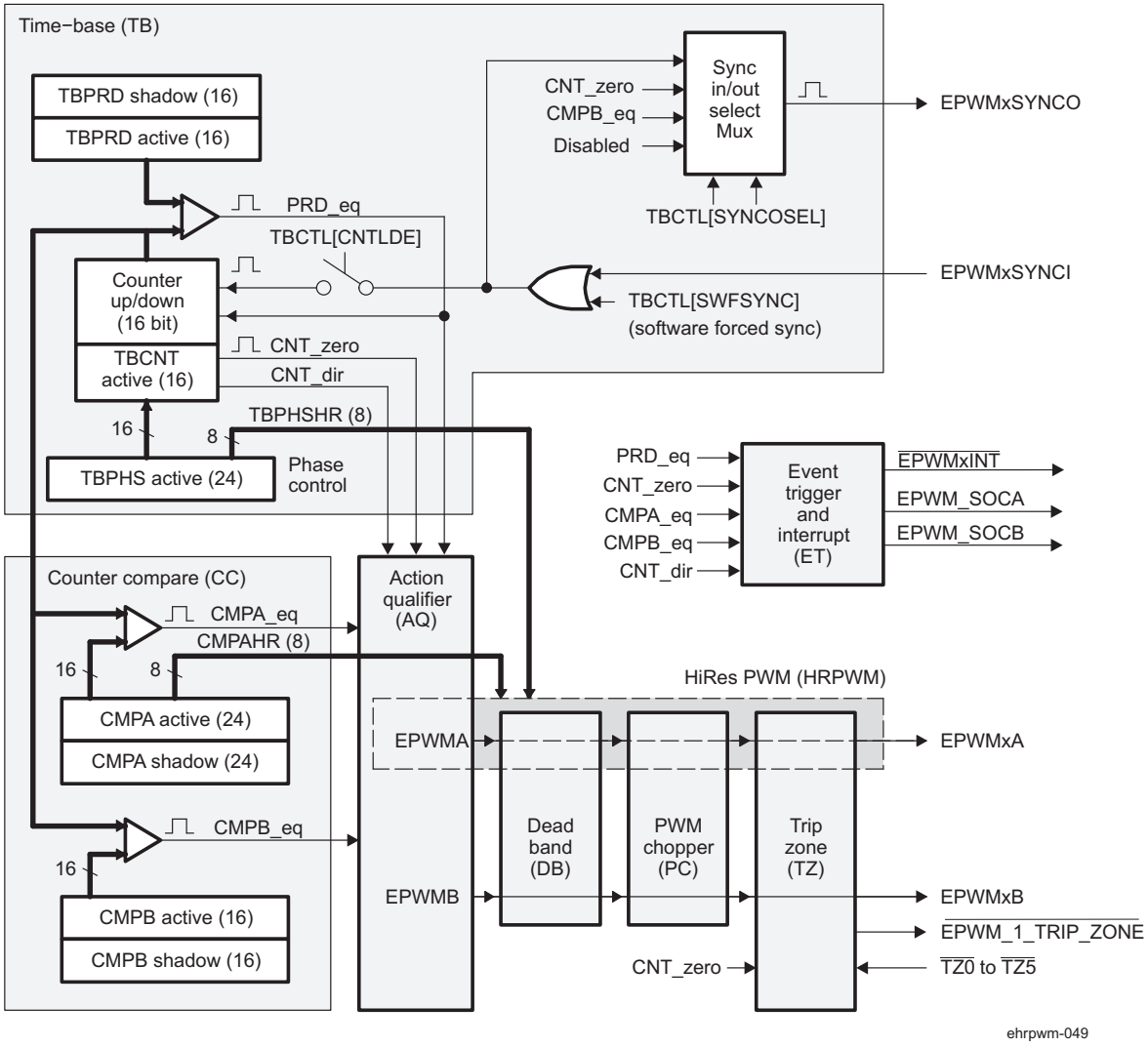
11.5.4.9 ePWM High Resolution (HRPWM) Submodule

This section describes the High-Resolution PWM (HRPWM) submodule in the PWM module.

11.5.4.9.1 Overview

Figure 11-397 shows the high-resolution PWM (HRPWM) submodule in the ePWM system. Some devices include the high-resolution PWM submodule, see Section 11.6.3 to determine which ePWM instances include this feature.

Figure 11-397. HRPWM System Interface



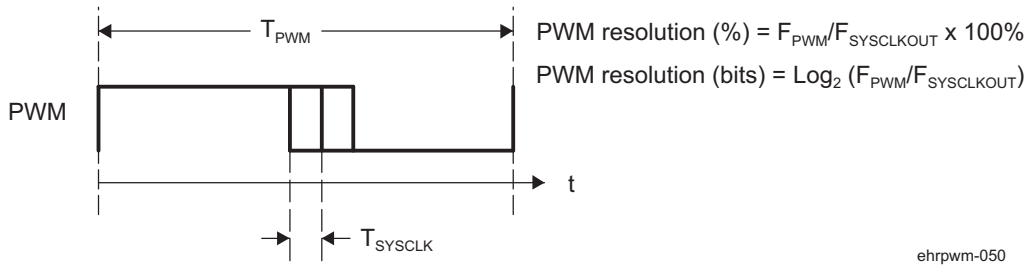
The high-resolution pulse-width modulator (HRPWM) extends the time resolution capabilities of the conventionally derived digital pulse-width modulator (PWM). HRPWM is typically used when PWM resolution falls below ~9-10 bits.
The key features of HRPWM are:

- Extended time resolution capability
- Used in both duty cycle and phase-shift control methods
- Finer time granularity control or edge positioning using extensions to the Compare A and Phase registers
- Implemented using the A signal path of PWM, that is, on the EPWMxA output. **EPWMxB output has**

conventional PWM capabilities.

The ePWM peripheral is used to perform a function that is mathematically equivalent to a digital-to-analog converter (DAC). As shown in Figure 11-398, the effective resolution for conventionally generated PWM is a function of PWM frequency (or period) and system clock frequency.

Figure 11-398. Resolution Calculations for Conventionally Generated PWM



Use HRPWMI when the required PWM operating frequency does not offer sufficient resolution in PWM mode. As an example of improved performance offered by HRPWM, Table 11-825 shows resolution in bits for various PWM frequencies. Table 11-825 values assume a MEP step size of 180 ps. See the device Data Manual for typical and maximum performance specifications for the MEP.

Table 11-825. Resolution for PWM and HRPWM

	Regular Resolution (PWM)		High Resolution (HRPWM)	
PWM Frequency (kHz)	Bits	%	Bits	%
20	12.3	0.0	18.1	0.000
50	11.0	0.0	16.8	0.001
100	10.0	0.1	15.8	0.002
150	9.4	0.2	15.2	0.003
200	9.0	0.2	14.8	0.004
250	8.6	0.3	14.4	0.005
500	7.6	0.5	13.8	0.007
1000	6.6	1.0	12.4	0.018
1500	6.1	1.5	11.9	0.027
2000	5.6	2.0	11.4	0.036

Although each application may differ, typical low-frequency PWM operation (below 250 kHz) may not require HRPWM. HRPWM capability is most useful for high-frequency PWM requirements of power conversion topologies such as:

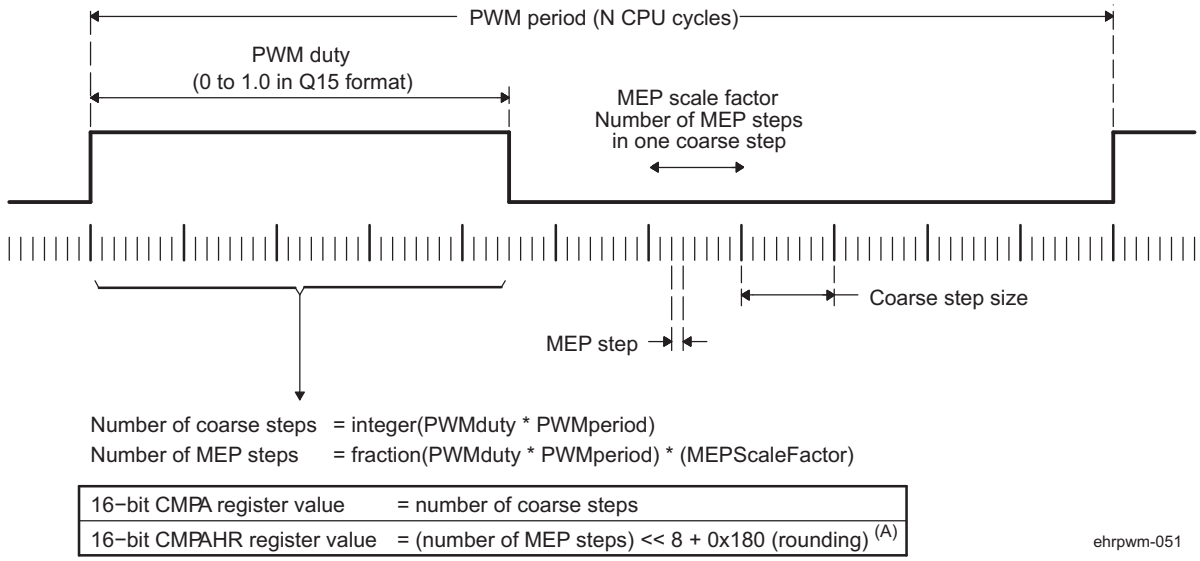
- Single-phase buck, boost, and flyback
- Multi-phase buck, boost, and flyback
- Phase-shifted full bridge
- Direct modulation of D-Class power amplifiers.

11.5.4.9.2 Architecture of the High-Resolution PWM Submodule

The HRPWM is based on micro edge positioner (MEP) technology. MEP logic is capable of positioning an edge very finely by sub-dividing one coarse system clock of a conventional PWM generator. The time step accuracy is on the order of 150 ps. The HRPWM also has a self-check software diagnostics mode to check if the MEP logic is running optimally, under all operating conditions.

Figure 11-399 shows the relationship between one coarse system clock and edge position in terms of MEP steps, which are controlled via an 8-bit field in the Compare A extension register (HRPWM_CMPAHR).

Figure 11-399. Operating Logic Using MEP



A For MEP range and rounding adjustment.

To generate an HRPWM waveform of a given frequency and polarity, the user needs to configure the TBM (Time-Base Module), CCM (Counter-Compare Module), and AQM (Action-Qualifier Module) registers. The HRPWM works together with the TBM, CCM, and AQM registers to extend edge resolution, and should be configured accordingly. Although many programming combinations are possible, only a few are needed and practical.

11.5.4.9.3 Controlling and Monitoring the High-Resolution PWM Submodule

The MEP of the HRPWM is controlled by two extension registers, each 8-bits wide. These two HRPWM registers are concatenated with the 16-bit EPWM_TBPHS and EPWM_CMPA registers used to control PWM operation.

- HRPWM_TBPHSHR — Time-Base Phase High-Resolution Register
- HRPWM_CMPAHR — Counter-Compare A High-Resolution Register.

Table 11-826 lists the registers used to control and monitor the high-resolution PWM submodule.

Table 11-826. HRPWM Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
HRPWM_TBPHSHR	Extension Register for HRPWM Phase	4h	No
HRPWM_CMPAHR	Extension Register for HRPWM Duty	10h	Yes
HRPWM_HRCTL	HRPWM Configuration Register	40h	No

11.5.4.9.4 Configuring the High-Resolution PWM Submodule

Once the ePWM has been configured to provide conventional PWM of a given frequency and polarity, the HRPWM is configured by programming the HRPWM_HRCTL register located at offset address 40h. This register provides configuration options for the following key operating modes:

- **Edge Mode:** The MEP can be programmed to provide precise position control on the rising edge (RE), falling edge (FE), or both edges (BE) at the same time. FE and RE are used for power topologies requiring duty cycle control, while BE is used for topologies requiring phase shifting, for example, phase shifted full bridge.
- **Control Mode:** The MEP is programmed to be controlled either from the HRPWM_CMPAHR register

(duty cycle control) or the HRPWM_TBPHSHR register (phase control). RE or FE control mode should be used with HRPWM_CMPAHR register. BE control mode should be used with HRPWM_TBPHSHR register.

- **Shadow Mode:** This mode provides the same shadowing (double buffering) option as in regular PWM mode. This option is valid only when operating from the HRPWM_CMPAHR register and should be chosen to be the same as the regular load option for the CMPA register. If HRPWM_TBPHSHR is used, then this option has no effect.

11.5.4.9.5 Operational Highlights for the High-Resolution PWM Submodule

The MEP logic is capable of placing an edge in one of 255 (8 bits) discrete time steps, each of which has a time resolution on the order of 150 ps. The MEP works with the TBM (Time-Base Module) and CCM (Counter-Compare Module) registers to be certain that time steps are optimally applied and that edge placement accuracy is maintained over a wide range of PWM frequencies, system clock frequencies and other operating conditions. Table 11-827 shows the typical range of operating frequencies supported by the HRPWM.

Table 11-827. Relationship Between MEP Steps, PWM Frequency and Resolution

System (MHz)	MEP Steps Per VBUS_CLK[1] [2] [3]	PWM Minimum (Hz)[4]	PWM Maximum (MHz)	Resolution at Maximum (Bits)[5]
50.0	111	763	2.50	11.1
60.0	93	916	3.00	10.9
70.0	79	1068	3.50	10.6
80.0	69	1221	4.00	10.4
90.0	62	1373	4.50	10.3
100.0	56	1526	5.00	10.1

(1) System frequency = VBUS_CLK, that is, CPU clock. TBCLK = VBUS_CLK
(2) Table data based on a MEP time resolution of 180 ps (this is an example value)
(3) MEP steps applied = T_{VBUS_CLK}/180 ps in this example.
(4) PWM minimum frequency is based on a maximum period value, TBPRD = 65 535. PWM mode is asymmetrical up-count.
(5) Resolution in bits is given for the maximum PWM frequency stated.

11.5.4.9.5.1 HRPWM Edge Positioning

In a typical power control loop (switch modes, digital motor control (DMC), uninterruptible power supply (UPS)), a digital controller (PID, 2pole/2zero, lag/lead, and so forth) issues a duty command, usually expressed in a per unit or percentage terms.

In the following example, assume that for a particular operating point, the demanded duty cycle is 0.405 or 40.5% on-time and the required converter PWM frequency is 1.25 MHz. In conventional PWM generation with a system clock of 100 MHz, the duty cycle choices are in the vicinity of 40.5%. Figure 11-400 shows that duty cycle of 40% (a compare value of 32 counts) is the closest to 40.5%. This is equivalent to an edge position of 320 ns instead of the desired 324 ns. This data is shown in Table 11-828.

Utilizing MEP allows to achieve an edge position much closer to the desired point of 324 ns. Table 11-828 shows that in addition to the CMPA value, 22 steps of the MEP (HRPWM_CMPAHR register) will position the edge at 323.96 ns, resulting in almost zero error. In this example, it is assumed that the MEP has a step resolution of 180 ns.

Figure 11-400. Required PWM Waveform for a Requested Duty = 40.5%

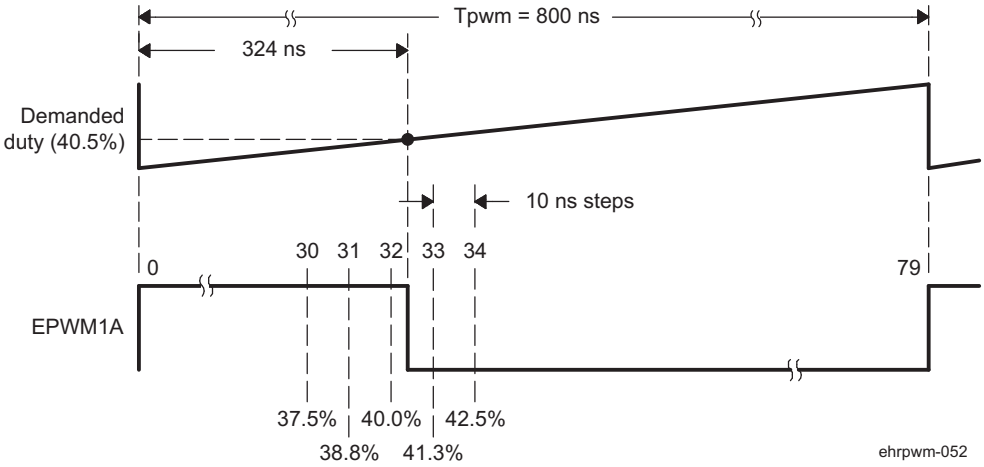


Table 11-828. CMPA vs Duty (left), and [CMPA:CMPAHR] vs Duty (right)

CMPA (count)[1] [2] [3]	DUTY (%)	High Time (ns)	CMPA (count)	CMPAHR (count)	Duty (%)	High Time (ns)
28	35.0	280	32	18	40.405	323.24
29	36.3	290	32	19	40.428	323.42
30	37.5	300	32	20	40.450	323.60
31	38.8	310	32	21	40.473	323.78
32	40.0	320	32	22	40.495	323.96
33	41.3	330	32	23	40.518	324.14
34	42.5	340	32	24	40.540	324.32
			32	25	40.563	324.50
Required			32	26	40.585	324.68
32.40	40.5	324	32	27	40.608	324.86

(1) System clock, VBUS_CLK and TBCLK = 100 MHz, 10 ns
(2) For a PWM Period register value of 80 counts, PWM Period = 80 × 10 ns = 800 ns, PWM frequency = 1/800 ns = 1.25 MHz
(3) Assumed MEP step size for the above example = 180 ps

11.5.4.9.5.2 HRPWM Scaling Considerations

The mechanics of how to position an edge precisely in time has been demonstrated using the resources of the standard (EPWM_CMPA) and MEP (HRPWM_CMPAHR) registers. In a practical application, however, it is necessary to seamlessly provide the CPU a mapping function from a per-unit (fractional) duty cycle to a final integer (non-fractional) representation that is written to the [CMPA:CMPAHR] register combination.

To do this, first examine the scaling or mapping steps involved. It is common in control software to express duty cycle in a per-unit or percentage basis. This has the advantage of performing all needed math calculations without concern for the final absolute duty cycle, expressed in clock counts or high time in ns. Furthermore, it makes the code more transportable across multiple converter types running different PWM frequencies.

To implement the mapping scheme, a two-step scaling procedure is required.

Assumptions for this example:

System clock, VBUS_CLK	=	10 ns (100 MHz)
PWM frequency	=	1.25 MHz (1/800 ns)
Required PWM duty cycle, **PWMDuty**	=	0.405 (40.5%)
PWM period in terms of coarse steps, **PWMperiod** (800 ns/10 ns)	=	80
Number of MEP steps per coarse step at 180 ps (10 ns/180 ps), **MEP_SF**	=	55
Value to keep CMPAHR within the range of 1-255 and fractional rounding constant (default value)	=	180h

Step 1: Percentage Integer Duty value conversion for EPWM_CMPA register

EPWM_CMPA register value	=	int(**PWMDuty** × **PWMperiod**); int means integer part
	=	int(0.405 × 80)
	=	int(32.4)
EPWM_CMPA register value	=	32 (20h)

Step 2: Fractional value conversion for HRPWM_CMPAHR register

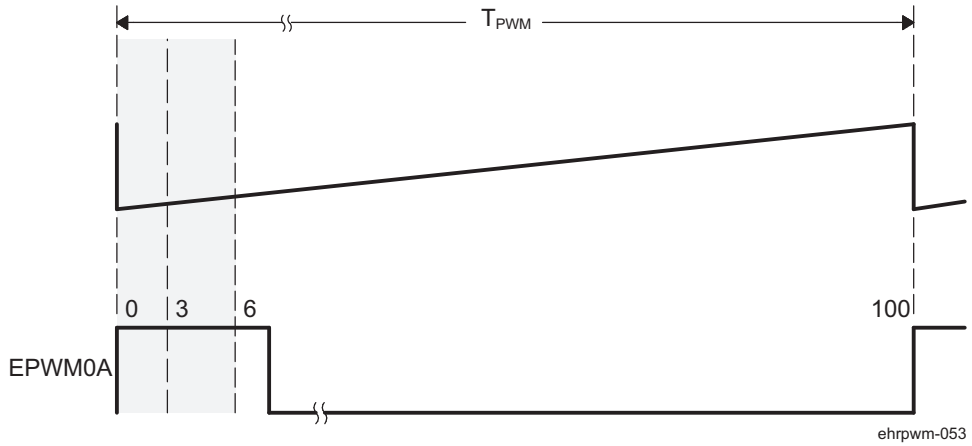
HRPWM_CMPAHR register value	=	(frac(**PWMDuty** × **PWMperiod**) × **MEP_SF**) << 8) + 180h; frac means fractional part
	=	(frac(32.4) × 55 <<8) + 180h; Shift is to move the value as CMPAHR high byte
	=	((0.4 × 55) <<8) + 180h
	=	(22 <<8) + 180h
	=	22 × 256 + 180h ; Shifting left by 8 is the same multiplying by 256.
	=	5632 + 180h
	=	1600h + 180h
HRPWM_CMPAHR value	=	1780h; HRPWM_CMPAHR value = 1700h, lower 8 bits will be ignored by hardware.

11.5.4.9.5.3 HRPWM Duty Cycle Range Limitation

In high resolution mode, the MEP is not active for 100% of the PWM period. It becomes operational 3 VBUS_CLK cycles after the period starts.

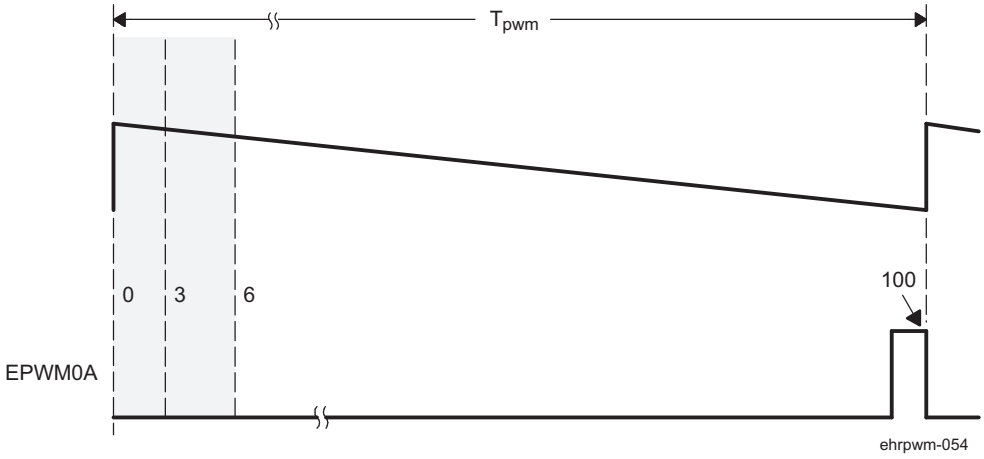
Duty cycle range limitations are illustrated in Figure 11-401. This limitation imposes a lower duty cycle limit on the MEP. For example, precision edge control is not available all the way down to 0% duty cycle. Although for the first 3 or 6 cycles, the HRPWM capabilities are not available, regular PWM duty control is still fully operational down to 0% duty. In most applications this should not be an issue as the controller regulation point is usually not designed to be close to 0% duty cycle.

Figure 11-401. Low % Duty Cycle Range Limitation Example When PWM Frequency = 1 MHz



If the application demands HRPWM operation in the low percent duty cycle region, then the HRPWM can be configured to operate in count-down mode with the rising edge position (REP) controlled by the MEP. This is illustrated in Figure 11-402. In this case low percent duty limitation is no longer an issue.

Figure 11-402. High % Duty Cycle Range Limitation Example when PWM Frequency = 1 MHz



11.5.4.10 ePWM / HRPWM Functional Register Groups

The Table 11-829 lists the groups of ePWM and the high-resolution PWM module registers according to their functionalities.

Table 11-829. ePWM / HRPWM Module Control and Status Registers Grouped by Submodule

Register Name	Offset	Size (×16)	Shadow	Register Description
Time-Base (TB) Submodule Registers				
EPWM_TBCTL	0h	1	No	Time-Base Control Register

Table 11-829. ePWM / HRPWM Module Control and Status Registers Grouped by Submodule (continued)

Register Name	Offset	Size (×16)	Shadow	Register Description
EPWM_TBSTS	2h	1	No	Time-Base Status Register
EPWM_TBPHS	6h	1	No	Time-Base Phase Register
EPWM_TBCNT	8h	1	No	Time-Base Counter Register
EPWM_TBPRD	Ah	1	Yes	Time-Base Period Register
Counter-Compare (CC) Submodule Registers				
EPWM_CMPCTL	Eh	1	No	Counter-Compare Control Register
EPWM_CMPA	12h	1	Yes	Counter-Compare A Register
EPWM_CMPB	14h	1	Yes	Counter-Compare B Register
Action-Qualifier (AQ) Submodule Registers				
EPWM_AQCTLA	16h	1	No	Action-Qualifier Control Register for Output A (EPWMxA)
EPWM_AQCTLB	18h	1	No	Action-Qualifier Control Register for Output B (EPWMxB)
EPWM_AQSFRC	1Ah	1	No	Action-Qualifier Software Force Register
EPWM_AQCSFRC	1Ch	1	Yes	Action-Qualifier Continuous S/W Force Register Set
Dead-Band (DB) Generator Submodule Registers				
EPWM_DBCTL	1Eh	1	No	Dead-Band Generator Control Register
EPWM_DBRED	20h	1	No	Dead-Band Generator Rising Edge Delay Count Register
EPWM_DBFED	22h	1	No	Dead-Band Generator Falling Edge Delay Count Register
Trip-Zone (TZ) Submodule Registers				
EPWM_TZSEL	24h	1	No	Trip-Zone Select Register
EPWM_TZCTL	28h	1	No	Trip-Zone Control Register (1)
EPWM_TZEINT	2Ah	1	No	Trip-Zone Enable Interrupt Register (1)
EPWM_TZFLG	2Ch	1	No	Trip-Zone Flag Register (1)
EPWM_TZCLR	2Eh	1	No	Trip-Zone Clear Register (1)
EPWM_TZFRC	30h	1	No	Trip-Zone Force Register (1)
Event-Trigger (ET) Submodule Registers				
EPWM_ETSEL	32h	1	No	Event-Trigger Selection Register
EPWM_ETPS	34h	1	No	Event-Trigger Pre-Scale Register
EPWM_ETFLG	36h	1	No	Event-Trigger Flag Register
EPWM_ETCLR	38h	1	No	Event-Trigger Clear Register
EPWM_ETFRC	3Ah	1	No	Event-Trigger Force Register
PWM-Chopper Submodule Registers				
EPWM_PCCTL	3Ch	1	No	PWM-Chopper Control Register
High-Resolution PWM (HRPWM) Submodule Registers				
HRPWM_TBPHSHR	4h	1	No	Extension for HRPWM Phase Register
HRPWM_CMPAHR	10h	1	No	Extension for HRPWM Counter-Compare A Register
HRPWM_HRCTL	40h	1	No	HRPWM Control Register

(1) EALLOW protected registers.

11.5.4.11 Proper ePWM Interrupt Initialization Procedure

When the ePWM peripheral clock is enabled it is possible that interrupt flags may be set due to spurious events as the ePWM registers not being properly initialized. The proper procedure for initializing the ePWM peripheral is:

1. Disable global interrupts (CPU INTM flag).
2. Disable ePWM interrupts.
3. Initialize peripheral registers.
4. Clear any spurious ePWM flags.
5. Enable ePWM interrupts.
6. Enable global interrupts.

11.5.5 ePWM Registers

Table 11-831 lists the memory-mapped registers for the ePWM. All register offset addresses not listed in Table 11-831 should be considered as reserved locations and the register contents should not be modified.

This section describes the ePWM instances registers. This section describes the ePWM instances registers.

Table 11-830. ePWM Instances

Instance	Base Address
EPWM_0	0021D 0000h
EPWM_1	021D 0400h
EPWM_2	021D 0800h
EPWM_3	021D 0C00h
EPWM_4	021D 1000h
EPWM_5	021D 1400h

Table 11-831. ePWM Registers

Offset	Acronym	Register Name	EPWM_0 Physical Address	EPWM_1 Physical Address	EPWM_2 Physical Address	Section
0h	EPWM_TBCTL	Time-Base Control Register	021D 0000h	021D 0400h	021D 0800h	Section 11.5.5.1
2h	EPWM_TBSTS	Time-Base Status Register	021D 0002h	021D 0402h	021D 0802h	Section 11.5.5.2
4h	HRPWM_TBPHSHR	Time-Base Phase High-Resolution PWM Register	021D 0004h	021D 0404h	021D 0804h	Section 11.5.5.3
6h	EPWM_TBPHS	Time-Base Counter Phase Register	021D 0006h	021D 0406h	021D 0806h	Section 11.5.5.4
8h	EPWM_TBCNT	Time-Base Counter Register	021D 0008h	021D 0408h	021D 0808h	Section 11.5.5.5
Ah	EPWM_TBPRD	Time-Base Period Register	021D 000Ah	021D 040Ah	021D 080Ah	Section 11.5.5.6
Eh	EPWM_CMPCTL	Counter-Compare Control Register	021D 000Eh	021D 040Eh	021D 080Eh	Section 11.5.5.7
10h	HRPWM_CMPAHR	Counter-Compare A High-Resolution Register	021D 0010h	021D 0410h	021D 0810h	Section 11.5.5.8
12h	EPWM_CMPA	Counter-Compare A Register	021D 0012h	021D 0412h	021D 0812h	Section 11.5.5.9
14h	EPWM_CMPB	Counter-Compare B Register	021D 0014h	021D 0414h	021D 0814h	Section 11.5.5.10
16h	EPWM_AQCTLA	Action Qualifier Control Register for Output A	021D 0016h	021D 0416h	021D 0816h	Section 11.5.5.11
18h	EPWM_AQCTLB	Action Qualifier Control Register for Output B	021D 0018h	021D 0418h	021D 0818h	Section 11.5.5.12
1Ah	EPWM_AQSFRC	Action Qualifier Software Force Register	021D 001Ah	021D 041Ah	021D 081Ah	Section 11.5.5.13

Table 11-831. ePWM Registers (continued)

Offset	Acronym	Register Name	EPWM_0 Physical Address	EPWM_1 Physical Address	EPWM_2 Physical Address	Section
1Ch	EPWM_AQCSFRC	Action Qualifier Continuous Software Force Register	021D 001Ch	021D 041Ch	021D 081Ch	Section 11.5.5.14
1Eh	EPWM_DBCTL	Dead Band Generator Control Register	021D 001Eh	021D 041Eh	021D 081Eh	Section 11.5.5.15
20h	EPWM_DBRED	Dead Band Generator Rising Edge Delay Count Register	021D 0020h	021D 0420h	021D 0820h	Section 11.5.5.16
22h	EPWM_DBFED	Dead Band Generator Falling Edge Delay Count Register	021D 0022h	021D 0422h	021D 0822h	Section 11.5.5.17
24h	EPWM_TZSEL	Trip-zone Select Register	021D 0024h	021D 0424h	021D 0824h	Section 11.5.5.18
28h	EPWM_TZCTL	Trip-zone Control Register	021D 0028h	021D 0428h	021D 0828h	Section 11.5.5.19
2Ah	EPWM_TZEINT	Trip-zone Enable Interrupt Register	021D 002Ah	021D 042Ah	021D 082Ah	Section 11.5.5.20
2Ch	EPWM_TZFLG	Trip-zone Flag Register	021D 002Ch	021D 042Ch	021D 082Ch	Section 11.5.5.21
2Eh	EPWM_TZCLR	Trip-zone Clear Register	021D 002Eh	021D 042Eh	021D 082Eh	Section 11.5.5.22
30h	EPWM_TZFRC	Trip-zone Force Register	021D 0030h	021D 0430h	021D 0830h	Section 11.5.5.23
32h	EPWM_ETSEL	Event Trigger Selection Register	021D 0032h	021D 0432h	021D 0832h	Section 11.5.5.24
34h	EPWM_ETPS	Event Trigger Pre-Scale Register	021D 0034h	021D 0434h	021D 0834h	Section 11.5.5.25
36h	EPWM_ETFLG	Event Trigger Flag Register	021D 0036h	021D 0436h	021D 0836h	Section 11.5.5.26
38h	EPWM_ETCLR	Event Trigger Clear Register	021D 0038h	021D 0438h	021D 0838h	Section 11.5.5.27
3Ah	EPWM_ETFRC	Event Trigger Force Register	021D 003Ah	021D 043Ah	021D 083Ah	Section 11.5.5.28
3Ch	EPWM_PCCTL	PWM Chopper Control Register	021D 003Ch	021D 043Ch	021D 083Ch	Section 11.5.5.29
40h	HRPWM_HRCTL	High-Resolution Control Register	021D 0040h	021D 0440h	021D 0840h	Section 11.5.5.30

Table 11-832. ePWM Registers

Offset	Acronym	Register Name	EPWM_3 Physical Address	EPWM_4 Physical Address	EPWM_5 Physical Address	Section
0h	EPWM_TBCTL	Time-Base Control Register	021D 0C00h	021D 1000h	021D 1400h	Section 11.5.5.1
2h	EPWM_TBSTS	Time-Base Status Register	021D 0C02h	021D 1002h	021D 1402h	Section 11.5.5.2
4h	HRPWM_TBPHSHR	Time-Base Phase High-Resolution PWM Register	021D 0C04h	021D 1004h	021D 1404h	Section 11.5.5.3
6h	EPWM_TBPHS	Time-Base Counter Phase Register	021D 0C06h	021D 1006h	021D 1406h	Section 11.5.5.4
8h	EPWM_TBCNT	Time-Base Counter Register	021D 0C08h	021D 1008h	021D 1408h	Section 11.5.5.5
Ah	EPWM_TBPRD	Time-Base Period Register	021D 0C0Ah	021D 100Ah	021D 140Ah	Section 11.5.5.6
Eh	EPWM_CMPCTL	Counter-Compare Control Register	021D 0C0Eh	021D 100Eh	021D 140Eh	Section 11.5.5.7

Table 11-832. ePWM Registers (continued)

Offset	Acronym	Register Name	EPWM_3 Physical Address	EPWM_4 Physical Address	EPWM_5 Physical Address	Section
10h	HRPWM_CMPAHR	Counter-Compare A High-Resolution Register	021D 0C10h	021D 1010h	021D 1410h	Section 11.5.5.8
12h	EPWM_CMPA	Counter-Compare A Register	021D 0C12h	021D 1012h	021D 1412h	Section 11.5.5.9
14h	EPWM_CMPB	Counter-Compare B Register	021D 0C14h	021D 1014h	021D 1414h	Section 11.5.5.10
16h	EPWM_AQCTLA	Action Qualifier Control Register for Output A	021D 0C16h	021D 1016h	021D 1416h	Section 11.5.5.11
18h	EPWM_AQCTLB	Action Qualifier Control Register for Output B	021D 0C18h	021D 1018h	021D 1418h	Section 11.5.5.12
1Ah	EPWM_AQSFRC	Action Qualifier Software Force Register	021D 0C1Ah	021D 101Ah	021D 141Ah	Section 11.5.5.13
1Ch	EPWM_AQCSFRC	Action Qualifier Continuous Software Force Register	021D 0C1Ch	021D 101Ch	021D 141Ch	Section 11.5.5.14
1Eh	EPWM_DBCTL	Dead Band Generator Control Register	021D 0C1Eh	021D 101Eh	021D 141Eh	Section 11.5.5.15
20h	EPWM_DBRED	Dead Band Generator Rising Edge Delay Count Register	021D 0C20h	021D 1020h	021D 1420h	Section 11.5.5.16
22h	EPWM_DBFED	Dead Band Generator Falling Edge Delay Count Register	021D 0C22h	021D 1022h	021D 1422h	Section 11.5.5.17
24h	EPWM_TZSEL	Trip-zone Select Register	021D 0C24h	021D 1024h	021D 1424h	Section 11.5.5.18
28h	EPWM_TZCTL	Trip-zone Control Register	021D 0C28h	021D 1028h	021D 1428h	Section 11.5.5.19
2Ah	EPWM_TZEINT	Trip-zone Enable Interrupt Register	021D 0C2Ah	021D 102Ah	021D 142Ah	Section 11.5.5.20
2Ch	EPWM_TZFLG	Trip-zone Flag Register	021D 0C2Ch	021D 102Ch	021D 142Ch	Section 11.5.5.21
2Eh	EPWM_TZCLR	Trip-zone Clear Register	021D 0C2Eh	021D 102Eh	021D 142Eh	Section 11.5.5.22
30h	EPWM_TZFRC	Trip-zone Force Register	021D 0C30h	021D 1030h	021D 1430h	Section 11.5.5.23
32h	EPWM_ETSEL	Event Trigger Selection Register	021D 0C32h	021D 1032h	021D 1432h	Section 11.5.5.24
34h	EPWM_ETPS	Event Trigger Pre-Scale Register	021D 0C34h	021D 1034h	021D 1434h	Section 11.5.5.25
36h	EPWM_ETFLG	Event Trigger Flag Register	021D 0C36h	021D 1036h	021D 1436h	Section 11.5.5.26
38h	EPWM_ETCLR	Event Trigger Clear Register	021D 0C38h	021D 1038h	021D 1438h	Section 11.5.5.27
3Ah	EPWM_ETFRC	Event Trigger Force Register	021D 0C3Ah	021D 103Ah	021D 143Ah	Section 11.5.5.28
3Ch	EPWM_PCCTL	PWM Chopper Control Register	021D 0C3Ch	021D 103Ch	021D 143Ch	Section 11.5.5.29
40h	HRPWM_HRCTL	High-Resolution Control Register	021D 0C40h	021D 1040h	021D 1440h	Section 11.5.5.30

11.5.5.1 EPWM_TBCTL Register (Offset = 0h) [reset = 0h]

EPWM_TBCTL is shown in Figure 11-403 and described in Table 11-834.

Table 11-833. EPWM_TBCTL Instances

Instance	Physical Address
EPWM_0	021D 0000h
EPWM_1	021D 0400h
EPWM_2	021D 0800h
EPWM_3	021D 0C00h
EPWM_4	021D 1000h
EPWM_5	021D 1400h

Figure 11-403. EPWM_TBCTL Register

15	14	13	12	11	10	9	8
FREE_SOFT		PHSDIR	CLKDIV			HSPCLKDIV	
R/W-0h		R/W-0h	R/W-0h			R/W-0h	
7	6	5	4	3	2	1	0
HSPCLKDIV	SWFSYNC	SYNCOSEL		PRDLD	PHSEN	CTRMODE	
R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-834. EPWM_TBCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	FREE_SOFT	R/W	0h	Emulation Mode Bits. These bits select the behavior of the ePWM time-base counter during emulation events: 0h = Stop after the next time-base counter increment or decrement 1h = Stop when counter completes a whole cycle. (a) Up-count mode: stop when the time-base counter = period (EPWM_TBCNT bitfield TBCNT = TBPRD in EPWM_TBPRD active register). (b) Down-count mode: stop when the time-base counter = 0000 (EPWM_TBCNT bitfield TBCNT = 0000h). (c) Up-down-count mode: stop when the time-base counter = 0000 (EPWM_TBCNT bitfield TBCNT = 0000h). 2h = Free run 3h = Free run
13	PHSDIR	R/W	0h	Phase Direction Bit. This bit is only used when the time-base counter is configured in the up-down-count mode. The PHSDIR bit indicates the direction the time-base counter (EPWM_TBCNT) will count after a synchronization event occurs and a new phase value is loaded from the phase (EPWM_TBPHS) register. This is irrespective of the direction of the counter before the synchronization event. In the up-count and down-count modes this bit is ignored. 0h = Count down after the synchronization event. 1h = Count up after the synchronization event.

Table 11-834. EPWM_TBCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12-10	CLKDIV	R/W	0h	Time-base Clock Prescale Bits. These bits determine part of the time-base clock prescale value. TBCLK = SYSCLKOUT / (HSPCLKDIV x CLKDIV) 0h = /1 (default on reset) 1h = /2 2h = /4 3h = /8 4h = /16 5h = /32 6h = /64 7h = /128
9-7	HSPCLKDIV	R/W	0h	High-Speed Time-base Clock Prescale Bits. These bits determine part of the time-base clock prescale value. TBCLK = SYSCLKOUT / (HSPCLKDIV x CLKDIV). This divisor emulates the HSPCLK in the TMS320x281x system as used on the Event Manager (EV) peripheral. 0h = /1 1h = /2 (default on reset) 2h = /4 3h = /6 4h = /8 5h = /10 6h = /12 7h = /14
6	SWFSYNC	R/W	0h	Software Forced Synchronization Pulse. 0h = Writing a 0 has no effect and reads always return a 0. 1h = Writing a 1 forces a one-time synchronization pulse to be generated. This event is ORed with the EPWMxSYNCI input of the ePWM module. SWFSYNC is valid (operates) only when EPWMxSYNCI is selected by SYNCOSEL = 00.
5-4	SYNCOSEL	R/W	0h	Synchronization Output Select. These bits select the source of the EPWMxSYNCO signal. 0h = EPWMxSYNC: 1h = TBCNT = 0: Time-base counter equal to zero (EPWM_TBCNT bitfield TBCNT= 0000h) 2h = TBCNT = CMPB: Time-base counter equal to counter-compare B (EPWM_TBCNT bitfield TBCNT = CMPB bitfield in EPWM_CMPB) 3h = Disable EPWMxSYNCO signal
3	PRDLD	R/W	0h	Active Period Register Load From Shadow Register Select 0h = The period register (EPWM_TBPRD) is loaded from its shadow register when the time-base counter, TBCNT, is equal to zero. A write or read to the EPWM_TBPRD register accesses the shadow register. 1h = Load the EPWM_TBPRD register immediately without using a shadow register. A write or read to the EPWM_TBPRD register directly accesses the active register.
2	PHSEN	R/W	0h	Counter Register Load From Phase Register Enable 0h = Do not load the time-base counter (EPWM_TBCNT) from the time-base phase register (EPWM_TBPHS) 1h = Load the time-base counter with the phase register when an EPWMxSYNCI input signal occurs or when a software synchronization is forced by the SWFSYNC bit.

Table 11-834. EPWM_TBCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	CTRMODE	R/W	0h	Counter Mode. The time-base counter mode is normally configured once and not changed during normal operation. If you change the mode of the counter, the change will take effect at the next TBCLK edge and the current counter value shall increment or decrement from the value before the mode change. These bits set the time-base counter mode of operation as follows: 0h = Up-count mode 1h = Down-count mode 2h = Up-down-count mode 3h = Stop-freeze counter operation (default on reset)

Table 11-835. Register Call Summary for EPWM_TBCTL

ePWM Integration • Daisy-Chain Connectivity between ePWM Modules: [0]
ePWM Registers • EPWM_TBSTS Register (Offset = 2h) [reset = 0h]: [0] • EPWM_TBPRD Register (Offset = Ah) [reset = 0h]: [0][1][2] • EPWM_TBPHS Register (Offset = 6h) [reset = 0h]: [0][1] • EPWM_TBCTL Register (Offset = 0h) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • ePWM Action-Qualifier (AQ) Submodule: [0][1][2][3][4][5] • Calculating PWM Period and Frequency: [0] • ePWM Time-Base Period Shadow Register: [0][1][2] • ePWM Time-Base Counter Synchronization: [0][1][2][3] • Phase Locking the Time-Base Clocks of Multiple ePWM Modules: [0] • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Time-Base Submodule: [0]

11.5.5.2 EPWM_TBSTS Register (Offset = 2h) [reset = 0h]

EPWM_TBSTS is shown in Figure 11-404 and described in Table 11-837.

Table 11-836. EPWM_TBSTS Instances

Instance	Physical Address
EPWM_0	021D 0002h
EPWM_1	021D 0402h
EPWM_2	021D 0802h
EPWM_3	021D 0C02h
EPWM_4	021D 1002h
EPWM_5	021D 1402h

Figure 11-404. EPWM_TBSTS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					CTRMAX	SYNCI	CTRDIR
R-0h					R/W1C-0h	R/W1C-0h	R-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-837. EPWM_TBSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	Reserved
2	CTRMAX	R/W1C	0h	Time-Base Counter Max Latched Status Bit. 0h = Reading a 0 indicates the time-base counter never reached its maximum value. Writing a 0 will have no effect. 1h = Reading a 1 on this bit indicates that the time-base counter reached the max value 0xFFFF. Writing a 1 to this bit will clear the latched event.
1	SYNCI	R/W1C	0h	Input Synchronization Latched Status Bit. 0h = Writing a 0 will have no effect. Reading a 0 indicates no external synchronization event has occurred. 1h = Reading a 1 on this bit indicates that an external synchronization event has occurred (EPWMxSYNCI). Writing a 1 to this bit will clear the latched event.
0	CTRDIR	R	0h	Time-Base Counter Direction Status Bit. At reset, the counter is frozen, therefore, this bit has no meaning. To make this bit meaningful, you must first set the appropriate mode via EPWM_TBCTL[1-0] CTRMODE. 0h = Time-Base Counter is currently counting down. 1h = Time-Base Counter is currently counting up.

Table 11-838. Register Call Summary for EPWM_TBSTS

ePWM Registers • EPWM_TBSTS Register (Offset = 2h) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Time-Base Submodule: [0]

11.5.5.3 HRPWM_TBPHSHR Register (Offset = 4h) [reset = 0h]

HRPWM_TBPHSHR is shown in Figure 11-405 and described in Table 11-840.

Table 11-839. HRPWM_TBPHSHR Instances

Instance	Physical Address
EPWM_0	021D 0004h
EPWM_1	021D 0404h
EPWM_2	021D 0804h
EPWM_3	021D 0C04h
EPWM_4	021D 1004h
EPWM_5	021D 1404h

Figure 11-405. HRPWM_TBPHSHR Register

15	14	13	12	11	10	9	8
TBPHSH							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-840. HRPWM_TBPHSHR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	TBPHSH	R/W	0h	Time-base phase high-resolution bits
7-0	RESERVED	R	0h	Reserved

Table 11-841. Register Call Summary for HRPWM_TBPHSHR

ePWM Registers • HRPWM_TBPHSHR Register (Offset = 4h) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the High-Resolution PWM Submodule: [0][1] • Controlling and Monitoring the ePWM Time-Base Submodule: [0] • Configuring the High-Resolution PWM Submodule: [0][1][2]

11.5.5.4 EPWM_TBPHS Register (Offset = 6h) [reset = 0h]

EPWM_TBPHS is shown in Figure 11-406 and described in Table 11-843.

Table 11-842. EPWM_TBPHS Instances

Instance	Physical Address
EPWM_0	021D 0006h
EPWM_1	021D 0406h
EPWM_2	021D 0806h
EPWM_3	021D 0C06h
EPWM_4	021D 1006h
EPWM_5	021D 1406h

Figure 11-406. EPWM_TBPHS Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBPHS															
R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-843. EPWM_TBPHS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TBPHS	R/W	0h	These bits set time-base counter phase of the selected ePWM relative to the time-base that is supplying the synchronization input signal. (a) If EPWM_TBCTL[2] PHSEN = 0, then the synchronization event is ignored and the time-base counter is not loaded with the phase. (b) If EPWM_TBCTL[2] PHSEN = 1, then the time-base counter (TBCNT) will be loaded with the phase (TBPHS) when a synchronization event occurs. The synchronization event can be initiated by the input synchronization signal (EPWMxSYNCI) or by a software forced synchronization.

Table 11-844. Register Call Summary for EPWM_TBPHS

ePWM Registers • EPWM_TBPHS Register (Offset = 6h) [reset = 0h]: [0] • EPWM_TBCTL Register (Offset = 0h) [reset = 0h]: [0][1] • ePWM Registers: [0][1]
ePWM Functional Description • ePWM Action-Qualifier (AQ) Submodule: [0][1][2][3][4][5] • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the High-Resolution PWM Submodule: [0] • Controlling and Monitoring the ePWM Time-Base Submodule: [0] • ePWM Time-Base Counter Synchronization: [0][1]

11.5.5.5 EPWM_TBCNT Register (Offset = 8h) [reset = 0h]

EPWM_TBCNT is shown in Figure 11-407 and described in Table 11-846.

Table 11-845. EPWM_TBCNT Instances

Instance	Physical Address
EPWM_0	021D 0008h
EPWM_1	021D 0408h
EPWM_2	021D 0808h
EPWM_3	021D 0C08h
EPWM_4	021D 1008h
EPWM_5	021D 1408h

Figure 11-407. EPWM_TBCNT Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBCNT															
R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-846. EPWM_TBCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TBCNT	R/W	0h	Reading these bits gives the current time-base counter value. Writing to these bits sets the current time-base counter value. The update happens as soon as the write occurs. The write is NOT synchronized to the time-base clock (TBCLK) and the register is not shadowed.

Table 11-847. Register Call Summary for EPWM_TBCNT

ePWM Registers • EPWM_TZFLG Register (Offset = 2Ch) [reset = 0h]: [0] • EPWM_TBCNT Register (Offset = 8h) [reset = 0h]: [0] • ePWM Registers: [0][1] • EPWM_CMPB Register (Offset = 14h) [reset = 0h]: [0] • EPWM_CMPA Register (Offset = 12h) [reset = 0h]: [0] • EPWM_TBCTL Register (Offset = 0h) [reset = 0h]: [0][1][2][3][4][5][6] • EPWM_CMPCTL Register (Offset = Eh) [reset = 0h]: [0][1][2][3]
ePWM Functional Description • ePWM Time-Base (TB) Submodule: [0][1] • Operational Highlights for the ePWM Counter-Compare Submodule: [0][1] • ePWM Action-Qualifier (AQ) Submodule: [0][1][2][3][4][5] • ePWM Time-Base Period Shadow Register: [0] • ePWM Time-Base Counter Synchronization: [0][1] • Operational Highlights for the ePWM Trip-Zone Submodule: [0] • Controlling and Monitoring the ePWM Time-Base Submodule: [0][1][2][3] • Operational Overview of the ePWM Event-Trigger Submodule: [0][1] • ePWM / HRPWM Functional Register Groups: [0]

11.5.5.6 EPWM_TBPRD Register (Offset = Ah) [reset = 0h]

EPWM_TBPRD is shown in Figure 11-408 and described in Table 11-849.

Table 11-848. EPWM_TBPRD Instances

Instance	Physical Address
EPWM_0	021D 000Ah
EPWM_1	021D 040Ah
EPWM_2	021D 080Ah
EPWM_3	021D 0C0Ah
EPWM_4	021D 100Ah
EPWM_5	021D 140Ah

Figure 11-408. EPWM_TBPRD Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBPRD															
R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-849. EPWM_TBPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TBPRD	R/W	0h	These bits determine the period of the time-base counter. This sets the PWM frequency. Shadowing of this register is enabled and disabled by the EPWM_TBCTL[3] PRDLD bit. By default this register is shadowed. (a) If EPWM_TBCTL[3] PRDLD = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the active register will be loaded from the shadow register when the time-base counter equals zero. (b) If EPWM_TBCTL[3] PRDLD = 1, then the shadow is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. (c) The active and shadow registers share the same memory map address.

Table 11-850. Register Call Summary for EPWM_TBPRD

ePWM Registers • EPWM_CMPCTL Register (Offset = Eh) [reset = 0h]: [0][1] • EPWM_TBPRD Register (Offset = Ah) [reset = 0h]: [0] • EPWM_TBCTL Register (Offset = 0h) [reset = 0h]: [0][1][2][3][4] • ePWM Registers: [0][1]
ePWM Functional Description • ePWM Time-Base (TB) Submodule: [0] • ePWM Action-Qualifier (AQ) Submodule: [0][1][2][3][4][5] • Calculating PWM Period and Frequency: [0][1] • ePWM Time-Base Period Shadow Register: [0][1][2][3][4][5][6] • Waveforms for Common ePWM Configurations: [0][1] • ePWM Action-Qualifier Event Priority: [0] • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Time-Base Submodule: [0] • Operational Overview of the ePWM Event-Trigger Submodule: [0]

11.5.5.7 EPWM_CMPCTL Register (Offset = Eh) [reset = 0h]

EPWM_CMPCTL is shown in Figure 11-409 and described in Table 11-852.

Table 11-851. EPWM_CMPCTL Instances

Instance	Physical Address
EPWM_0	021D 000Eh
EPWM_1	021D 040Eh
EPWM_2	021D 080Eh
EPWM_3	021D 0C0Eh
EPWM_4	021D 100Eh
EPWM_5	021D 140Eh

Figure 11-409. EPWM_CMPCTL Register

15	14	13	12	11	10	9	8
RESERVED						SHDWBFULL	SHDWAFULL
R-0h						R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	SHDWBMODE	RESERVED	SHDWAMODE	LOADBMODE		LOADAMODE	
R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-852. EPWM_CMPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9	SHDWBFULL	R	0h	Counter-compare B (EPWM_CMPB) Shadow Register Full Status Flag. This bit self clears once a load-strobe occurs. 0h = CMPB shadow FIFO not full yet 1h = Indicates the CMPB shadow FIFO is full. A CPU write will overwrite current shadow value.
8	SHDWAFULL	R	0h	Counter-compare A (EPWM_CMPA) Shadow Register Full Status Flag. The flag bit is set when a 32 bit write to CMPA:CMPAHR register or a 16 bit write to EPWM_CMPA register is made. A 16 bit write to HRPWM_CMPAHR register will not affect the flag. This bit self clears once a load-strobe occurs. 0h = CMPA shadow FIFO not full yet 1h = Indicates the CMPA shadow FIFO is full, a CPU write will overwrite the current shadow value.
7	RESERVED	R	0h	Reserved
6	SHDWBMODE	R/W	0h	Counter-compare B (EPWM_CMPB) Register Operating Mode. 0h = Shadow mode. Operates as a double buffer. All writes via the CPU access the shadow register. 1h = Immediate mode. Only the active compare B register is used. All writes and reads directly access the active register for immediate compare action.
5	RESERVED	R	0h	
4	SHDWAMODE	R/W	0h	Counter-compare A (EPWM_CMPA) Register Operating Mode. 0h = Shadow mode. Operates as a double buffer. All writes via the CPU access the shadow register. 1h = Immediate mode. Only the active compare register is used. All writes and reads directly access the active register for immediate compare action

Table 11-852. EPWM_CMPCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	LOADBMODE	R/W	0h	Active Counter-Compare B (CMPB) Load From Shadow Select Mode. This bit has no effect in immediate mode (EPWM_CMPCTL[6] SHDWBMODE = 1). 0h = Load on TBCNT = 0: Time-base counter equal to zero (EPWM_TBCNT bitfield TBCNT= 0000h) 1h = Load on TBCNT = TBPRD: Time-base counter equal to period (EPWM_TBCNT bitfield TBCNT = TBPRD bitfield of the active EPWM_TBPRD) 2h = Load on either TBCNT = 0 or TBCNT = TBPRD 3h = Freeze (no loads possible)
1-0	LOADAMODE	R/W	0h	Active Counter-Compare A (EPWM_CMPA) Load From Shadow Select Mode. This bit has no effect in immediate mode (EPWM_CMPCTL[4] SHDWAMODE = 1). 0h = Load on TBCNT = 0: Time-base counter equal to zero (EPWM_TBCNT bitfield TBCNT = 0000h) 1h = Load on TBCNT = TBPRD: Time-base counter equal to period (EPWM_TBCNT bitfield TBCNT= TBPRD bitfield of the active EPWM_TBPRD) 2h = Load on either TBCNT = 0 or TBCNT = TBPRD 3h = Freeze (no loads possible)

Table 11-853. Register Call Summary for EPWM_CMPCTL

ePWM Registers
- EPWM_CMPB Register (Offset = 14h) [reset = 0h]: [0][1][2][3][4]
- HRPWM_HRCTL Register (Offset = 40h) [reset = 0h]: [0]
- EPWM_CMPCTL Register (Offset = Eh) [reset = 0h]: [0][1][2]
- EPWM_CMPA Register (Offset = 12h) [reset = 0h]: [0][1][2][3][4]
- ePWM Registers: [0][1]

ePWM Functional Description
- Operational Highlights for the ePWM Counter-Compare Submodule: [0][1][2][3][4][5][6][7]
- ePWM / HRPWM Functional Register Groups: [0]
- Controlling and Monitoring the ePWM Counter-Compare Submodule: [0]
- ePWM Action-Qualifier (AQ) Submodule: [0][1][2][3][4][5]

11.5.5.8 HRPWM_CMPAHR Register (Offset = 10h) [reset = 100h]

HRPWM_CMPAHR is shown in Figure 11-410 and described in Table 11-855.

Table 11-854. HRPWM_CMPAHR Instances

Instance	Physical Address
EPWM_0	021D 0010h
EPWM_1	021D 0410h
EPWM_2	021D 0810h
EPWM_3	021D 0C10h
EPWM_4	021D 1010h
EPWM_5	021D 1410h

Figure 11-410. HRPWM_CMPAHR Register

15	14	13	12	11	10	9	8
CMPAHR							
R/W-1h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-855. HRPWM_CMPAHR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	CMPAHR	R/W	1h	Compare A High-Resolution register bits for MEP step control. A minimum value of 1h is needed to enable HRPWM capabilities. Valid MEP range of operation 1-255h.
7-0	RESERVED	R	0h	Reserved

Table 11-856. Register Call Summary for HRPWM_CMPAHR

ePWM Registers • HRPWM_CMPAHR Register (Offset = 10h) [reset = 100h]: [0] • EPWM_CMPCTL Register (Offset = Eh) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • Operational Highlights for the ePWM Counter-Compare Submodule: [0] • HRPWM Edge Positioning: [0] • HRPWM Scaling Considerations: [0][1][2][3][4] • Controlling and Monitoring the ePWM Counter-Compare Submodule: [0] • Architecture of the High-Resolution PWM Submodule: [0] • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the High-Resolution PWM Submodule: [0][1] • Configuring the High-Resolution PWM Submodule: [0][1][2]

11.5.5.9 EPWM_CMPA Register (Offset = 12h) [reset = 0h]

EPWM_CMPA is shown in Figure 11-411 and described in Table 11-858.

Table 11-857. EPWM_CMPA Instances

Instance	Physical Address
EPWM_0	021D 0012h
EPWM_1	021D 0412h
EPWM_2	021D 0812h
EPWM_3	021D 0C12h
EPWM_4	021D 1012h
EPWM_5	021D 1412h

Figure 11-411. EPWM_CMPA Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPA															
R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-858. EPWM_CMPA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	CMPA	R/W	0h	The value in the active EPWM_CMPA register is continuously compared to the time-base counter (register EPWM_TBCNT). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare A" event. This event is sent to the action-qualifier where it is qualified and converted it into one or more actions. These actions can be applied to either the EPWMxA or the EPWMxB output depending on the configuration of the EPWM_AQCTLA and EPWM_AQCTLB registers. The actions that can be defined in the EPWM_AQCTLA and EPWM_AQCTLB registers include the following. (a) Do nothing the event is ignored. (b) Clear: Pull the EPWMxA and/or EPWMxB signal low. (c) Set: Pull the EPWMxA and/or EPWMxB signal high. (d) Toggle the EPWMxA and/or EPWMxB signal. Shadowing of this register is enabled and disabled by the EPWM_CMPCTL[4] SHDWAMODE bit. By default this register is shadowed. (a) If EPWM_CMPCTL[4] SHDWAMODE = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the EPWM_CMPCTL[1-0] LOADAMODE bit field determines which event will load the active register from the shadow register. (b) Before a write, the EPWM_CMPCTL[8] SHDWAFULL bit can be read to determine if the shadow register is currently full. (c) If EPWM_CMPCTL[4] SHDWAMODE = 1, then the shadow register is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. (d) In either mode, the active and shadow registers share the same memory map address.

Table 11-859. Register Call Summary for EPWM_CMPA

ePWM Registers • EPWM_AQCTLB Register (Offset = 18h) [reset = 0h]: [0][1] • EPWM_CMPCTL Register (Offset = Eh) [reset = 0h]: [0][1][2][3] • EPWM_CMPA Register (Offset = 12h) [reset = 0h]: [0][1] • EPWM_AQCTLA Register (Offset = 16h) [reset = 0h]: [0][1] • ePWM Registers: [0][1]
ePWM Functional Description • ePWM Action-Qualifier (AQ) Submodule: [0][1][2][3][4][5][6][7][8][9][10][11] • Overview: [0][1] • Controlling and Monitoring the ePWM Counter-Compare Submodule: [0] • Operational Highlights for the ePWM Counter-Compare Submodule: [0][1][2][3][4][5] • HRPWM Scaling Considerations: [0][1][2][3] • ePWM / HRPWM Functional Register Groups: [0] • Waveforms for Common ePWM Configurations: [0][1][2][3][4][5][6][7][8] • Controlling and Monitoring the High-Resolution PWM Submodule: [0] • Operational Overview of the ePWM Event-Trigger Submodule: [0][1]

11.5.5.10 EPWM_CMPB Register (Offset = 14h) [reset = 0h]

EPWM_CMPB is shown in Figure 11-412 and described in Table 11-861.

Table 11-860. EPWM_CMPB Instances

Instance	Physical Address
EPWM_0	021D 0014h
EPWM_1	021D 0414h
EPWM_2	021D 0814h
EPWM_3	021D 0C14h
EPWM_4	021D 1014h
EPWM_5	021D 1414h

Figure 11-412. EPWM_CMPB Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPB															
R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-861. EPWM_CMPB Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	CMPB	R/W	0h	The value in the active EPWM_CMPB register is continuously compared to the time-base counter (register EPWM_TBCNT). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare B" event. This event is sent to the action-qualifier where it is qualified and converted it into one or more actions. These actions can be applied to either the EPWMxA or the EPWMxB output depending on the configuration of the EPWM_AQCTLA and EPWM_AQCTLB registers. The actions that can be defined in the EPWM_AQCTLA and EPWM_AQCTLB registers include the following. (a) Do nothing, the event is ignored. (b) Clear: Pull the EPWMxA and/or EPWMxB signal low. (c) Set: Pull the EPWMxA and/or EPWMxB signal high. (d) Toggle the EPWMxA and/or EPWMxB signal. Shadowing of this register is enabled and disabled by the EPWM_CMPCTL[6] SHDWBMODE bit. By default this register is shadowed. (a) If EPWM_CMPCTL[6] SHDWBMODE = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the EPWM_CMPCTL[3-2] LOADBMODE bit field determines which event will load the active register from the shadow register: (b) Before a write, the EPWM_CMPCTL[9] SHDWBFULL bit can be read to determine if the shadow register is currently full. (c) If EPWM_CMPCTL[6] SHDWBMODE = 1, then the shadow register is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. (d) In either mode, the active and shadow registers share the same memory map address.

Table 11-862. Register Call Summary for EPWM_CMPB

ePWM Registers • EPWM_AQCTLB Register (Offset = 18h) [reset = 0h]: [0][1] • EPWM_AQCTLA Register (Offset = 16h) [reset = 0h]: [0][1] • ePWM Registers: [0][1] • EPWM_CMPB Register (Offset = 14h) [reset = 0h]: [0][1] • EPWM_TBCTL Register (Offset = 0h) [reset = 0h]: [0] • EPWM_CMPCTL Register (Offset = Eh) [reset = 0h]: [0][1]
ePWM Functional Description • ePWM Action-Qualifier (AQ) Submodule: [0][1][2][3][4][5][6][7][8][9][10][11] • Operational Highlights for the ePWM Counter-Compare Submodule: [0][1][2][3] • Overview: [0][1] • Controlling and Monitoring the ePWM Counter-Compare Submodule: [0] • Waveforms for Common ePWM Configurations: [0][1][2][3][4][5][6][7] • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Time-Base Submodule: [0] • Operational Overview of the ePWM Event-Trigger Submodule: [0][1]

11.5.5.11 EPWM_AQCTLA Register (Offset = 16h) [reset = 0h]

EPWM_AQCTLA is shown in Figure 11-413 and described in Table 11-864.

Table 11-863. EPWM_AQCTLA Instances

Instance	Physical Address
EPWM_0	021D 0016h
EPWM_1	021D 0416h
EPWM_2	021D 0816h
EPWM_3	021D 0C16h
EPWM_4	021D 1016h
EPWM_5	021D 1416h

Figure 11-413. EPWM_AQCTLA Register

15	14	13	12	11	10	9	8
RESERVED				CBD		CBU	
R-0h				R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
CAD		CAU		PRD		ZRO	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-864. EPWM_AQCTLA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-10	CBD	R/W	0h	Action when the time-base counter equals the active EPWM_CMPB register and the counter is decrementing. 0h = Do nothing (action disabled) 1h = Clear: force EPWMxA output low. 2h = Set: force EPWMxA output high. 3h = Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
9-8	CBU	R/W	0h	Action when the counter equals the active EPWM_CMPB register and the counter is incrementing. 0h = Do nothing (action disabled) 1h = Clear: force EPWMxA output low. 2h = Set: force EPWMxA output high. 3h = Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
7-6	CAD	R/W	0h	Action when the counter equals the active EPWM_CMPA register and the counter is decrementing. 0h = Do nothing (action disabled) 1h = Clear: force EPWMxA output low. 2h = Set: force EPWMxA output high. 3h = Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
5-4	CAU	R/W	0h	Action when the counter equals the active EPWM_CMPA register and the counter is incrementing. 0h = Do nothing (action disabled) 1h = Clear: force EPWMxA output low. 2h = Set: force EPWMxA output high. 3h = Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.

Table 11-864. EPWM_AQCTLA Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	PRD	R/W	0h	Action when the counter equals the period. Note: By definition, in count up-down mode when the counter equals period the direction is defined as 0 or counting down. 0h = Do nothing (action disabled) 1h = Clear: force EPWMxA output low. 2h = Set: force EPWMxA output high. 3h = Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
1-0	ZRO	R/W	0h	Action when counter equals zero. Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 0h = Do nothing (action disabled) 1h = Clear: force EPWMxA output low. 2h = Set: force EPWMxA output high. 3h = Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.

Table 11-865. Register Call Summary for EPWM_AQCTLA

ePWM Registers • EPWM_CMPB Register (Offset = 14h) [reset = 0h]: [0][1] • EPWM_CMPA Register (Offset = 12h) [reset = 0h]: [0][1] • EPWM_AQCTLA Register (Offset = 16h) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • ePWM Action-Qualifier (AQ) Submodule: [0][1][2][3][4][5] • Controlling and Monitoring the ePWM Action-Qualifier Submodule: [0] • ePWM / HRPWM Functional Register Groups: [0]

11.5.5.12 EPWM_AQCTLB Register (Offset = 18h) [reset = 0h]

EPWM_AQCTLB is shown in Figure 11-414 and described in Table 11-867.

Table 11-866. EPWM_AQCTLB Instances

Instance	Physical Address
EPWM_0	021D 0018h
EPWM_1	021D 0418h
EPWM_2	021D 0818h
EPWM_3	021D 0C18h
EPWM_4	021D 1018h
EPWM_5	021D 1418h

Figure 11-414. EPWM_AQCTLB Register

15	14	13	12	11	10	9	8
RESERVED				CBD		CBU	
R-0h				R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
CAD		CAU		PRD		ZRO	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-867. EPWM_AQCTLB Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-10	CBD	R/W	0h	Action when the counter equals the active EPWM_CMPB register and the counter is decrementing. 0h = Do nothing (action disabled) 1h = Clear: force EPWMxB output low. 2h = Set: force EPWMxB output high. 3h = Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.
9-8	CBU	R/W	0h	Action when the counter equals the active EPWM_CMPB register and the counter is incrementing. 0h = Do nothing (action disabled) 1h = Clear: force EPWMxB output low. 2h = Set: force EPWMxB output high. 3h = Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.
7-6	CAD	R/W	0h	Action when the counter equals the active EPWM_CMPA register and the counter is decrementing. 0h = Do nothing (action disabled) 1h = Clear: force EPWMxB output low. 2h = Set: force EPWMxB output high. 3h = Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.
5-4	CAU	R/W	0h	Action when the counter equals the active EPWM_CMPA register and the counter is incrementing. 0h = Do nothing (action disabled) 1h = Clear: force EPWMxB output low. 2h = Set: force EPWMxB output high. 3h = Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.

Table 11-867. EPWM_AQCTLB Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	PRD	R/W	0h	Action when the counter equals the period. Note: By definition, in count up-down mode when the counter equals period the direction is defined as 0 or counting down. 0h = Do nothing (action disabled) 1h = Clear: force EPWMxB output low. 2h = Set: force EPWMxB output high. 3h = Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.
1-0	ZRO	R/W	0h	Action when counter equals zero. Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 0h = Do nothing (action disabled) 1h = Clear: force EPWMxB output low. 2h = Set: force EPWMxB output high. 3h = Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low.

Table 11-868. Register Call Summary for EPWM_AQCTLB

ePWM Registers • EPWM_CMPB Register (Offset = 14h) [reset = 0h]: [0][1] • EPWM_CMPA Register (Offset = 12h) [reset = 0h]: [0][1] • EPWM_AQCTLB Register (Offset = 18h) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • ePWM Action-Qualifier (AQ) Submodule: [0][1][2][3][4][5] • Controlling and Monitoring the ePWM Action-Qualifier Submodule: [0] • ePWM / HRPWM Functional Register Groups: [0]

11.5.5.13 EPWM_AQSFRC Register (Offset = 1Ah) [reset = 0h]

EPWM_AQSFRC is shown in Figure 11-415 and described in Table 11-870.

Table 11-869. EPWM_AQSFRC Instances

Instance	Physical Address
EPWM_0	021D 001Ah
EPWM_1	021D 041Ah
EPWM_2	021D 081Ah
EPWM_3	021D 0C1Ah
EPWM_4	021D 101Ah
EPWM_5	021D 141Ah

Figure 11-415. EPWM_AQSFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RLDCSF		OTSFB	ACTSFB		OTSFA	ACTSFA	
R/W-0h		R/W-0h	R/W-0h		R/W-0h	R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-870. EPWM_AQSFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-6	RLDCSF	R/W	0h	EPWM_AQCSFRC Active Register Reload From Shadow Options.
5	OTSFB	R/W	0h	One-Time Software Forced Event on Output B. 0h = Writing a 0 (zero) has no effect. Always reads back a 0. This bit is auto cleared once a write to this register is complete, that is, a forced event is initiated. This is a one-shot forced event. It can be overridden by another subsequent event on output B. 1h = Initiates a single s/w forced event
4-3	ACTSFB	R/W	0h	Action when One-Time Software Force B Is invoked 0h = Does nothing (action disabled) 1h = Clear (low) 2h = Set (high) 3h = Toggle (Low -> High, High -> Low). Note: This action is not qualified by counter direction (CNT_dir)
2	OTSFA	R/W	0h	One-Time Software Forced Event on Output A. 0h = Writing a 0 (zero) has no effect. Always reads back a 0. This bit is auto cleared once a write to this register is complete (that is, a forced event is initiated). 1h = Initiates a single software forced event.
1-0	ACTSFA	R/W	0h	Action When One-Time Software Force A Is Invoked. 0h = Does nothing (action disabled). 1h = Clear (low). 2h = Set (high). 3h = Toggle (Low -> High, High -> Low). Note: This action is not qualified by counter direction (CNT_dir)

Table 11-871. Register Call Summary for EPWM_AQSFRC

ePWM Registers • EPWM_AQSFRC Register (Offset = 1Ah) [reset = 0h]: [0] • EPWM_AQCSFRC Register (Offset = 1Ch) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • Controlling and Monitoring the ePWM Action-Qualifier Submodule: [0][1] • ePWM / HRPWM Functional Register Groups: [0]

11.5.5.14 EPWM_AQCSFRC Register (Offset = 1Ch) [reset = 0h]

EPWM_AQCSFRC is shown in Figure 11-416 and described in Table 11-873.

Table 11-872. EPWM_AQCSFRC Instances

Instance	Physical Address
EPWM_0	021D 001Ch
EPWM_1	021D 041Ch
EPWM_2	021D 081Ch
EPWM_3	021D 0C1Ch
EPWM_4	021D 101Ch
EPWM_5	021D 141Ch

Figure 11-416. EPWM_AQCSFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CSFB		CSFA	
R-0h				R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-873. EPWM_AQCSFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-2	CSFB	R/W	0h	Continuous Software Force on Output B. In immediate mode, a continuous force takes effect on the next TBCLK edge. In shadow mode, a continuous force takes effect on the next TBCLK edge after a shadow load into the active register. To configure shadow mode, use EPWM_AQSFRC[7-6] RLDCSF. 0h = Forcing disabled, that is, has no effect 1h = Forces a continuous low on output B 2h = Forces a continuous high on output B 3h = Software forcing is disabled and has no effect
1-0	CSFA	R/W	0h	Continuous Software Force on Output A In immediate mode, a continuous force takes effect on the next TBCLK edge. In shadow mode, a continuous force takes effect on the next TBCLK edge after a shadow load into the active register. 0h = Forcing disabled, that is, has no effect 1h = Forces a continuous low on output A 2h = Forces a continuous high on output A 3h = Software forcing is disabled and has no effect

Table 11-874. Register Call Summary for EPWM_AQCSFRC

ePWM Registers • EPWM_AQSFRC Register (Offset = 1Ah) [reset = 0h]: [0] • EPWM_AQCSFRC Register (Offset = 1Ch) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • Controlling and Monitoring the ePWM Action-Qualifier Submodule: [0][1] • ePWM / HRPWM Functional Register Groups: [0]

11.5.5.15 EPWM_DBCTL Register (Offset = 1Eh) [reset = 0h]

EPWM_DBCTL is shown in Figure 11-417 and described in Table 11-876.

Table 11-875. EPWM_DBCTL Instances

Instance	Physical Address
EPWM_0	021D 001Eh
EPWM_1	021D 041Eh
EPWM_2	021D 081Eh
EPWM_3	021D 0C1Eh
EPWM_4	021D 101Eh
EPWM_5	021D 141Eh

Figure 11-417. EPWM_DBCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		IN_MODE		POLSEL		OUT_MODE	
R-0h		R/W-0h		R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-876. EPWM_DBCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-6	RESERVED	R	0h	Reserved
5-4	IN_MODE	R/W	0h	Dead Band Input Mode Control. Bit 5 controls the S5 switch and bit 4 controls the S4 switch. This allows you to select the input source to the falling-edge and rising-edge delay. To produce classical dead-band waveforms, the default is EPWMxA In is the source for both falling and rising-edge delays. 0h = EPWMxA In (from the action-qualifier) is the source for both falling-edge and rising-edge delay. 1h = EPWMxB In (from the action-qualifier) is the source for rising-edge delayed signal. EPWMxA In (from the action-qualifier) is the source for falling-edge delayed signal. 2h = EPWMxA In (from the action-qualifier) is the source for rising-edge delayed signal. EPWMxB In (from the action-qualifier) is the source for falling-edge delayed signal. 3h = EPWMxB In (from the action-qualifier) is the source for both rising-edge delay and falling-edge delayed signal.
3-2	POLSEL	R/W	0h	Polarity Select Control. Bit 3 controls the S3 switch and bit 2 controls the S2 switch. This allows you to selectively invert one of the delayed signals before it is sent out of the dead-band submodule. The following descriptions correspond to classical upper/lower switch control as found in one leg of a digital motor control inverter. These assume that EPWM_DBCTL[1-0] OUT_MODE = 0b11 and EPWM_DBCTL[5-4] IN_MODE = 0b00. Other enhanced modes are also possible, but not regarded as typical usage modes. 0h = Active high (AH) mode. Neither EPWMxA nor EPWMxB is inverted (default). 1h = Active low complementary (ALC) mode. EPWMxA is inverted. 2h = Active high complementary (AHC). EPWMxB is inverted. 3h = Active low (AL) mode. Both EPWMxA and EPWMxB are inverted.

Table 11-876. EPWM_DBCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	OUT_MODE	R/W	0h	Dead-band Output Mode Control. Bit 1 controls the S1 switch and bit 0 controls the S0 switch. This allows you to selectively enable or bypass the dead-band generation for the falling-edge and rising-edge delay. 0h = Dead-band generation is bypassed for both output signals. In this mode, both the EPWMxA and EPWMxB output signals from the action-qualifier are passed directly to the PWM-chopper submodule. In this mode, the POLSEL and IN_MODE bits have no effect. 1h = Disable rising-edge delay. The EPWMxA signal from the action-qualifier is passed straight through to the EPWMxA input of the PWM-chopper submodule. The falling-edge delayed signal is seen on output EPWMxB. The input signal for the delay is determined by EPWM_DBCTL[5-4] IN_MODE. 2h = Disable falling-edge delay. The EPWMxB signal from the action-qualifier is passed straight through to the EPWMxB input of the PWM-chopper submodule. The rising-edge delayed signal is seen on output EPWMxA. The input signal for the delay is determined by EPWM_DBCTL[5-4] IN_MODE. 3h = Dead-band is fully enabled for both rising-edge delay on output EPWMxA and falling-edge delay on output EPWMxB. The input signal for the delay is determined by EPWM_DBCTL[5-4] IN_MODE.

Table 11-877. Register Call Summary for EPWM_DBCTL

ePWM Registers • EPWM_DBCTL Register (Offset = 1Eh) [reset = 0h]: [0][1][2][3][4][5] • ePWM Registers: [0][1]
ePWM Functional Description • Operational Highlights for the ePWM Dead-Band Generator Submodule: [0][1][2][3][4][5][6] • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Dead-Band Submodule: [0]

11.5.5.16 EPWM_DBRED Register (Offset = 20h) [reset = 0h]

EPWM_DBRED is shown in Figure 11-418 and described in Table 11-879.

Table 11-878. EPWM_DBRED Instances

Instance	Physical Address
EPWM_0	021D 0020h
EPWM_1	021D 0420h
EPWM_2	021D 0820h
EPWM_3	021D 0C20h
EPWM_4	021D 1020h
EPWM_5	021D 1420h

Figure 11-418. EPWM_DBRED Register

15	14	13	12	11	10	9	8
RESERVED						DEL	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
DEL							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-879. EPWM_DBRED Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	DEL	R/W	0h	Rising Edge Delay Count. 10 bit counter.

Table 11-880. Register Call Summary for EPWM_DBRED

ePWM Registers • EPWM_DBRED Register (Offset = 20h) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • Operational Highlights for the ePWM Dead-Band Generator Submodule: [0][1] • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Dead-Band Submodule: [0]

11.5.5.17 EPWM_DBFED Register (Offset = 22h) [reset = 0h]

EPWM_DBFED is shown in Figure 11-419 and described in Table 11-882.

Table 11-881. EPWM_DBFED Instances

Instance	Physical Address
EPWM_0	021D 0022h
EPWM_1	021D 0422h
EPWM_2	021D 0822h
EPWM_3	021D 0C22h
EPWM_4	021D 1022h
EPWM_5	021D 1422h

Figure 11-419. EPWM_DBFED Register

15	14	13	12	11	10	9	8
RESERVED						DEL	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
DEL							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-882. EPWM_DBFED Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	DEL	R/W	0h	Falling Edge Delay Count. 10 bit counter

Table 11-883. Register Call Summary for EPWM_DBFED

ePWM Registers • EPWM_DBFED Register (Offset = 22h) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • Operational Highlights for the ePWM Dead-Band Generator Submodule: [0][1] • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Dead-Band Submodule: [0]

11.5.5.18 EPWM_TZSEL Register (Offset = 24h) [reset = 0h]

EPWM_TZSEL is shown in Figure 11-420 and described in Table 11-885.

Table 11-884. EPWM_TZSEL Instances

Instance	Physical Address
EPWM_0	021D 0024h
EPWM_1	021D 0424h
EPWM_2	021D 0824h
EPWM_3	021D 0C24h
EPWM_4	021D 1024h
EPWM_5	021D 1424h

Figure 11-420. EPWM_TZSEL Register

15	14	13	12	11	10	9	8
OSHTN							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							CBC0
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-885. EPWM_TZSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R	0h	Reserved
13	OSHT6	R/W	0h	Trip-zone 5 (TZ5) select. One-Shot (OSHT) trip-zone enable/disable. When any of the enabled pins go low, a one-shot trip event occurs for this ePWM module. When the event occurs, the action defined in the EPWM_TZCTL register is taken on the EPWMxA and EPWMxB outputs. The one-shot trip condition remains latched until you clear the condition via the EPWM_TZCLR register. 0h: Disable TZ5 as a one-shot trip source for this ePWM module. 1h: Enable TZ5 as a one-shot trip source for this ePWM module.
12	OSHT5	R/W	0h	Trip-zone 4 (TZ4) select. One-Shot (OSHT) trip-zone enable/disable. When any of the enabled pins go low, a one-shot trip event occurs for this ePWM module. When the event occurs, the action defined in the EPWM_TZCTL register is taken on the EPWMxA and EPWMxB outputs. The one-shot trip condition remains latched until you clear the condition via the EPWM_TZCLR register. 0h: Disable TZ4 as a one-shot trip source for this ePWM module. 1h: Enable TZ4 as a one-shot trip source for this ePWM module.
11	OSHT4	R/W	0h	Trip-zone 3 (TZ3) select. One-Shot (OSHT) trip-zone enable/disable. When any of the enabled pins go low, a one-shot trip event occurs for this ePWM module. When the event occurs, the action defined in the EPWM_TZCTL register is taken on the EPWMxA and EPWMxB outputs. The one-shot trip condition remains latched until you clear the condition via the EPWM_TZCLR register. 0h: Disable TZ3 as a one-shot trip source for this ePWM module. 1h: Enable TZ3 as a one-shot trip source for this ePWM module.

Table 11-885. EPWM_TZSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	OSHT3	R/W	0h	Trip-zone 2 (TZ2) select. One-Shot (OSHT) trip-zone enable/disable. When any of the enabled pins go low, a one-shot trip event occurs for this ePWM module. When the event occurs, the action defined in the EPWM_TZCTL register is taken on the EPWMxA and EPWMxB outputs. The one-shot trip condition remains latched until you clear the condition via the EPWM_TZCLR register. 0h: Disable TZ2 as a one-shot trip source for this ePWM module. 1h: Enable TZ2 as a one-shot trip source for this ePWM module.
9	OSHT2	R/W	0h	Trip-zone 1 (TZ1) select. One-Shot (OSHT) trip-zone enable/disable. When any of the enabled pins go low, a one-shot trip event occurs for this ePWM module. When the event occurs, the action defined in the EPWM_TZCTL register is taken on the EPWMxA and EPWMxB outputs. The one-shot trip condition remains latched until you clear the condition via the EPWM_TZCLR register. 0h: Disable TZ1 as a one-shot trip source for this ePWM module. 1h: Enable TZ1 as a one-shot trip source for this ePWM module.
8	OSHT1	R/W	0h	Trip-zone 0 (TZ0) select. One-Shot (OSHT) trip-zone enable/disable. When any of the enabled pins go low, a one-shot trip event occurs for this ePWM module. When the event occurs, the action defined in the EPWM_TZCTL register is taken on the EPWMxA and EPWMxB outputs. The one-shot trip condition remains latched until you clear the condition via the EPWM_TZCLR register. 0h: Disable TZ0 as a one-shot trip source for this ePWM module. 1h: Enable TZ0 as a one-shot trip source for this ePWM module.
7-6	RESERVED	R	0h	Reserved
5	CBC5	R/W	0h	Trip-zone 5 (TZ5) select. Cycle-by-Cycle (CBC) trip-zone enable/disable. When any of the enabled pins go low, a cycle-by-cycle trip event occurs for this ePWM module. When the event occurs, the action defined in the EPWM_TZCTL register is taken on the EPWMxA and EPWMxB outputs. A cycle-by-cycle trip condition is automatically cleared when the time-base counter reaches zero. 0h: Disable TZ5 as a CBC trip source for this ePWM module. 1h: Enable TZ5 as a CBC trip source for this ePWM module.
4	CBC4	R/W	0h	Trip-zone 4 (TZ4) select. Cycle-by-Cycle (CBC) trip-zone enable/disable. When any of the enabled pins go low, a cycle-by-cycle trip event occurs for this ePWM module. When the event occurs, the action defined in the EPWM_TZCTL register is taken on the EPWMxA and EPWMxB outputs. A cycle-by-cycle trip condition is automatically cleared when the time-base counter reaches zero. 0h: Disable TZ4 as a CBC trip source for this ePWM module. 1h: Enable TZ4 as a CBC trip source for this ePWM module.
3	CBC3	R/W	0h	Trip-zone 3 (TZ3) select. Cycle-by-Cycle (CBC) trip-zone enable/disable. When any of the enabled pins go low, a cycle-by-cycle trip event occurs for this ePWM module. When the event occurs, the action defined in the EPWM_TZCTL register is taken on the EPWMxA and EPWMxB outputs. A cycle-by-cycle trip condition is automatically cleared when the time-base counter reaches zero. 0h: Disable TZ3 as a CBC trip source for this ePWM module. 1h: Enable TZ3 as a CBC trip source for this ePWM module.

Table 11-885. EPWM_TZSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	CBC2	R/W	0h	Trip-zone 2 (TZ2) select. Cycle-by-Cycle (CBC) trip-zone enable/disable. When any of the enabled pins go low, a cycle-by-cycle trip event occurs for this ePWM module. When the event occurs, the action defined in the EPWM_TZCTL register is taken on the EPWMxA and EPWMxB outputs. A cycle-by-cycle trip condition is automatically cleared when the time-base counter reaches zero. 0h: Disable TZ2 as a CBC trip source for this ePWM module. 1h: Enable TZ2 as a CBC trip source for this ePWM module.
1	CBC1	R/W	0h	Trip-zone 1 (TZ1) select. Cycle-by-Cycle (CBC) trip-zone enable/disable. When any of the enabled pins go low, a cycle-by-cycle trip event occurs for this ePWM module. When the event occurs, the action defined in the EPWM_TZCTL register is taken on the EPWMxA and EPWMxB outputs. A cycle-by-cycle trip condition is automatically cleared when the time-base counter reaches zero. 0h: Disable TZ1 as a CBC trip source for this ePWM module. 1h: Enable TZ1 as a CBC trip source for this ePWM module.
0	CBC0	R/W	0h	Trip-zone 0 (TZ0) select. Cycle-by-Cycle (CBC) trip-zone enable/disable. When any of the enabled pins go low, a cycle-by-cycle trip event occurs for this ePWM module. When the event occurs, the action defined in the EPWM_TZCTL register is taken on the EPWMxA and EPWMxB outputs. A cycle-by-cycle trip condition is automatically cleared when the time-base counter reaches zero. 0h: Disable TZ0 as a CBC trip source for this ePWM module. 1h: Enable TZ0 as a CBC trip source for this ePWM module.

Table 11-886. Register Call Summary for EPWM_TZSEL

ePWM Registers • EPWM_TZSEL Register (Offset = 24h) [reset = 0h]: [0] • EPWM_TZCTL Register (Offset = 28h) [reset = 0h]: [0][1] • ePWM Registers: [0][1]
ePWM Functional Description • Operational Highlights for the ePWM Trip-Zone Submodule: [0][1][2][3][4] • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Trip-Zone Submodule: [0]

11.5.5.19 EPWM_TZCTL Register (Offset = 28h) [reset = 0h]

EPWM_TZCTL is shown in Figure 11-421 and described in Table 11-888.

Table 11-887. EPWM_TZCTL Instances

Instance	Physical Address
EPWM_0	021D 0028h
EPWM_1	021D 0428h
EPWM_2	021D 0828h
EPWM_3	021D 0C28h
EPWM_4	021D 1028h
EPWM_5	021D 1428h

Figure 11-421. EPWM_TZCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				TZB		TZA	
R-0h				R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-888. EPWM_TZCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-2	TZB	R/W	0h	When a trip event occurs the following action is taken on output EPWMxB. Which trip-zone pins can cause an event is defined in the EPWM_TZSEL register. 0h = High impedance (EPWMxB = High-impedance state) 1h = Force EPWMxB to a high state 2h = Force EPWMxB to a low state 3h = Do nothing, no action is taken on EPWMxB.
1-0	TZA	R/W	0h	When a trip event occurs the following action is taken on output EPWMxA. Which trip-zone pins can cause an event is defined in the EPWM_TZSEL register. 0h = High impedance (EPWMxA = High-impedance state) 1h = Force EPWMxA to a high state 2h = Force EPWMxA to a low state 3h = Do nothing, no action is taken on EPWMxA.

Table 11-889. Register Call Summary for EPWM_TZCTL

ePWM Registers • EPWM_TZSEL Register (Offset = 24h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11] • EPWM_TZCTL Register (Offset = 28h) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • Operational Highlights for the ePWM Trip-Zone Submodule: [0][1][2][3][4][5][6][7][8][9] • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Trip-Zone Submodule: [0]

11.5.5.20 EPWM_TZEINT Register (Offset = 2Ah) [reset = 0h]

EPWM_TZEINT is shown in Figure 11-422 and described in Table 11-891.

Table 11-890. EPWM_TZEINT Instances

Instance	Physical Address
EPWM_0	021D 002Ah
EPWM_1	021D 042Ah
EPWM_2	021D 082Ah
EPWM_3	021D 0C2Ah
EPWM_4	021D 102Ah
EPWM_5	021D 142Ah

Figure 11-422. EPWM_TZEINT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					OST	CBC	RESERVED
R-0h					R/W-0h	R/W-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-891. EPWM_TZEINT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	Reserved
2	OST	R/W	0h	Trip-zone One-Shot Interrupt Enable 0h = Disable one-shot interrupt generation 1h = Enable Interrupt generation a one-shot trip event will cause a EPWMxTZINT interrupt.
1	CBC	R/W	0h	Trip-zone Cycle-by-Cycle Interrupt Enable 0h = Disable cycle-by-cycle interrupt generation. 1h = Enable interrupt generation a cycle-by-cycle trip event will cause an EPWMxTZINT interrupt.
0	RESERVED	R	0h	Reserved

Table 11-892. Register Call Summary for EPWM_TZEINT

ePWM Registers • EPWM_TZEINT Register (Offset = 2Ah) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • Operational Highlights for the ePWM Trip-Zone Submodule: [0][1] • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Trip-Zone Submodule: [0]

11.5.5.21 EPWM_TZFLG Register (Offset = 2Ch) [reset = 0h]

EPWM_TZFLG is shown in Figure 11-423 and described in Table 11-894.

Table 11-893. EPWM_TZFLG Instances

Instance	Physical Address
EPWM_0	021D 002Ch
EPWM_1	021D 042Ch
EPWM_2	021D 082Ch
EPWM_3	021D 0C2Ch
EPWM_4	021D 102Ch
EPWM_5	021D 142Ch

Figure 11-423. EPWM_TZFLG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					OST	CBC	INT
R-0h					R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-894. EPWM_TZFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	Reserved
2	OST	R	0h	Latched Status Flag for A One-Shot Trip Event. 0h = No one-shot trip event has occurred. 1h = Indicates a trip event has occurred on a pin selected as a one-shot trip source. This bit is cleared by writing the appropriate value to the EPWM_TZCLR register.
1	CBC	R	0h	Latched Status Flag for Cycle-By-Cycle Trip Event 0h = No cycle-by-cycle trip event has occurred. 1h = Indicates a trip event has occurred on a pin selected as a cycle-by-cycle trip source. The EPWM_TZFLG[1] CBC bit will remain set until it is manually cleared by the user. If the cycle-by-cycle trip event is still present when the CBC bit is cleared, then CBC will be immediately set again. The specified condition on the pins is automatically cleared when the ePWM time-base counter reaches zero (EPWM_TBCNT bitfield TBCNT = 0000h) if the trip condition is no longer present. The condition on the pins is only cleared when the TBCNT = 0000h no matter where in the cycle the CBC flag is cleared. This bit is cleared by writing the appropriate value to the EPWM_TZCLR register.
0	INT	R	0h	Latched Trip Interrupt Status Flag 0h = Indicates no interrupt has been generated. 1h = Indicates an EPWMxTZINT interrupt was generated because of a trip condition. No further EPWMxTZINT interrupts will be generated until this flag is cleared. If the interrupt flag is cleared when either CBC or OST is set, then another interrupt pulse will be generated. Clearing all flag bits will prevent further interrupts. This bit is cleared by writing the appropriate value to the EPWM_TZCLR register.

Table 11-895. Register Call Summary for EPWM_TZFLG

ePWM Registers • EPWM_TZFLG Register (Offset = 2Ch) [reset = 0h]: [0][1] • EPWM_TZCLR Register (Offset = 2Eh) [reset = 0h]: [0][1] • EPWM_TZFRC Register (Offset = 30h) [reset = 0h]: [0][1] • ePWM Registers: [0][1]
ePWM Functional Description • Operational Highlights for the ePWM Trip-Zone Submodule: [0][1][2][3] • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Trip-Zone Submodule: [0]

11.5.5.22 EPWM_TZCLR Register (Offset = 2Eh) [reset = 0h]

EPWM_TZCLR is shown in Figure 11-424 and described in Table 11-897.

Table 11-896. EPWM_TZCLR Instances

Instance	Physical Address
EPWM_0	021D 002Eh
EPWM_1	021D 042Eh
EPWM_2	021D 082Eh
EPWM_3	021D 0C2Eh
EPWM_4	021D 102Eh
EPWM_5	021D 142Eh

Figure 11-424. EPWM_TZCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					OST	CBC	INT
R-0h					R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-897. EPWM_TZCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	Reserved
2	OST	R/W	0h	Clear Flag for One-Shot Trip (OST) Latch 0h = Has no effect. Always reads back a 0. 1h = Clears this Trip (set) condition.
1	CBC	R/W	0h	Clear Flag for Cycle-By-Cycle (CBC) Trip Latch 0h = Has no effect. Always reads back a 0. 1h = Clears this Trip (set) condition.
0	INT	R/W	0h	Global Interrupt Clear Flag 0h = Has no effect. Always reads back a 0. 1h = Clears the trip-interrupt flag for this ePWM module (EPWM_TZFLG[0] INT). Note: No further EPWMxTZINT interrupts will be generated until the flag is cleared. If the EPWM_TZFLG[0] INT bit is cleared and any of the other flag bits are set, then another interrupt pulse will be generated. Clearing all flag bits will prevent further interrupts.

Table 11-898. Register Call Summary for EPWM_TZCLR

ePWM Registers • EPWM_TZFLG Register (Offset = 2Ch) [reset = 0h]: [0][1][2] • EPWM_TZSEL Register (Offset = 24h) [reset = 0h]: [0][1][2][3][4][5] • EPWM_TZCLR Register (Offset = 2Eh) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • Operational Highlights for the ePWM Trip-Zone Submodule: [0][1] • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Trip-Zone Submodule: [0]

11.5.5.23 EPWM_TZFRC Register (Offset = 30h) [reset = 0h]

EPWM_TZFRC is shown in Figure 11-425 and described in Table 11-900.

Table 11-899. EPWM_TZFRC Instances

Instance	Physical Address
EPWM_0	021D 0030h
EPWM_1	021D 0430h
EPWM_2	021D 0830h
EPWM_3	021D 0C30h
EPWM_4	021D 1030h
EPWM_5	021D 1430h

Figure 11-425. EPWM_TZFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					OST	CBC	RESERVED
R-0h					R/W-0h	R/W-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-900. EPWM_TZFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	Reserved
2	OST	R/W	0h	Force a One-Shot Trip Event via Software 0h = Writing of 0 is ignored. Always reads back a 0. 1h = Forces a one-shot trip event and sets the EPWM_TZFLG[2] OST bit.
1	CBC	R/W	0h	Force a Cycle-by-Cycle Trip Event via Software 0h = Writing of 0 is ignored. Always reads back a 0. 1h = Forces a cycle-by-cycle trip event and sets the EPWM_TZFLG[1] CBC bit.
0	RESERVED	R	0h	Reserved

Table 11-901. Register Call Summary for EPWM_TZFRC

ePWM Registers • EPWM_TZFRC Register (Offset = 30h) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Trip-Zone Submodule: [0]

11.5.5.24 EPWM_ETSEL Register (Offset = 32h) [reset = 0h]

EPWM_ETSEL is shown in Figure 11-426 and described in Table 11-903.

Table 11-902. EPWM_ETSEL Instances

Instance	Physical Address
EPWM_0	021D 0032h
EPWM_1	021D 0432h
EPWM_2	021D 0832h
EPWM_3	021D 0C32h
EPWM_4	021D 1032h
EPWM_5	021D 1432h

Figure 11-426. EPWM_ETSEL Register

15	14	13	12	11	10	9	8
SOCB	SOCBSEL			SOCA	SOCASEL		
R/W-0h	R/W-0h			R/W-0h	R-0h		
7	6	5	4	3	2	1	0
RESERVED				INTEN	INTSEL		
R-0h				R/W-0h	R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-903. EPWM_ETSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SOCB	R/W	0h	Enable SOCB pulse when set to 1.
14-12	SOCBSEL	R/W	0h	EPWMxSOCB Selection Options: 0h: Reserved 1h: Enable event time-base counter equal to zero (CNT_zero event). 2h: Enable event time-base counter equal to period (PRD_eq event). 3h: Reserved 4h: Enable event time-base counter equal to CMPA when the timer is incrementing (CMPA_eq_UC event). 5h: Enable event time-base counter equal to CMPA when the timer is decrementing (CMPA_eq_DC event). 6h: Enable event time-base counter equal to CMPB when the timer is incrementing (CMPB_eq_UC event). 7h: Enable event time-base counter equal to CMPB when the timer is decrementing (CMPB_eq_DC event).
11	SOCA	R/W	0h	Enable SOCA pulse when set to 1.
10-8	SOCASEL	R/W	0h	EPWMxSOCA Selection Options: 0h: Reserved 1h: Enable event time-base counter equal to zero (CNT_zero event). 2h: Enable event time-base counter equal to period (PRD_eq event). 3h: Reserved 4h: Enable event time-base counter equal to CMPA when the timer is incrementing (CMPA_eq_UC event). 5h: Enable event time-base counter equal to CMPA when the timer is decrementing (CMPA_eq_DC event). 6h: Enable event time-base counter equal to CMPB when the timer is incrementing (CMPB_eq_UC event). 7h: Enable event time-base counter equal to CMPB when the timer is decrementing (CMPB_eq_DC event).
7-4	RESERVED	R	0h	Reserved
3	INTEN	R/W	0h	Enable ePWM Interrupt (EPWMx_INT) Generation 0h: Disable EPWMx_INT generation 1h: Enable EPWMx_INT generation

Table 11-903. EPWM_ETSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	INTSEL	R/W	0h	ePWM Interrupt (EPWMx_INT) Selection Options 0h: Reserved 1h: Enable event time-base counter equal to zero. (TBCNT = 0000h) 2h: Enable event time-base counter equal to period (TBCNT = TBPRD) 3h: Reserved 4h: Enable event time-base counter equal to CMPA when the timer is incrementing. 5h: Enable event time-base counter equal to CMPA when the timer is decrementing. 6h: Enable event time-base counter equal to CMPB when the timer is incrementing. 7h: Enable event time-base counter equal to CMPB when the timer is decrementing.

Table 11-904. Register Call Summary for EPWM_ETSEL

ePWM Registers • EPWM_ETFRC Register (Offset = 3Ah) [reset = 0h]: [0][1][2] • EPWM_ETSEL Register (Offset = 32h) [reset = 0h]: [0] • EPWM_ETPS Register (Offset = 34h) [reset = 0h]: [0][1][2][3] • ePWM Registers: [0][1]
ePWM Functional Description • ePWM Event-Trigger (ET) Submodule: [0][1][2] • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Event-Trigger Submodule: [0] • Operational Overview of the ePWM Event-Trigger Submodule: [0][1][2]

11.5.5.25 EPWM_ETPS Register (Offset = 34h) [reset = 0h]

EPWM_ETPS is shown in Figure 11-427 and described in Table 11-906.

Table 11-905. EPWM_ETPS Instances

Instance	Physical Address
EPWM_0	021D 0034h
EPWM_1	021D 0434h
EPWM_2	021D 0834h
EPWM_3	021D 0C34h
EPWM_4	021D 1034h
EPWM_5	021D 1434h

Figure 11-427. EPWM_ETPS Register

15	14	13	12	11	10	9	8
SOCBCNT		SOCBPRD		SOCACNT		SOCAPRD	
R-0h		R/W-0h		R-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED				INTCNT		INTPRD	
R-0h				R-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-906. EPWM_ETPS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	SOCBCNT	R	0h	EPWMxSOCB Counter Register: These bits indicate how many selected events have occurred: 0h: No events 1h: 1 events 2h: 2 events 3h: 3 event
13-12	SOCBPRD	R/W	0h	EPWMxSOCB Period Select: These bits select how many selected event need to occur before an SOCB pulse is generated: 0h: Disable counter 1h: Generate pulse on SOCBCNT = 1 (1st event) 2h: Generate pulse on SOCBCNT = 2 (2nd event) 3h: Generate pulse on SOCBCNT = 3 (3rd event)
11-10	SOCACNT	R	0h	EPWMxSOCA Counter Register: These bits indicate how many selected events have occurred: 0h: No events 1h: 1 events 2h: 2 events 3h: 3 events
9-8	SOCAPRD	R/W	0h	EPWMxSOCA Period Select: These bits select how many selected event need to occur before an SOCA pulse is generated: 0h: Disable counter 1h: Generate pulse on SOCACNT = 1 (1st event) 2h: Generate pulse on SOCACNT = 2 (2nd event) 3h: Generate pulse on SOCACNT = 3 (3rd event)
7-4	RESERVED	R	0h	Reserved

Table 11-906. EPWM_ETPS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	INTCNT	R	0h	ePWM Interrupt Event (EPWMx_INT) Counter Register. These bits indicate how many selected EPWM_ETSEL[2-0] INTSEL events have occurred. These bits are automatically cleared when an interrupt pulse is generated. If interrupts are disabled, EPWM_ETSEL[0] INT = 0 or the interrupt flag is set, EPWM_ETFLG[0] INT = 1, the counter will stop counting events when it reaches the period value EPWM_ETPS[3-2] INTCNT = EPWM_ETPS[1-0] INTPRD. 0h: No events have occurred. 1h: 1 event has occurred. 2h: 2 events have occurred. 3h: 3 events have occurred.
1-0	INTPRD	R/W	0h	ePWM Interrupt (EPWMx_INT) Period Select. These bits determine how many selected EPWM_ETSEL[2-0] INTSEL events need to occur before an interrupt is generated. To be generated, the interrupt must be enabled (EPWM_ETSEL[0] INT = 1). If the interrupt status flag is set from a previous interrupt (EPWM_ETFLG[0] INT = 1) then no interrupt will be generated until the flag is cleared via the EPWM_ETCLR[0] INT bit. This allows for one interrupt to be pending while another is still being serviced. Once the interrupt is generated, the EPWM_ETPS[3-2] INTCNT bits will automatically be cleared. Writing a INTPRD value that is the same as the current counter value will trigger an interrupt if it is enabled and the status flag is clear. Writing a INTPRD value that is less than the current counter value will result in an undefined state. If a counter event occurs at the same instant as a new zero or non-zero INTPRD value is written, the counter is incremented. 0h: Disable the interrupt event counter. No interrupt will be generated and EPWM_ETFRC[0] INT is ignored. 1h: Generate an interrupt on the first event INTCNT = 01 (first event) 2h: Generate interrupt on EPWM_ETPS[3-2] INTCNT = 0b10 (second event) 3h: Generate interrupt on EPWM_ETPS[3-2] INTCNT = 0b11 (third event)

Table 11-907. Register Call Summary for EPWM_ETPS

ePWM Registers • EPWM_ETPS Register (Offset = 34h) [reset = 0h]: [0][1][2][3][4][5] • ePWM Registers: [0][1]
ePWM Functional Description • ePWM Event-Trigger (ET) Submodule: [0][1] • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Event-Trigger Submodule: [0] • Operational Overview of the ePWM Event-Trigger Submodule: [0][1][2][3][4][5][6][7][8][9][10]

11.5.5.26 EPWM_ETFLG Register (Offset = 36h) [reset = 0h]

EPWM_ETFLG is shown in Figure 11-428 and described in Table 11-909.

Table 11-908. EPWM_ETFLG Instances

Instance	Physical Address
EPWM_0	021D 0036h
EPWM_1	021D 0436h
EPWM_2	021D 0836h
EPWM_3	021D 0C36h
EPWM_4	021D 1036h
EPWM_5	021D 1436h

Figure 11-428. EPWM_ETFLG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				SOCB	SOCA	RESERVED	INT
R-0h				R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-909. EPWM_ETFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	SOCB	R	0h	Latched SOCB Flag Bit Status: 0h: Indicates no event occurred. 1h: Indicates that a start of conversion pulse was generated on EPWMxSOCB. Note: Unlike the INT flag bit, the EPWMxSOCB output will continue to pulse even if the flag bit is set.
2	SOCA	R	0h	Latched SOCA Flag Bit Status: 0h: Indicates no event occurred. 1h: Indicates that a start of conversion pulse was generated on ePWMxSOCA. Note: Unlike the INT flag bit, the EPWMxSOCA output will continue to pulse even if the flag bit is set.
1	RESERVED	R	0h	Reserved
0	INT	R	0h	Latched ePWM Interrupt (EPWMx_INT) Status Flag: 0h: Indicates no event occurred. 1h: Indicates that an ePWMx interrupt (EWPMx_INT) was generated. No further interrupts will be generated until the flag bit is cleared. Up to one interrupt can be pending while the EPWM_ETFLG[0] INT bit is still set. If an interrupt is pending, it will not be generated until after the EPWM_ETFLG[0] INT bit is cleared.

Table 11-910. Register Call Summary for EPWM_ETFLG

ePWM Registers
- EPWM_ETCLR Register (Offset = 38h) [reset = 0h]: [0][1][2]
- EPWM_ETFRC Register (Offset = 3Ah) [reset = 0h]: [0][1]
- EPWM_ETFLG Register (Offset = 36h) [reset = 0h]: [0][1][2]
- EPWM_ETPS Register (Offset = 34h) [reset = 0h]: [0][1]
- ePWM Registers: [0][1]

Table 11-910. Register Call Summary for EPWM_ETFLG (continued)

ePWM Functional Description • ePWM Event-Trigger (ET) Submodule: [0] • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Event-Trigger Submodule: [0] • Operational Overview of the ePWM Event-Trigger Submodule: [0][1][2][3]

11.5.5.27 EPWM_ETCLR Register (Offset = 38h) [reset = 0h]

EPWM_ETCLR is shown in Figure 11-429 and described in Table 11-912.

Table 11-911. EPWM_ETCLR Instances

Instance	Physical Address
EPWM_0	021D 0038h
EPWM_1	021D 0438h
EPWM_2	021D 0838h
EPWM_3	021D 0C38h
EPWM_4	021D 1038h
EPWM_5	021D 1438h

Figure 11-429. EPWM_ETCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				SOCB	SOCA	RESERVED	INT
R-0h				R/W-0h	R/W-0h	R-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-912. EPWM_ETCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	SOCB	R/W	0h	SOCB Flag Clear Bit: 0h: Writing a 0 has no effect. Always reads back a 0. 1h: Writing a 1 clears the EPWM_ETFLG[3] SOCB flag bit.
2	SOCA	R/W	0h	SOCA Flag Clear Bit: 0h: Writing a 0 has no effect. Always reads back a 0. 1h: Writing a 1 clears the EPWM_ETFLG[2] SOCA flag bit.
1	RESERVED	R	0h	Reserved
0	INT	R/W	0h	ePWM Interrupt (EPWMx_INT) Flag Clear Bit: 0h: Writing a 0 has no effect. Always reads back a 0. 1h: Writing 1 clears the EPWM_ETFLG[0] INT flag bit and enable further interrupts pulses to be generated.

Table 11-913. Register Call Summary for EPWM_ETCLR

ePWM Registers • EPWM_ETCLR Register (Offset = 38h) [reset = 0h]: [0] • EPWM_ETPS Register (Offset = 34h) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Event-Trigger Submodule: [0]

11.5.5.28 EPWM_ETFRC Register (Offset = 3Ah) [reset = 0h]

EPWM_ETFRC is shown in Figure 11-430 and described in Table 11-915.

Table 11-914. EPWM_ETFRC Instances

Instance	Physical Address
EPWM_0	021D 003Ah
EPWM_1	021D 043Ah
EPWM_2	021D 083Ah
EPWM_3	021D 0C3Ah
EPWM_4	021D 103Ah
EPWM_5	021D 143Ah

Figure 11-430. EPWM_ETFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				SOCB	SOCA	RESERVED	INT
R-0h				R/W-0h	R/W-0h	R-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-915. EPWM_ETFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	SOCB	R/W	0h	SOCB Force Bit: 0h: Writing 0 to this bit will be ignored. Always reads back a 0. 1h: Writing 1 will generate a pulse on EPWMxSOCB and set the EPWM_ETFLG[3] SOCB flag bit. This bit is used for test purposes. Note: The SOCB pulse will only be generated if the event is enabled in the EPWM_ETSEL register. The SOCB flag bit will be set regardless.
2	SOCA	R/W	0h	SOCA Force Bit: 0h: Writing 0 to this bit will be ignored. Always reads back a 0. 1h: Writing 1 will generate a pulse on EPWMxSOCA and set the EPWM_ETFLG[2] SOCA flag bit. This bit is used for test purposes. Note: The SOCA pulse will only be generated if the event is enabled in the EPWM_ETSEL register. The SOCA flag bit will be set regardless.
1	RESERVED	R	0h	Reserved
0	INT	R/W	0h	INT Force Bit. 0h: Writing 0 to this bit will be ignored. Always reads back a 0. 1h: Writing 1 generates an interrupt on EPWMxINT and set the INT flag bit. This bit is used for test purposes. Note: The interrupt will only be generated if the event is enabled in the EPWM_ETSEL register. The INT flag bit will be set regardless.

Table 11-916. Register Call Summary for EPWM_ETFRC

ePWM Registers • EPWM_ETFRC Register (Offset = 3Ah) [reset = 0h]: [0] • EPWM_ETPS Register (Offset = 34h) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the ePWM Event-Trigger Submodule: [0] • Operational Overview of the ePWM Event-Trigger Submodule: [0][1]

11.5.5.29 EPWM_PCCTL Register (Offset = 3Ch) [reset = 0h]

EPWM_PCCTL is shown in Figure 11-431 and described in Table 11-918.

Table 11-917. EPWM_PCCTL Instances

Instance	Physical Address
EPWM_0	021D 003Ch
EPWM_1	021D 043Ch
EPWM_2	021D 083Ch
EPWM_3	021D 0C3Ch
EPWM_4	021D 103Ch
EPWM_5	021D 143Ch

Figure 11-431. EPWM_PCCTL Register

15	14	13	12	11	10	9	8
RESERVED					CHPDUTY		
R-0h					R/W-0h		
7	6	5	4	3	2	1	0
CHPFREQ			OSHTWTH				CHPEN
R/W-0h			R/W-0h				R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-918. EPWM_PCCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10-8	CHPDUTY	R/W	0h	Chopping Clock Duty Cycle 0h = Duty = 1/8 (12.5%) 1h = Duty = 2/8 (25.0%) 2h = Duty = 3/8 (37.5%) 3h = Duty = 4/8 (50.0%) 4h = Duty = 5/8 (62.5%) 5h = Duty = 6/8 (75.0%) 6h = Duty = 7/8 (87.5%) 7h = Reserved.
7-5	CHPFREQ	R/W	0h	Chopping Clock Frequency 0h = Divide by 1 (no prescale). 1h = Divide by 2. 2h = Divide by 3. 3h = Divide by 4. 4h = Divide by 5. 5h = Divide by 6. 6h = Divide by 7. 7h = Divide by 8.
4-1	OSHTWTH	R/W	0h	One-Shot Pulse Width 0h = 1 - SYSCLKOUT/8 wide 1h = 2 - SYSCLKOUT/8 wide 2h = 3 - SYSCLKOUT/8 wide 3h = 4 - SYSCLKOUT/8 wide Fh = 16 - SYSCLKOUT/8 wide
0	CHPEN	R/W	0h	PWM-chopping Enable 0h = Disable (bypass) PWM chopping function 1h = Enable chopping function

Table 11-919. Register Call Summary for EPWM_PCCTL

ePWM Registers • EPWM_PCCTL Register (Offset = 3Ch) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • ePWM / HRPWM Functional Register Groups: [0] • Controlling the ePWM-Chopper Submodule: [0] • Operational Highlights for the ePWM-Chopper Submodule: [0]

11.5.5.30 HRPWM_HRCTL Register (Offset = 40h) [reset = 0h]

HRPWM_HRCTL is shown in Figure 11-432 and described in Table 11-921.

This register is only available on ePWM instances that include the high-resolution PWM (HRPWM) extension, otherwise this location is reserved.

Table 11-920. HRPWM_HRCTL Instances

Instance	Physical Address
EPWM_0	021D 0040h
EPWM_1	021D 0440h
EPWM_2	021D 0840h
EPWM_3	021D 0C40h
EPWM_4	021D 1040h
EPWM_5	021D 1440h

Figure 11-432. HRPWM_HRCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				PULSESEL	DELBUSSEL	DELMODE	
R-0h				R/W-0h	R/W-0h	R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-921. HRPWM_HRCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	PULSESEL	R/W	0h	Shadow mode bit. Selects the time event that loads the CMPAHR shadow value into the active register. Note: Load mode selection is valid only if CTLMODE = 0 has been selected. The user should select this event to match the selection of the CMPA load mode EPWM_CMPCTL[LOADMODE] bits in the EPWM module as follows. 0h: Load on CTR = 0. Time-base counter equal to zero (TBCNT = 0000h). 1h: Load on CTR = PRD. Time-base counter equal to period (TBCNT = TBPRD) 2h: Load on either CTR = 0 or CTR = PRD (should not be used with HRPWM) 3h: Freeze (No loads possible should not be used with HRPWM). 0h = CTR = PRD (counter equal period) 1h = CTR = 0 (counter equals zero)
2	DELBUSSEL	R/W	0h	Control Mode Bits. Selects the register (CMP or TBPHS) that controls the MEP. 0h = Select CMPAHR(8) register controls the edge position (this is duty control mode). (default on reset). 1h = Select TBPHSHR(8) register controls the edge position (this is phase control mode).

Table 11-921. HRPWM_HRCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	DELMODE	R/W	0h	Edge Mode Bits. Selects the edge of the PWM that is controlled by the micro-edge position (MEP) logic. 0h = HRPWM capability is disabled (default on reset) 1h = MEP control of rising edge 2h = MEP control of falling edge 3h = MEP control of both edges

Table 11-922. Register Call Summary for HRPWM_HRCTL

ePWM Registers • HRPWM_HRCTL Register (Offset = 40h) [reset = 0h]: [0] • ePWM Registers: [0][1]
ePWM Functional Description • ePWM / HRPWM Functional Register Groups: [0] • Controlling and Monitoring the High-Resolution PWM Submodule: [0] • Configuring the High-Resolution PWM Submodule: [0]

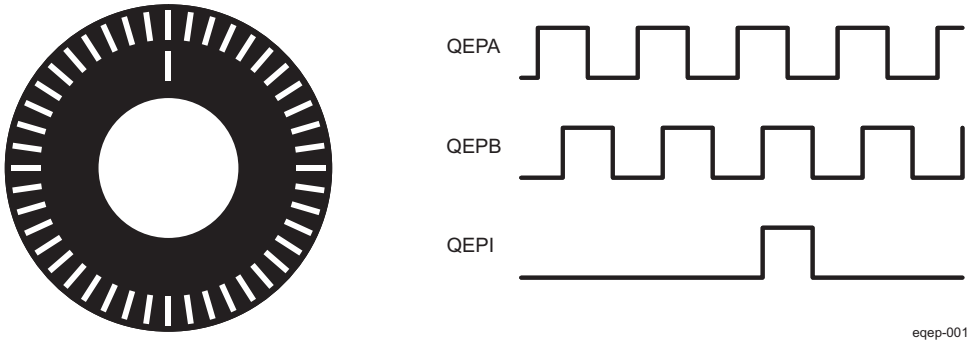
11.6 Enhanced Quadrature Encoder Pulse (eQEP) Module

This section describes the Enhanced Quadrature Encoder Pulse (eQEP) module for the device.

11.6.1 eQEP Overview

A single track of slots patterns the periphery of an incremental encoder disk, as shown in Figure 11-433. These slots create an alternating pattern of dark and light lines. The disk count is defined as the number of dark/light line pairs that occur per revolution (lines per revolution). As a rule, a second track is added to generate a signal that occurs once per revolution (index signal: QEPI), which can be used to indicate an absolute position. Encoder manufacturers identify the index pulse using different terms such as index, marker, home position and zero reference.

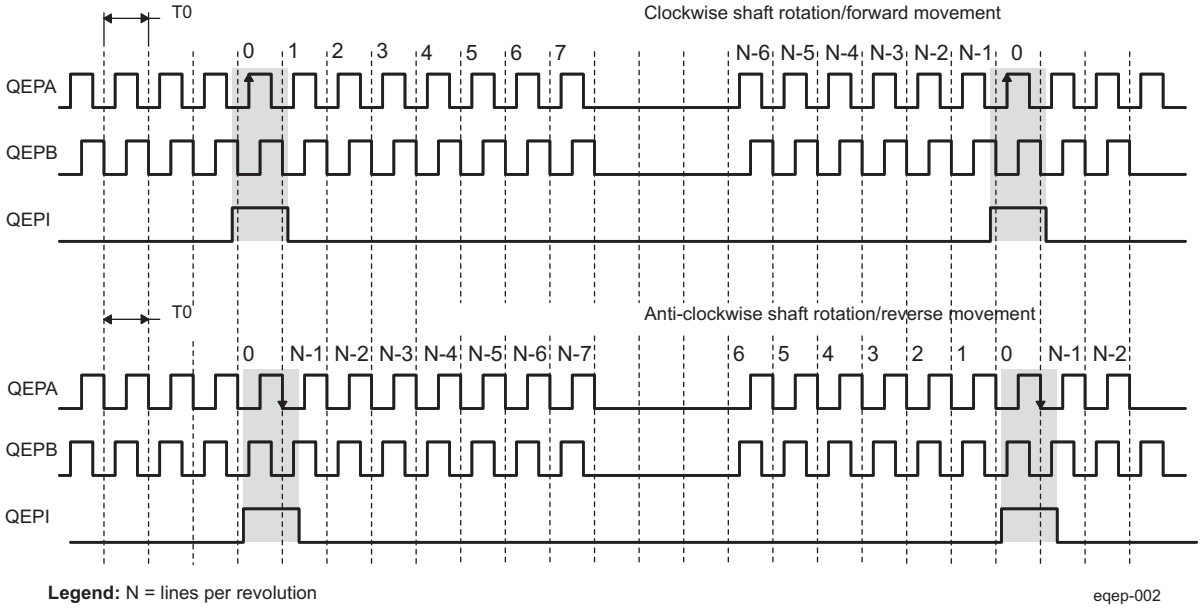
Figure 11-433. Optical Encoder Disk



To derive direction information, the lines on the disk are read out by two different photo-elements that "look" at the disk pattern with a mechanical shift of 1/4 the pitch of a line pair between them. This shift is realized with a reticle or mask that restricts the view of the photo-element to the desired part of the disk lines. As the disk rotates, the two photo-elements generate signals that are shifted 90 degrees out of phase from each other. These are commonly called the quadrature QEPA and QEPB signals. The clockwise direction for most encoders is defined as the QEPA channel going positive before the QEPB channel and vise versa as shown in Figure 11-434.

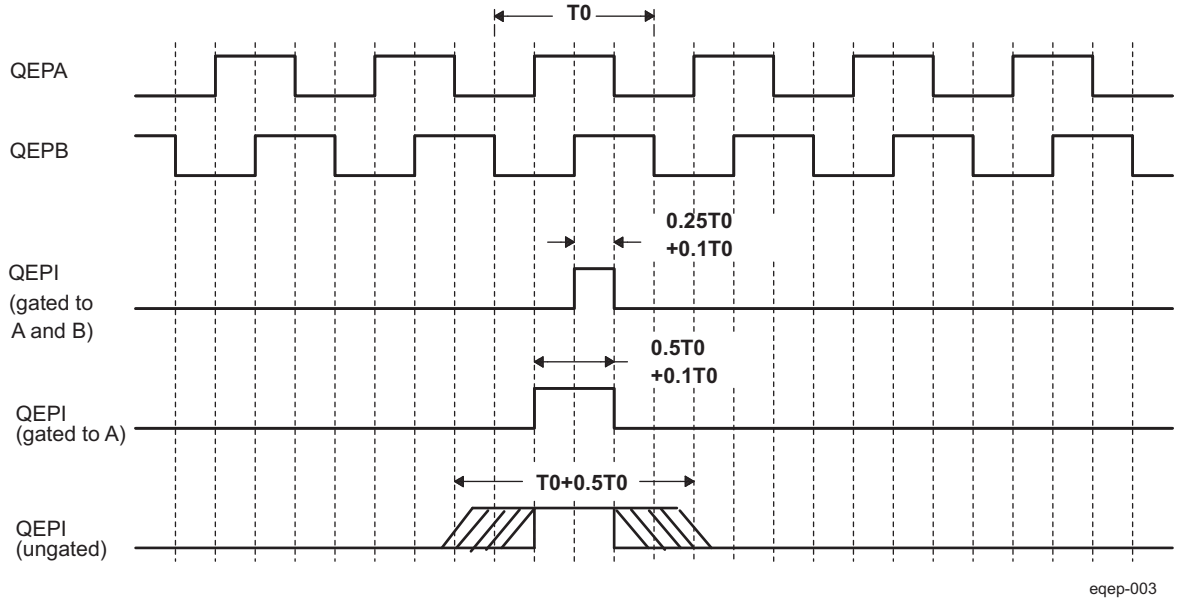
The encoder wheel typically makes one revolution for every revolution of the motor or the wheel may be at a geared rotation ratio with respect to the motor. Therefore, the frequency of the digital signal coming from the QEPA and QEPB outputs varies proportionally with the velocity of the motor. For example, a 2000-line encoder directly coupled to a motor running at 5000 revolutions per minute (rpm) results in a frequency of 166.6 KHz, so by measuring the frequency of either the QEPA or QEPB output, the processor can determine the velocity of the motor.

Figure 11-434. QEP Encoder Output Signal for Forward/Reverse Movement



Quadrature encoders from different manufacturers come with two forms of index pulse (gated index pulse or ungated index pulse) as shown in Figure 11-435. A nonstandard form of index pulse is ungated. In the ungated configuration, the index edges are not necessarily coincident with A and B signals. The gated index pulse is aligned to any of the four quadrature edges and width of the index pulse and can be equal to a quarter, half, or full period of the quadrature signal.

Figure 11-435. Index Pulse Example



Some typical applications of shaft encoders include robotics and even computer input in the form of a mouse. Inside a PC mouse – the mouse ball spins a pair of axles (a left/right, and an up/down axle). These axles are connected to optical shaft encoders that effectively tell the computer how fast and in what direction the mouse is moving.

General Issues: Estimating velocity from a digital position sensor is a cost-effective strategy in motor control. Two different first order approximations for velocity may be written as:

$$V(k) \approx \frac{x(k) - x(k-1)}{T} = \frac{\Delta X}{T} \quad \text{eqep-004} \tag{5}$$

$$V(k) \approx \frac{X}{t(k) - t(k-1)} = \frac{X}{\Delta T} \quad \text{eqep-005} \tag{6}$$

where

v(k): Velocity at time instant k

x(k): Position at time instant k

x(k-1): Position at time instant k - 1

T: Fixed unit time or inverse of velocity calculation rate

ΔX: Incremental position movement in unit time

t(k): Time instant "k"

t(k-1): Time instant "k - 1"

X: Fixed unit position

ΔT: Incremental time elapsed for unit position movement.

Equation 5 is the conventional approach to velocity estimation and it requires a time base to provide unit time event for velocity calculation. Unit time is basically the inverse of the velocity calculation rate.

The encoder count (position) is read once during each unit time event. The quantity [x(k) - x(k-1)] is formed by subtracting the previous reading from the current reading. Then the velocity estimate is computed by multiplying by the known constant 1/T (where T is the constant time between unit time events and is known in advance).

Estimation based on Equation 5 has an inherent accuracy limit directly related to the resolution of the position sensor and the unit time period T. For example, consider a 500-line per revolution quadrature encoder with a velocity calculation rate of 400 Hz. When used for position the quadrature encoder gives a four-fold increase in resolution, in this case, 2000 counts per revolution. The minimum rotation that can be detected is therefore 0.0005 revolutions, which gives a velocity resolution of 12 rpm when sampled at 400 Hz. While this resolution may be satisfactory at moderate or high speeds, for example, 1% error at 1200 rpm, it would clearly prove inadequate at low speeds. In fact, at speeds below 12 rpm, the speed estimate would erroneously be zero much of the time.

At low speed, Equation 6 provides a more accurate approach. It requires a position sensor that outputs a fixed interval pulse train, such as the aforementioned quadrature encoder. The width of each pulse is defined by motor speed for a given sensor resolution. Equation 6 can be used to calculate motor speed by measuring the elapsed time between successive quadrature pulse edges. However, this method suffers from the opposite limitation, as does Equation 5. A combination of relatively large motor speeds and high sensor resolution makes the time interval ΔT small, and thus more greatly influenced by the timer resolution. This can introduce considerable error into high-speed estimates.

For systems with a large speed range (that is, speed estimation is needed at both low and high speeds), one approach is to use Equation 6 at low speed and have the software switch over to Equation 5 when the motor speed rises above some specified threshold.

11.6.2 eQEP Environment

11.6.2.1 eQEP I/O Interface

Table 11-923 shows the device integrated eQEP subsystems interface signals to external devices.

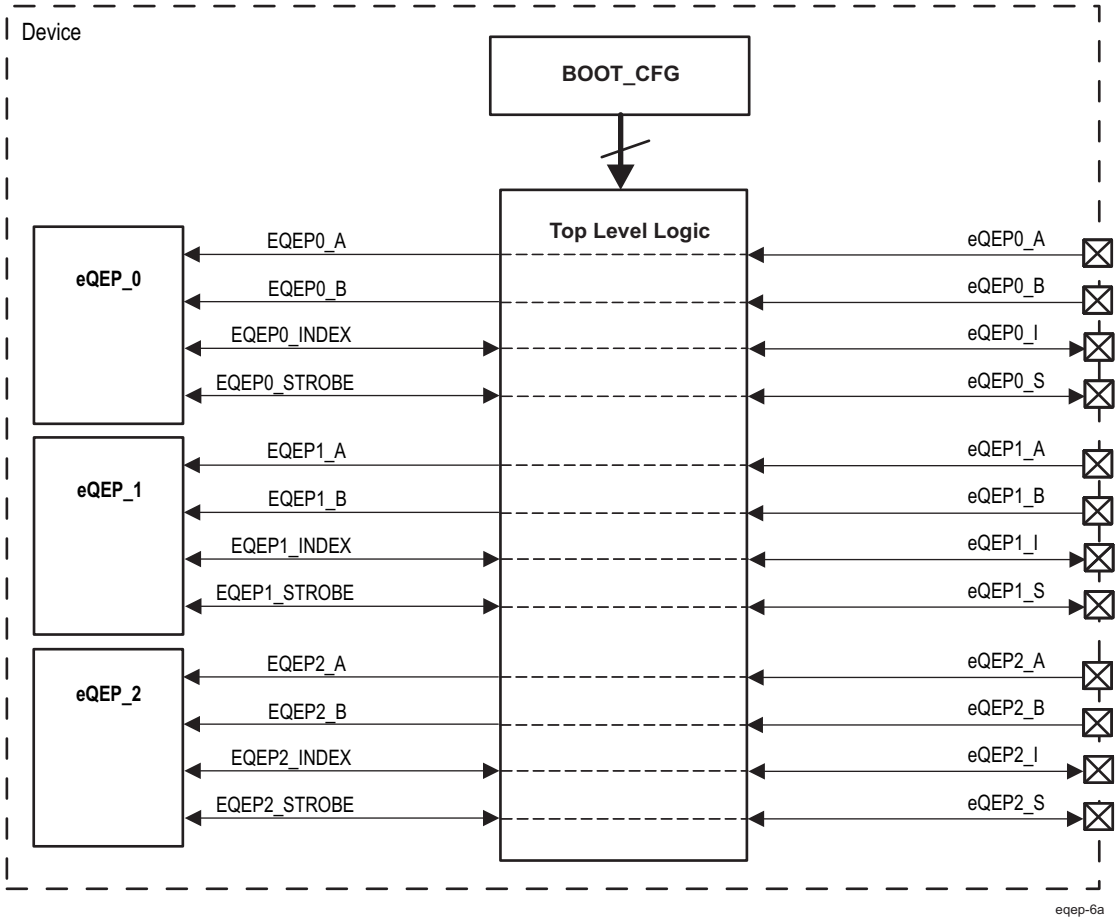
Table 11-923. eQEP Subsystems I/O Signals

eQEP Modules Level Signal Name	Device Level Signal Name	I/O Type[1]	Description	Module Pin Reset Value
eQEP_0				
EQEP0_A	eQEP0_A	I	eQEP_0 Quadrature input	HiZ
EQEP0_B	eQEP0_B	I	eQEP_0 Quadrature input	HiZ
EQEP0_INDEX	eQEP0_I	I/O[2]	eQEP_0 Index input/output	HiZ
EQEP0_STROBE	eQEP0_S	I/O[2]	eQEP_0 Strobe input/output	HiZ
eQEP_1				
EQEP1_A	eQEP1_A	I	eQEP_1 Quadrature input	HiZ
EQEP1_B	eQEP1_B	I	eQEP_1 Quadrature input	HiZ
EQEP1_INDEX	eQEP1_I	I/O[2]	eQEP_1 Index input/output	HiZ
EQEP1_STROBE	eQEP1_S	I/O[2]	eQEP_1 Strobe input/output	HiZ
eQEP_2				
EQEP2_A	eQEP2_A	I	eQEP_2 Quadrature input	HiZ
EQEP2_B	eQEP2_B	I	eQEP_2 Quadrature input	HiZ
EQEP2_INDEX	eQEP2_I	I/O[2]	eQEP_2 Index input/output	HiZ
EQEP2_STROBE	eQEP2_S	I/O[2]	eQEP_2 Strobe input/output	HiZ

(1) I = Input; O = Output

(2) Output functionality is not supported. Only inputs are pinned out. For more information see Section 11.6.3.1, *Device Specific eQEP Features*.

Figure 11-436. eQEP External Interface I/Os



11.6.3 eQEP Integration

This section describes module integration in the device, including information about clocks, resets, and hardware requests.

Figure 11-437 shows the eQEP integration.

There are three instances of the Enhanced Quadrature Encoded Pulse (eQEP) module integrated in the device.

Figure 11-437. eQEP Integration

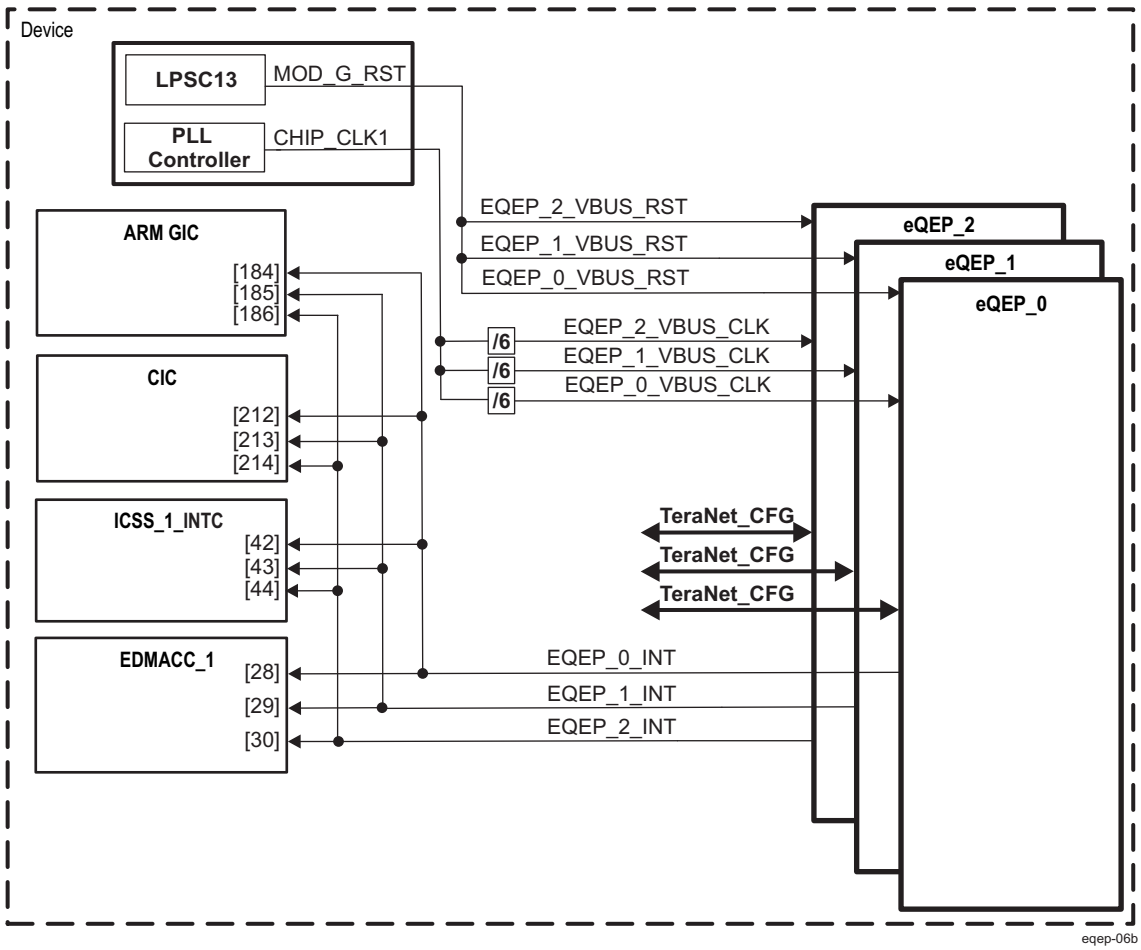


Table 11-924 through Table 11-926 summarize the integration of the module in the device.

Table 11-924. eQEP Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
eQEP_0	PD5	LPSC13	TeraNet_CFG
eQEP_1	PD5	LPSC13	TeraNet_CFG
eQEP_2	PD5	LPSC13	TeraNet_CFG

Table 11-925. eQEP Clocks and Resets

Clocks				
Module Instance	Destination Signal	Source Signal	Source	Description

Table 11-925. eQEP Clocks and Resets (continued)

eQEP_0	EQEP_0_VBUS_CLK	CHIP_CLK1 / 6	PLL_CONTROLLER	eQEP_0 functional and interface clock
eQEP_1	EQEP_1_VBUS_CLK	CHIP_CLK1 / 6	PLL_CONTROLLER	eQEP_1 functional and interface clock
eQEP_2	EQEP_2_VBUS_CLK	CHIP_CLK1 / 6	PLL_CONTROLLER	eQEP_2 functional and interface clock
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
eQEP_0	EQEP_0_VBUS_RST	MOD_G_RST	LPSC13	Module Reset
eQEP_1	EQEP_1_VBUS_RST	MOD_G_RST	LPSC13	Module Reset
eQEP_2	EQEP_2_VBUS_RST	MOD_G_RST	LPSC13	Module Reset

Table 11-926. eQEP Hardware Requests

Interrupt Requests					
Module Instance	**Event Name**	**Mapped To Input Event [Number]**			**Description**
		ARM_GIC	**CIC**	**ICSS_1_INTC**	
eQEP_0	EQEP_0_INT	[184]	[212]	[42]	eQEP_0 interrupt
eQEP_1	EQEP_1_INT	[185]	[213]	[43]	eQEP_1 interrupt
eQEP_2	EQEP_2_INT	[186]	[214]	[44]	eQEP_2 interrupt
DMA Requests					
Module Instance	**Event Name**	**Mapped To Input Event [Number]**			**Description**
		EDMACC_0		**EDMACC_1**	
eQEP_0	EQEP_0_INT	–		[28]	eQEP_0 event to the EDMA_1
eQEP_1	EQEP_1_INT	–		[29]	eQEP_1 event to the EDMA_1
eQEP_2	EQEP_2_INT	–		[30]	eQEP_2 event to the EDMA_1

11.6.3.1 Device Specific eQEP Features

The eQEP module interface have some restrictions in functionality. The eQEP restrictions are, as follows:

- **Only eQEP inputs (A/ B / INDEX and STROBE) are available to user at chip level**
- **eQEP phase error output (EQEP_ERR_PHASE_ERR) is also available**
  - The status of the phase error can be observed by software using the BOOTCFG_EQEP_STAT status register in the BOOT_CFG module.

The eQEP functional interface signals which are NOT available to user are summarized in Table 11-927.

Table 11-927. Device Limitations for the eQEP Functional Interfaces

Module Interface	Signal	Description	Comment
eQEP_n (where n= 0 to 2 for the device)	EQEP_EQEPA_O	eQEP quadrature A output	Not available at chip boundary (can not be used)
	EQEP_EQEPB_O	eQEP quadrature B output	
	EQEP_EQEPA_OEN	eQEP quadrature A output enable	
	EQEP_EQEPB_OEN	eQEP quadrature B output enable	

11.6.4 eQEP Functional Description

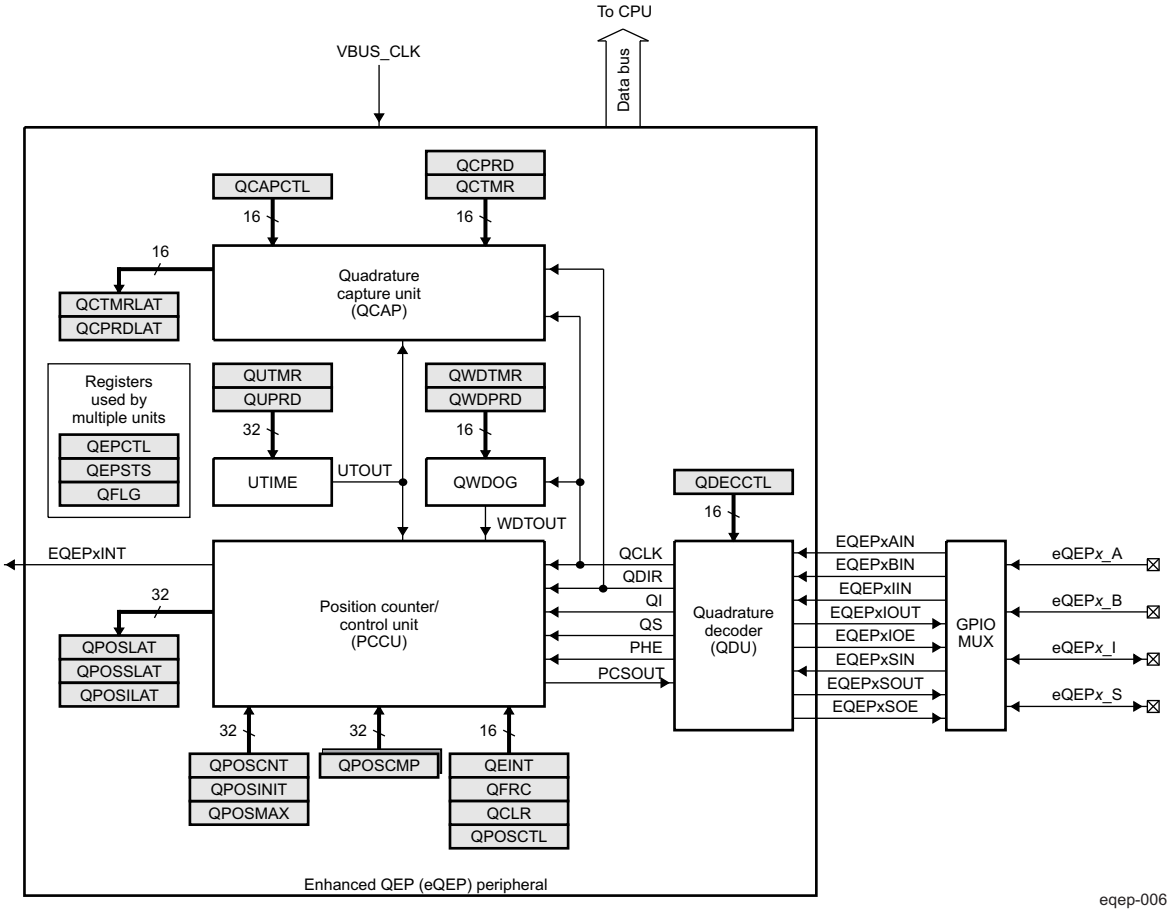
This section provides the eQEP functional description and corrsponding functional details about EQEPx inputs.

NOTE: Multiple identical eQEP modules can be contained in a system. For actual number of eQEP modules integrated in the device, refer to the Section 11.6.3, *eQEP Integration*. The letter x within a signal or module name is used to indicate a generic eQEP instance on a device. For example, output interrupt request, EQEP_0_INT belongs to eQEP0, EQEP_1_INT belongs to eQEP1, and so forth.

The eQEP peripheral contains the following major functional units (as shown in Figure 11-438):

- Programmable input qualification for each pin (part of the GPIO MUX)
- Quadrature decoder unit (QDU)
- Position counter and control unit for position measurement (PCCU)
- Quadrature edge-capture unit for low-speed measurement (QCAP)
- Unit time base for speed/frequency measurement (UTIME)
- Watchdog timer for detecting stalls (QWDOG).

Figure 11-438. Functional Block Diagram of the eQEP Peripheral



11.6.4.1 eQEP Inputs

The eQEP inputs include two pins for quadrature-clock mode or direction-count mode, an index (or 0 marker), and a strobe input.

- eQEPx_A and eQEPx_B: These two pins can be used in quadrature-clock mode or direction-count

mode.

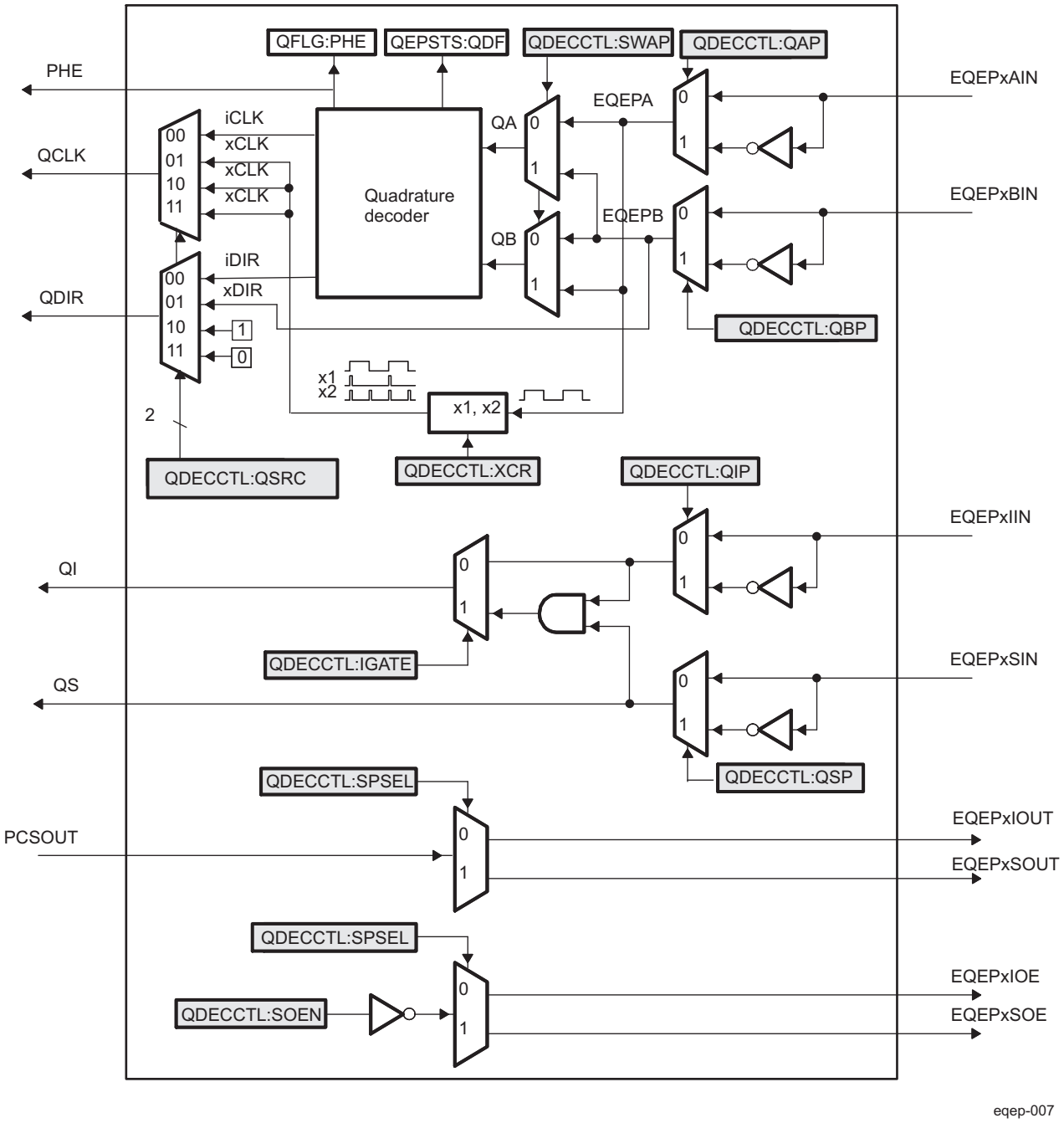
  - Quadrature-clock Mode: The eQEP encoders provide two square wave signals (A and B) 90 electrical degrees out of phase whose phase relationship is used to determine the direction of rotation of the input shaft and number of eQEP pulses from the index position to derive the relative position information. For forward or clockwise rotation, QEPA signal leads QEPB signal and vice versa. The quadrature decoder uses these two inputs to generate quadrature-clock and direction signals.
  - Direction-count Mode: In direction-count mode, direction and clock signals are provided directly from the external source. Some position encoders have this type of output instead of quadrature output. The eQEPx_A pin provides the clock input and the eQEPx_B pin provides the direction input.

- eQEPx_I: Index or Zero Marker: The eQEP encoder uses an index signal to assign an absolute start position from which position information is incrementally encoded using quadrature pulses. This pin is connected to the index output of the eQEP encoder to optionally reset the position counter for each revolution. This signal can be used to initialize or latch the position counter on the occurrence of a desired event on the index pin.
- eQEPx_S: Strobe Input: This general-purpose strobe signal can initialize or latch the position counter on the occurrence of a desired event on the strobe pin. This signal is typically connected to a sensor or limit switch to notify that the motor has reached a defined position.

11.6.4.2 eQEP Quadrature Decoder Unit (QDU)

Figure 11-439 shows a functional block diagram of the QDU.

Figure 11-439. Functional Block Diagram of Decoder Unit



11.6.4.2.1 eQEP Position Counter Input Modes

Clock and direction input to position counter is selected using the QSRC bit in the eQEP decoder control register (EQEP_QDECCTL), based on interface input requirement as follows:

- Quadrature-count mode
- Direction-count mode
- UP-count mode
- DOWN-count mode.

11.6.4.2.1.1 Quadrature Count Mode

The quadrature decoder generates the direction and clock to the position counter in quadrature count mode.

Direction Decoding — The direction decoding logic of the eQEP circuit determines which one of the sequences (EQEPA, EQEPB) is the leading sequence and accordingly updates the direction information in the QDF bit in the eQEP status register (EQEP_QEPSTS). Table 11-928 and Figure 11-440 show the direction decoding logic in truth table and state machine form. Both edges of the EQEPA and EQEPB signals are sensed to generate count pulses for the position counter. Therefore, the frequency of the clock generated by the eQEP logic is four times that of each input sequence. Figure 11-441 shows the direction decoding and clock generation from the eQEP input signals.

Phase Error Flag — In normal operating conditions, quadrature inputs EQEPA and EQEPB will be 90 degrees out of phase. The phase error flag (PHE) is set in the EQEP_QFLG register when edge transition is detected simultaneously on the EQEPA and EQEPB signals to optionally generate interrupts. State transitions marked by dashed lines in Figure 11-440 are invalid transitions that generate a phase error.

Count Multiplication — The eQEP position counter provides 4x times the resolution of an input clock by generating a quadrature-clock (QCLK) on the rising/falling edges of both eQEP input clocks (EQEPA and EQEPB) as shown in Figure 11-441.

Reverse Count — In normal quadrature count operation, EQEPA input is fed to the QA input of the quadrature decoder and the EQEPB input is fed to the QB input of the quadrature decoder. Reverse counting is enabled by setting the SWAP bit in the eQEP decoder control register (EQEP_QDECCTL). This will swap the input to the quadrature decoder thereby reversing the counting direction.

Table 11-928. Quadrature Decoder Truth Table

Previous Edge	Present Edge	QDIR	QPOSCNT
QA↑	QB↑	UP	Increment
	QB↓	DOWN	Decrement
	QA↓	TOGGLE	Increment or Decrement
QA↓	QB↓	UP	Increment
	QB↑	DOWN	Decrement
	QA↑	TOGGLE	Increment or Decrement
QB↑	QA↑	DOWN	Increment
	QA↓	UP	Decrement
	QB↓	TOGGLE	Increment or Decrement
QB↓	QA↓	DOWN	Increment
	QA↑	UP	Decrement
	QB↑	TOGGLE	Increment or Decrement

Figure 11-440. Quadrature Decoder State Machine

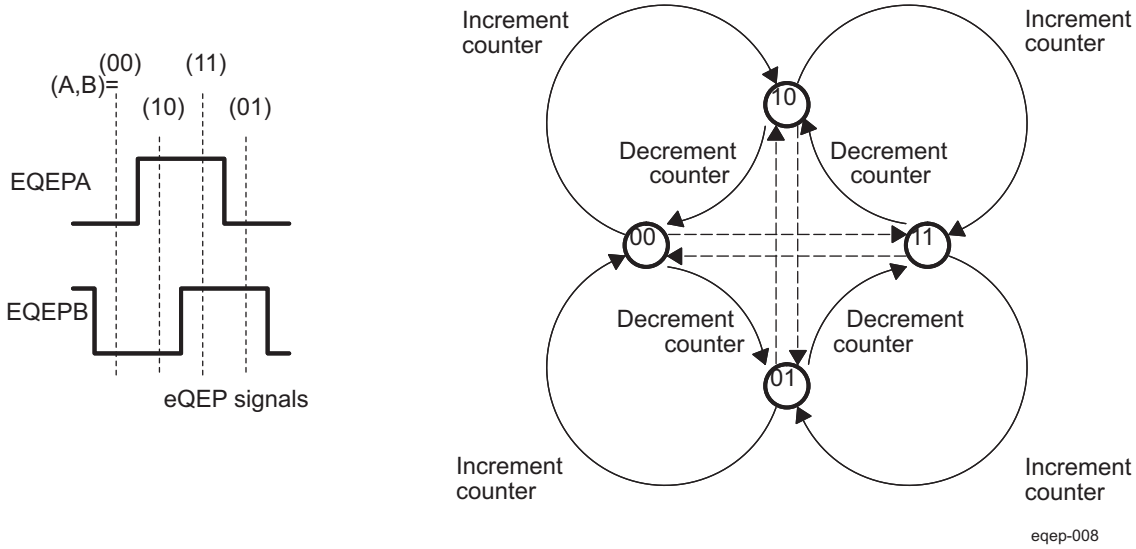
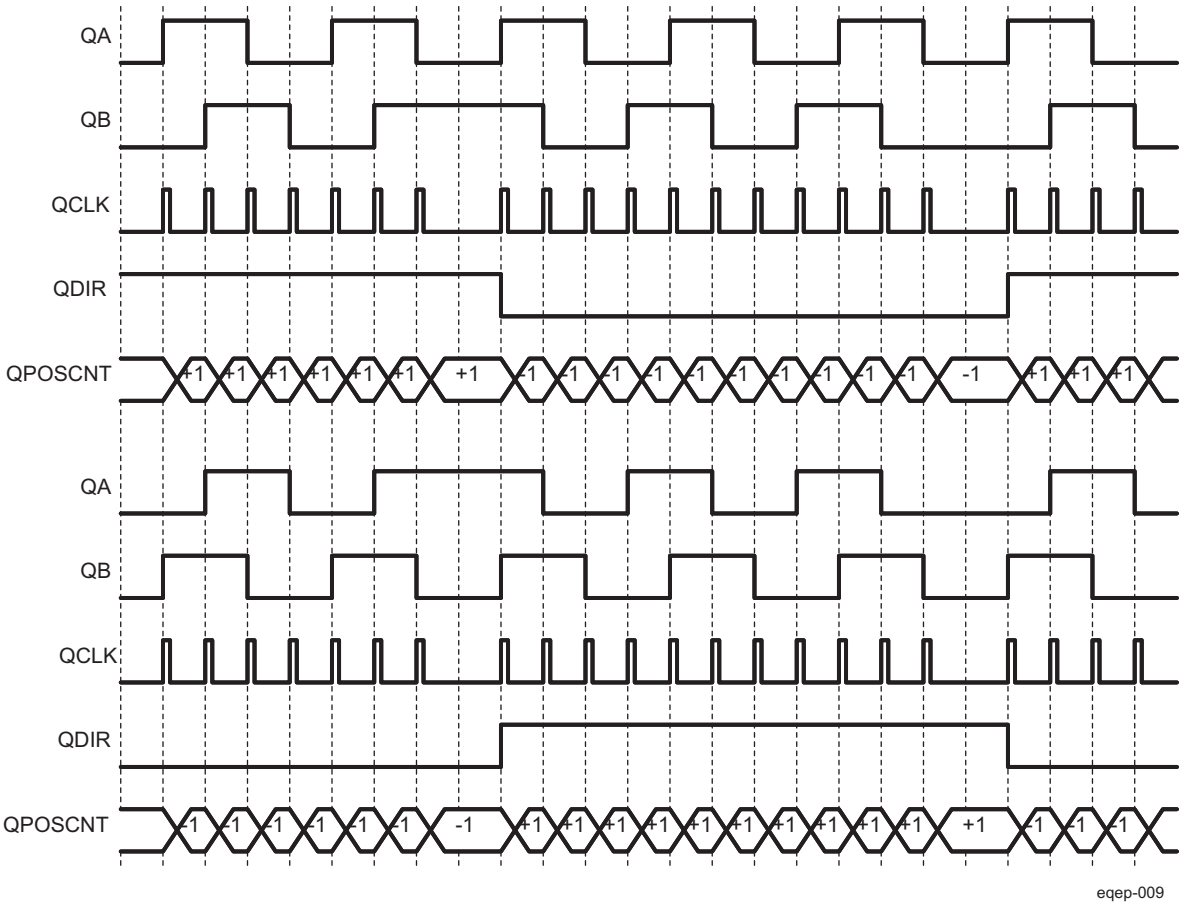


Figure 11-441. Quadrature-clock and Direction Decoding



11.6.4.2.1.2 eQEP Direction-count Mode

Some position encoders provide direction and clock outputs, instead of quadrature outputs. In such cases, direction-count mode can be used. EQEPA input will provide the clock for position counter and the EQEPB input will have the direction information. The position counter is incremented on every rising edge of a EQEPA input when the direction input is high and decremented when the direction input is low.

11.6.4.2.1.3 eQEP Up-Count Mode

The counter direction signal is hard-wired for up count and the position counter is used to measure the frequency of the EQEPA input. Setting of the XCR bit in the eQEP decoder control register (EQEP_QDECCTL) enables clock generation to the position counter on both edges of the QEPA input, thereby increasing the measurement resolution by 2× factor.

11.6.4.2.1.4 eQEP Down-Count Mode

The counter direction signal is hardwired for a down count and the position counter is used to measure the frequency of the EQEPA input. Setting of the XCR bit in the eQEP decoder control register (EQEP_QDECCTL) enables clock generation to the position counter on both edges of a EQEPA input, thereby increasing the measurement resolution by 2× factor.

11.6.4.2.2 eQEP Input Polarity Selection

Each eQEP input can be inverted using the in the eQEP decoder control register (EQEP_QDECCTL[8-5]) control bits. As an example, setting of the QIP bit in EQEP_QDECCTL inverts the index input.

11.6.4.2.3 eQEP Position-Compare Sync Output

The eQEP peripheral includes a position-compare unit that is used to generate the position-compare sync signal on compare match between the position counter register (EQEP_QPOSCNT) and the position-compare register (EQEP_QPOSCMP). This sync signal can be output using an index pin or strobe pin of the EQEP peripheral.

Setting the SOEN bit in the eQEP decoder control register (EQEP_QDECCTL) enables the position-compare sync output and the SPSEL bit in EQEP_QDECCTL selects either an eQEP index pin or an eQEP strobe pin.

11.6.4.3 eQEP Position Counter and Control Unit (PCCU)

The position counter and control unit provides two configuration registers (EQEP_QEPCTL and EQEP_QPOSCTL) for setting up position counter operational modes, position counter initialization/latch modes and position-compare logic for sync signal generation.

11.6.4.3.1 eQEP Position Counter Operating Modes

Position counter data may be captured in different manners. In some systems, the position counter is accumulated continuously for multiple revolutions and the position counter value provides the position information with respect to the known reference. An example of this is the quadrature encoder mounted on the motor controlling the print head in the printer. Here the position counter is reset by moving the print head to the home position and then position counter provides absolute position information with respect to home position.

In other systems, the position counter is reset on every revolution using index pulse and position counter provides rotor angle with respect to index pulse position.

Position counter can be configured to operate in following four modes

- Position Counter Reset on Index Event
- Position Counter Reset on Maximum Position
- Position Counter Reset on the first Index Event
- Position Counter Reset on Unit Time Out Event (Frequency Measurement).

In all the above operating modes, position counter is reset to 0 on overflow and to QPOSMAX bifield value in EQEP_QPOSMAX register on underflow. Overflow occurs when the position counter counts up after QPOSMAX value. Underflow occurs when position counter counts down after "0". Interrupt flag is set to indicate overflow/underflow in EQEP_QFLG register.

11.6.4.3.1.1 eQEP Position Counter Reset on Index Event (EQEP_QEPCTL[31-12] PCRM] = 0b00)

If the index event occurs during the forward movement, then position counter is reset to 0 on the next eQEP clock. If the index event occurs during the reverse movement, then the position counter is reset to the value in the EQEP_QPOSMAX register on the next eQEP clock.

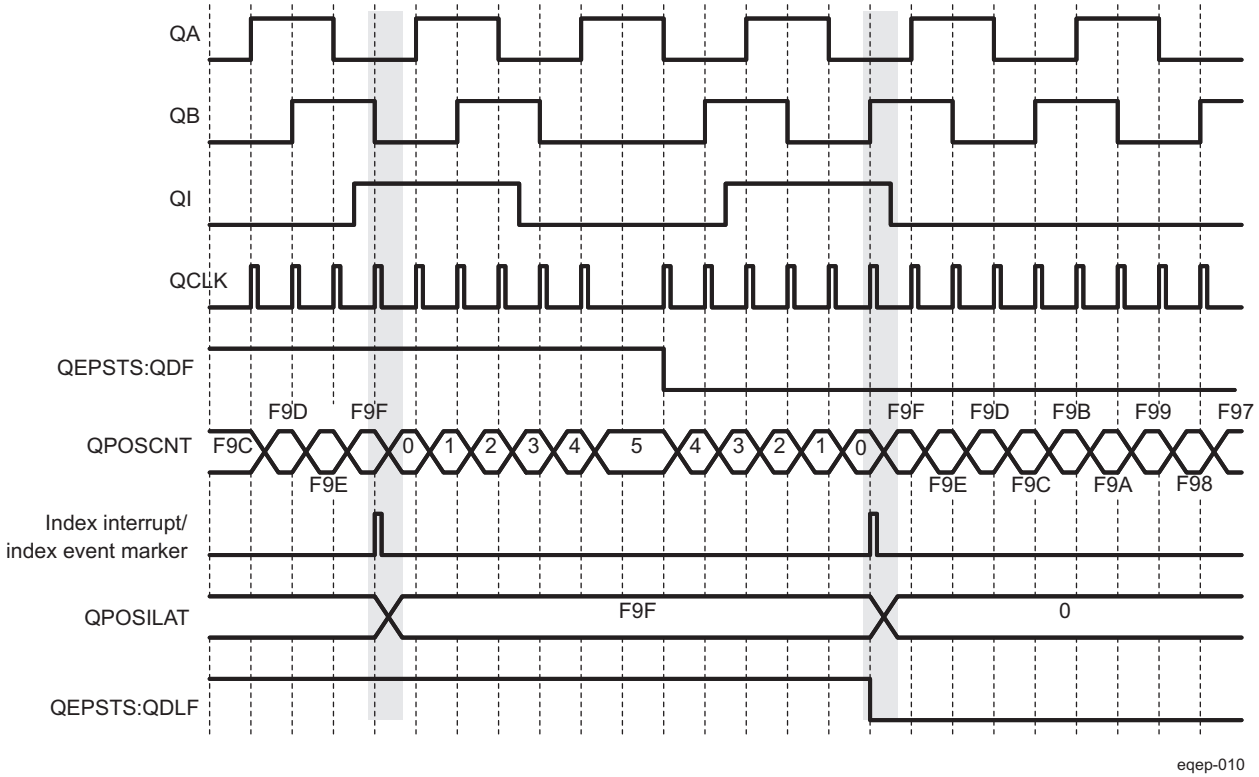
First index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (EQEP_QEPSTS[FIMF]) and direction on the first index event marker (EQEP_QEPSTS[FIDF]) in EQEP_QEPSTS registers, it also remembers the quadrature edge on the first index marker so that same relative quadrature transition is used for index event reset operation.

For example, if the first reset operation occurs on the falling edge of EQEPB during the forward direction, then all the subsequent reset must be aligned with the falling edge of EQEPB for the forward rotation and on the rising edge of EQEPB for the reverse rotation as shown in Figure 11-442.

The position-counter value is latched to the EQEP_QPOSILAT register and direction information is recorded in the EQEP_QEPSTS[QDLF] bit on every index event marker. The position-counter error flag (EQEP_QEPSTS[PCEF]) and error interrupt flag (EQEP_QFLG[PCE]) are set if the latched value is not equal to 0 or QPOSMAX. The position-counter error flag (EQEP_QEPSTS[PCEF]) is updated on every index event marker and an interrupt flag (EQEP_QFLG[PCE]) will be set on error that can be cleared only through software.

The index event latch configuration EQEP_QEPCTL[5-4] IEL bits are ignored in this mode and position counter error flag/interrupt flag are generated only in index event reset mode.

Figure 11-442. Position Counter Reset by Index Pulse for 1000 Line Encoder (QPOSMAX = 3999 or F9Fh)

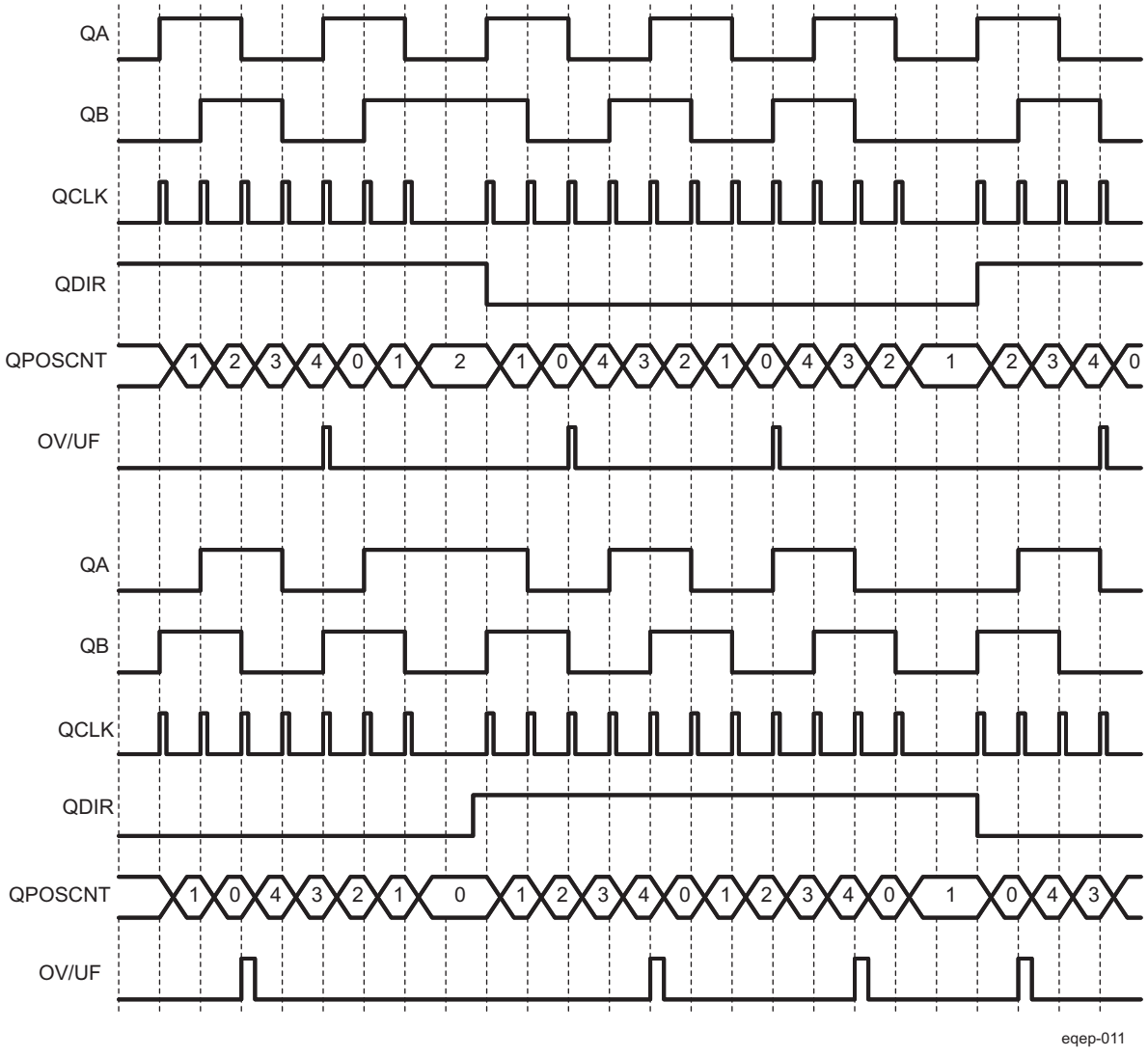


11.6.4.3.1.2 eQEP Position Counter Reset on Maximum Position (EQEP_QEPCTL[13-12] PCRM=0b01)

If the position counter is equal to QPOSMAX (in EQEP_QPOSMAX register), then the position counter is reset to 0 on the next eQEP clock for forward movement and position counter overflow flag is set. If the position counter is equal to ZERO, then the position counter is reset to QPOSMAX on the next QEP clock for reverse movement and position counter underflow flag is set. Figure 11-443 shows the position counter reset operation in this mode.

First index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (EQEP_QEPSTS[FIMF]) and direction on the first index event marker (EQEP_QEPSTS[FIDF]) in the EQEP_QEPSTS registers; it also remembers the quadrature edge on the first index marker so that the same relative quadrature transition is used for the software index marker (EQEP_QEPCTL[5-4] IEL=0b11).

Figure 11-443. Position Counter Underflow/Overflow (QPOSMAX = 4)



11.6.4.3.1.3 Position Counter Reset on the First Index Event (EQEP_QEPCTL[13-12] PCRM = 0b10)

If the index event occurs during forward movement, then the position counter is reset to 0 on the next eQEP clock. If the index event occurs during the reverse movement, then the position counter is reset to the value in the EQEP_QPOSMAX register on the next eQEP clock. Note that this is done only on the first occurrence and subsequently the position counter value is not reset on an index event; rather, it is reset based on maximum position as described in Section 11.6.4.3.1.2.

First index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (EQEP_QEPSTS[FIMF]) and direction on the first index event marker (EQEP_QEPSTS[FIDF]) in EQEP_QEPSTS registers. It also remembers the quadrature edge on the first index marker so that same relative quadrature transition is used for software index marker (EQEP_QEPCTL[5-4] IEL= 0b11).

11.6.4.3.1.4 Position Counter Reset on Unit Time out Event (EQEP_QEPCTL[13-12] PCRM = 0b11)

In this mode, the QPOSCNT value is latched to the EQEP_QPOSLAT register and then the QPOSCNT field is reset (to 0 or the QPOSMAX value in the EQEP_QPOSMAX register, depending on the direction mode selected by EQEP_QDECCTL[QSRC] bits on a unit time event). This is useful for frequency measurement.

11.6.4.3.2 eQEP Position Counter Latch

The eQEP index and strobe input can be configured to latch the position counter QPOSCNT (EQEP_QPOSCNT) into QPOSILAT (EQEP_QPOSILAT register) and QPOSSLAT (EQEP_QPOSSLAT register) bitfields, respectively, on occurrence of a definite event on these pins.

11.6.4.3.2.1 Index Event Latch

In some applications, it may not be desirable to reset the position counter on every index event and instead it may be required to operate the position counter in full 32-bit mode (EQEP_QEPCTL[13-12] PCRM = 0b01 and EQEP_QEPCTL[13-12] PCRM = 0b10 modes).

In such cases, the eQEP position counter can be configured to latch on the following events and direction information is recorded in the EQEP_QEPSTS[QDLF] bit on every index event marker.

- Latch on Rising edge (EQEP_QEPCTL[5-4] IEL = 0b01)
- Latch on Falling edge (EQEP_QEPCTL[5-4] IEL = 0b10)
- Latch on Index Event Marker (EQEP_QEPCTL[5-4] IEL = 0b11)

This is particularly useful as an error checking mechanism to check if the position counter accumulated the correct number of counts between index events. As an example, the 1000-line encoder must count 4000 times when moving in the same direction between the index events.

The index event latch interrupt flag (EQEP_QFLG[10] IEL) is set when the position counter is latched to the EQEP_QPOSILAT register. The index event latch configuration bits are ignored when EQEP_QEPCTL[13-12] PCRM = 0b00.

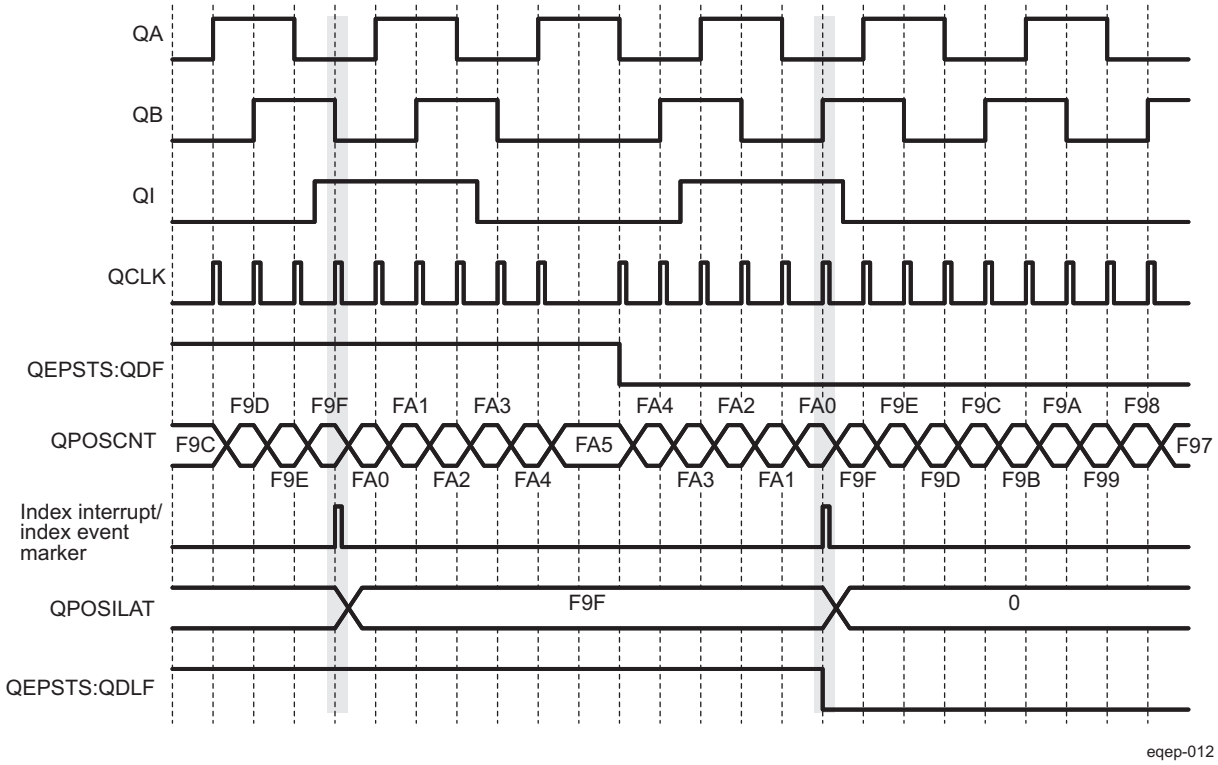
Latch on Rising Edge (EQEP_QEPCTL[5-4] IEL = 0b01) — The position counter value (QPOSCNT) is latched to the EQEP_QPOSILAT register on every rising edge of an index input.

Latch on Falling Edge (EQEP_QEPCTL[5-4] IEL = 0b10) — The position counter value (QPOSCNT) is latched to the EQEP_QPOSILAT register on every falling edge of index input.

Latch on Index Event Marker/Software Index Marker (EQEP_QEPCTL[5-4] IEL = 0b11) — The first index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (EQEP_QEPSTS[FIMF]) and direction on the first index event marker (EQEP_QEPSTS[FIDF]) in the EQEP_QEPSTS registers. It also remembers the quadrature edge on the first index marker so that same relative quadrature transition is used for latching the position counter (EQEP_QEPCTL[5-4] IEL = 0b11).

Figure 11-444 shows the position counter latch using an index event marker.

Figure 11-444. Software Index Marker for 1000-line Encoder (EQEP_QEPCTL[5-4] IEL = 0b01)



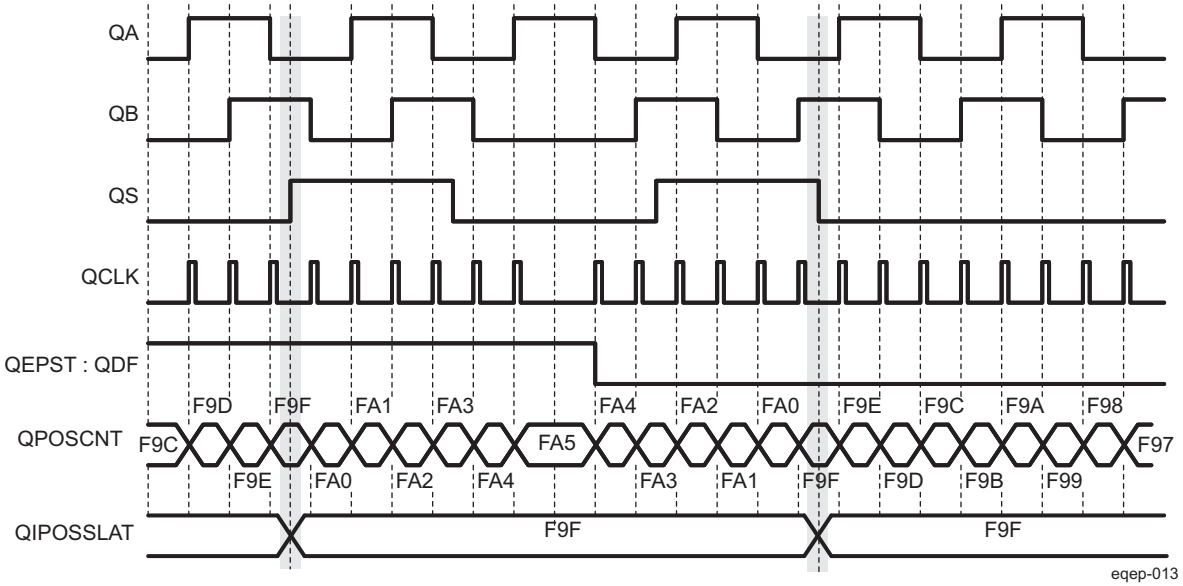
11.6.4.3.2.2 eQEP Strobe Event Latch

The position-counter value is latched to the EQEP_QPOSSLAT register on the rising edge of the strobe input by clearing the EQEP_QEPCTL[6] SEL bit.

If the EQEP_QEPCTL[6] SEL bit is set, then the position counter value is latched to the EQEP_QPOSSLAT register on the rising edge of the strobe input for forward direction and on the falling edge of the strobe input for reverse direction as shown in Figure 11-445.

The strobe event latch interrupt flag (EQEP_QFLG[SEL]) is set when the position counter is latched to the EQEP_QPOSSLAT register.

Figure 11-445. eQEP Strobe Event Latch (EQEP_QEPCTL[6] SEL = 0b1)



11.6.4.3.3 eQEP Position Counter Initialization

The position counter can be initialized using following events:

- Index event
- Strobe event
- Software initialization.

Index Event Initialization (IEI) — The eQEPx_I index input can be used to trigger the initialization of the position counter at the rising or falling edge of the index input.

If the EQEP_QEPCTL[9-8] IEI bits are 0b10, then the position counter (EQEP_QPOSCNT) is initialized with a value in the EQEP_QPOSINIT register on the rising edge of strobe input for forward direction and on the falling edge of strobe input for reverse direction.

The index event initialization interrupt flag (EQEP_QFLG[IEI]) is set when the position counter is initialized with a value in the EQEP_QPOSINIT register.

Strobe Event Initialization (SEI) — If the EQEP_QEPCTL[11-10] SEI bits are 0b10, then the position counter is initialized with a value in the EQEP_QPOSINIT register on the rising edge of strobe input.

If the EQEP_QEPCTL[11-10] SEI bits are 0b11, then the position counter (EQEP_QPOSCNT) is initialized with a value in the EQEP_QPOSINIT register on the rising edge of strobe input for forward direction and on the falling edge of strobe input for reverse direction.

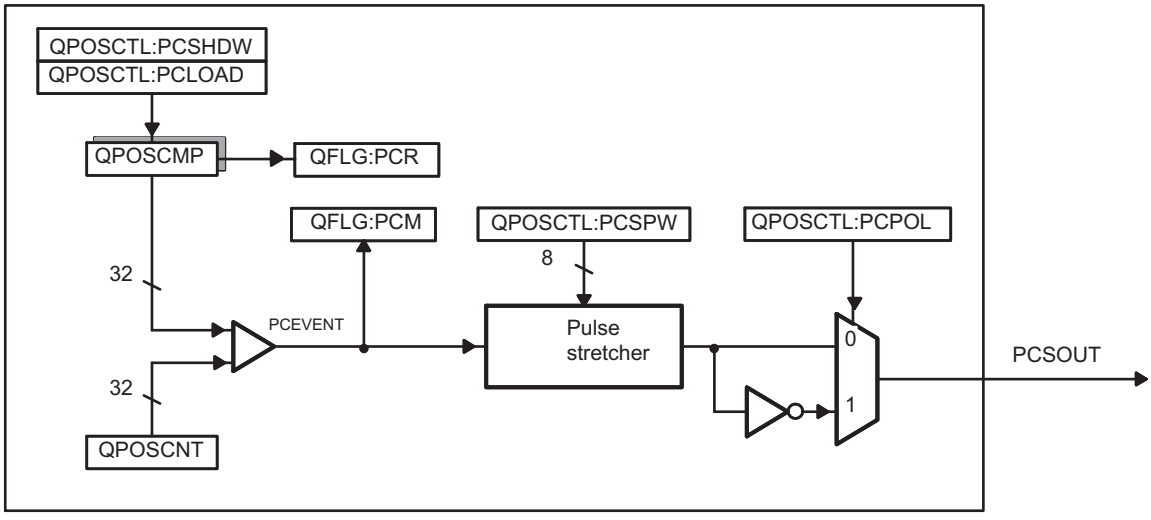
The strobe event initialization interrupt flag (EQEP_QFLG[SEI]) is set when the position counter is initialized with a value in the EQEP_QPOSINIT register.

Software Initialization (SWI) — The position counter can be initialized in software by writing a '1' to the EQEP_QEPCTL[7] SWI bit, which will automatically be cleared after initialization.

11.6.4.3.4 eQEP Position-Compare Unit

The eQEP peripheral includes a position-compare unit that is used to generate a sync output and/or interrupt on a position-compare match. Figure 11-446 shows a diagram. The position-compare (EQEP_QPOSCMP) register is shadowed and shadow mode can be enabled or disabled using the EQEP_QPOSCTL[PSSHDW] bit. If the shadow mode is not enabled, the CPU writes directly to the active position compare register.

Figure 11-446. eQEP Position-compare Unit



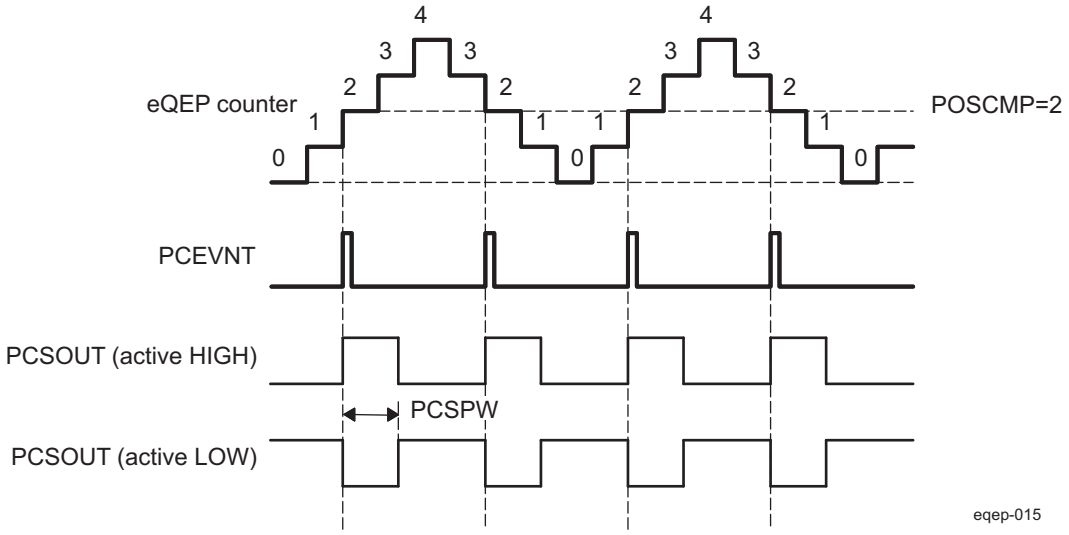
In shadow mode, SW can configure the position-compare unit (EQEP_QPOSCTL[PCLOAD]) to load the shadow register value into the active register on the following events and to generate the position-compare ready (EQEP_QFLG[PCR]) interrupt after loading.

- Load on compare match
- Load on position-counter zero event.

The position-compare match (EQEP_QFLG[PCM]) is set when the position-counter value (QPOSCNT) matches with the active position-compare register (EQEP_QPOSCMP) and the position-compare sync output of the programmable pulse width is generated on compare match to trigger an external device.

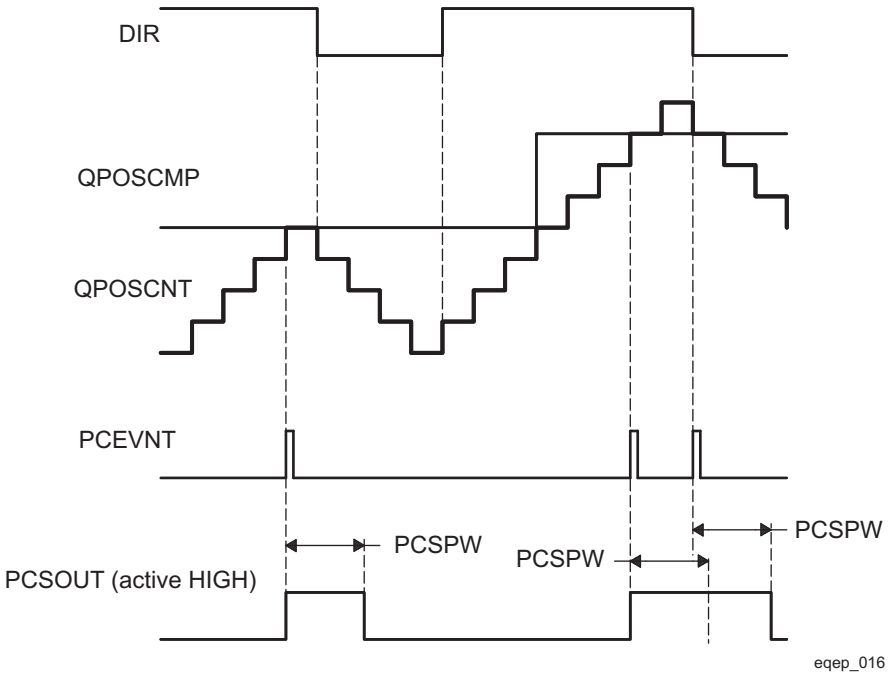
For example, if EQEP_QPOSCMP bitfield QPOSCMP = 0x2, the position-compare unit generates a position-compare event on 1 to 2 transitions of the eQEP position counter for forward counting direction and on 3 to 2 transitions of the eQEP position counter for reverse counting direction (see Figure 11-447).

Figure 11-447. eQEP Position-compare Event Generation Points



The pulse stretcher logic in the position-compare unit generates a programmable position-compare sync pulse output on the position-compare match. In the event of a new position-compare match while a previous position-compare pulse is still active, then the pulse stretcher generates a pulse of specified duration from the new position-compare event as shown in Figure 11-448.

Figure 11-448. eQEP Position-compare Sync Output Pulse Stretcher



11.6.4.4 eQEP Edge Capture Unit

The eQEP peripheral includes an integrated edge capture unit to measure the elapsed time between the unit position events as shown in Figure 11-449. This feature is typically used for low speed measurement using the following equation:

$$V(k) = \frac{X}{t(k) - t(k-1)} = \frac{X}{\Delta T}$$

eqep-017 (7)

where,

- X - Unit position is defined by integer multiple of quadrature edges (see Figure 11-450)
- ΔT - Elapsed time between unit position events
- v(k) - Velocity at time instant "k"

The eQEP capture timer (QCTMR bitfield in EQEP_QCTMR register) runs from prescaled SYSCLKOUT and the prescaler is programmed by the EQEP_QCAPCTL[CCPS] bits. The capture timer QCTMR value is latched into the capture period register (EQEP_QCPRD) on every unit position event and then the capture timer is reset, a flag is set in EQEP_QEPSTS[UPEVNT] to indicate that new value is latched into the EQEP_QCPRD register. Software can check this status flag before reading the period register for low speed measurement and clear the flag by writing 1.

NOTE: The system clock — SYSCLKOUT is the eQEP functional clock derived from the PWMSSn gateable interface and functional clock PWMSSn_GICLK, described in Section 11.6.3, *eQEP Integration*.

Time measurement (ΔT) between unit position events will be correct if the following conditions are met:

- No more than 65,535 counts have occurred between unit position events.
- No direction change between unit position events.

The capture unit sets the eQEP overflow error flag (EQEP_QEPSTS[COEF]) in the event of capture timer overflow between unit position events. If a direction change occurs between the unit position events, then an error flag is set in the status register (EQEP_QEPSTS[CDEF]).

Capture Timer (EQEP_QCTMR register) and Capture period register (EQEP_QCPRD) can be configured to latch on following events.

- CPU read of EQEP_QPOSCNT register
- Unit time-out event

If the EQEP_QEPCTL[2] QCLM bit is cleared, then the capture timer and capture period values are latched into the EQEP_QCTMRLAT and EQEP_QCPRDLAT registers, respectively, when the CPU reads the position counter in EQEP_QPOSCNT.

If the EQEP_QEPCTL[2] QCLM bit is set, then the position counter, capture timer, and capture period values are latched into the EQEP_QPOSLAT, EQEP_QCTMRLAT and EQEP_QCPRDLAT registers, respectively, on unit time out.

Figure 11-451 shows the capture unit operation along with the position counter.

NOTE: The EQEP_QCAPCTL register should not be modified dynamically (such as switching CAPCLK prescaling mode from SYSCLKOUT/4 to SYSCLKOUT/8, where SYSCLKOUT is equivalent to EQEP_x_VBUS_CLK). The capture unit must be disabled before changing the prescaler.

Figure 11-449. eQEP Edge Capture Unit

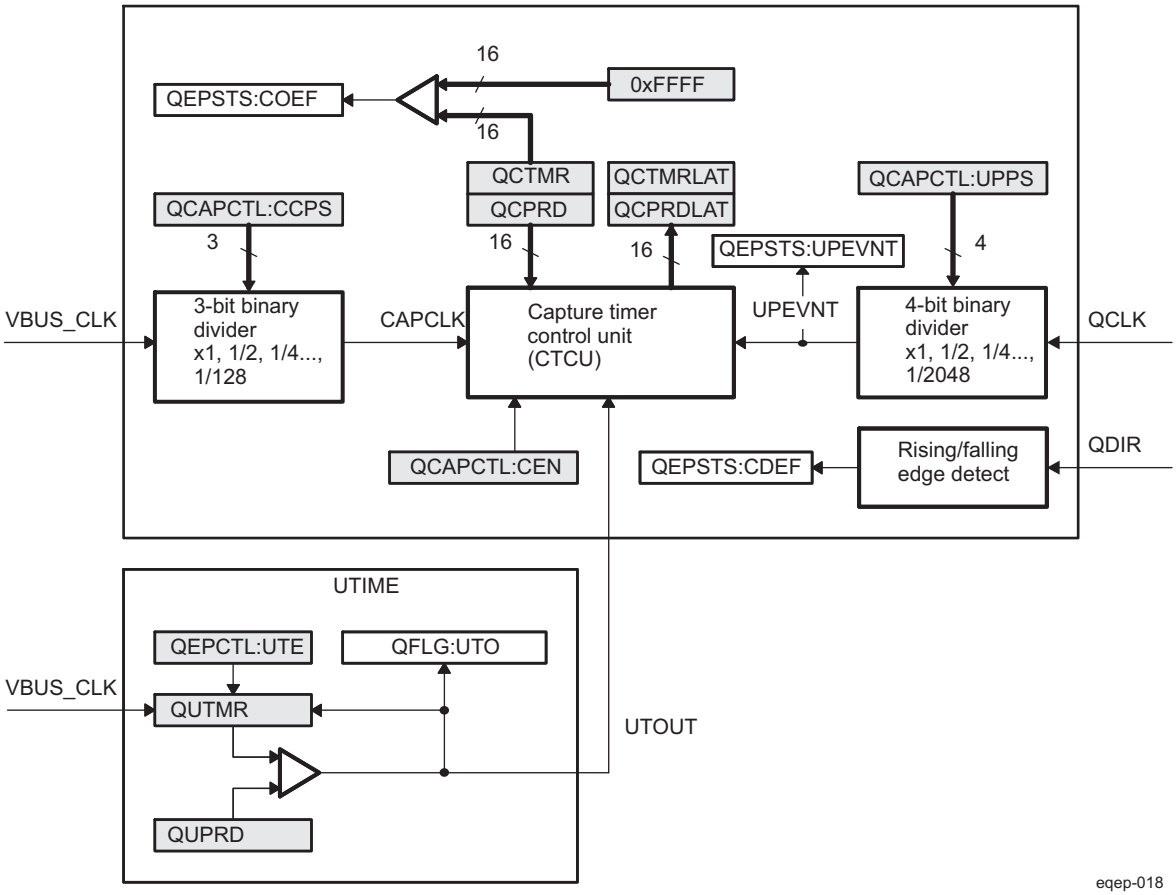
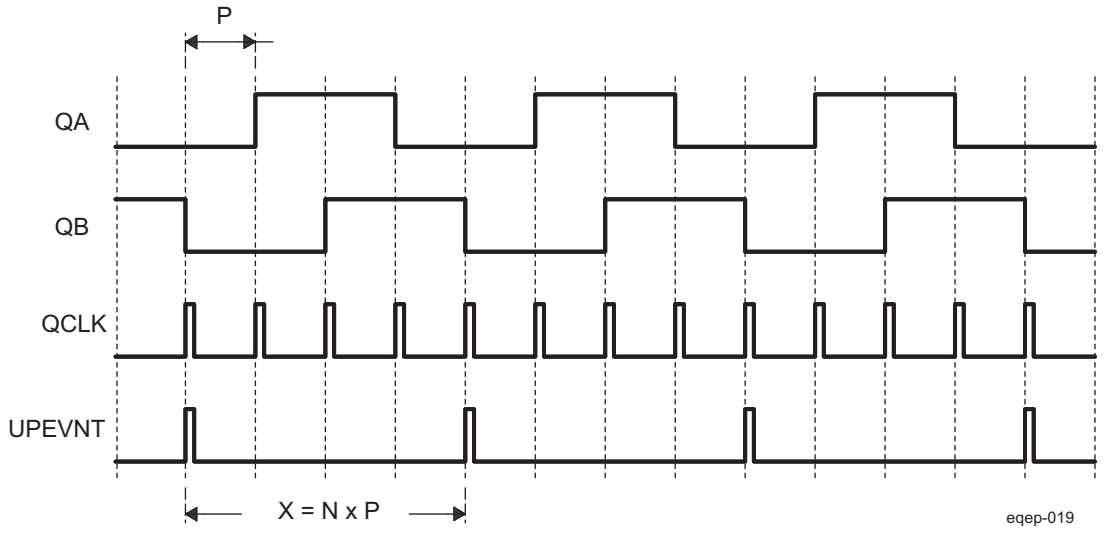
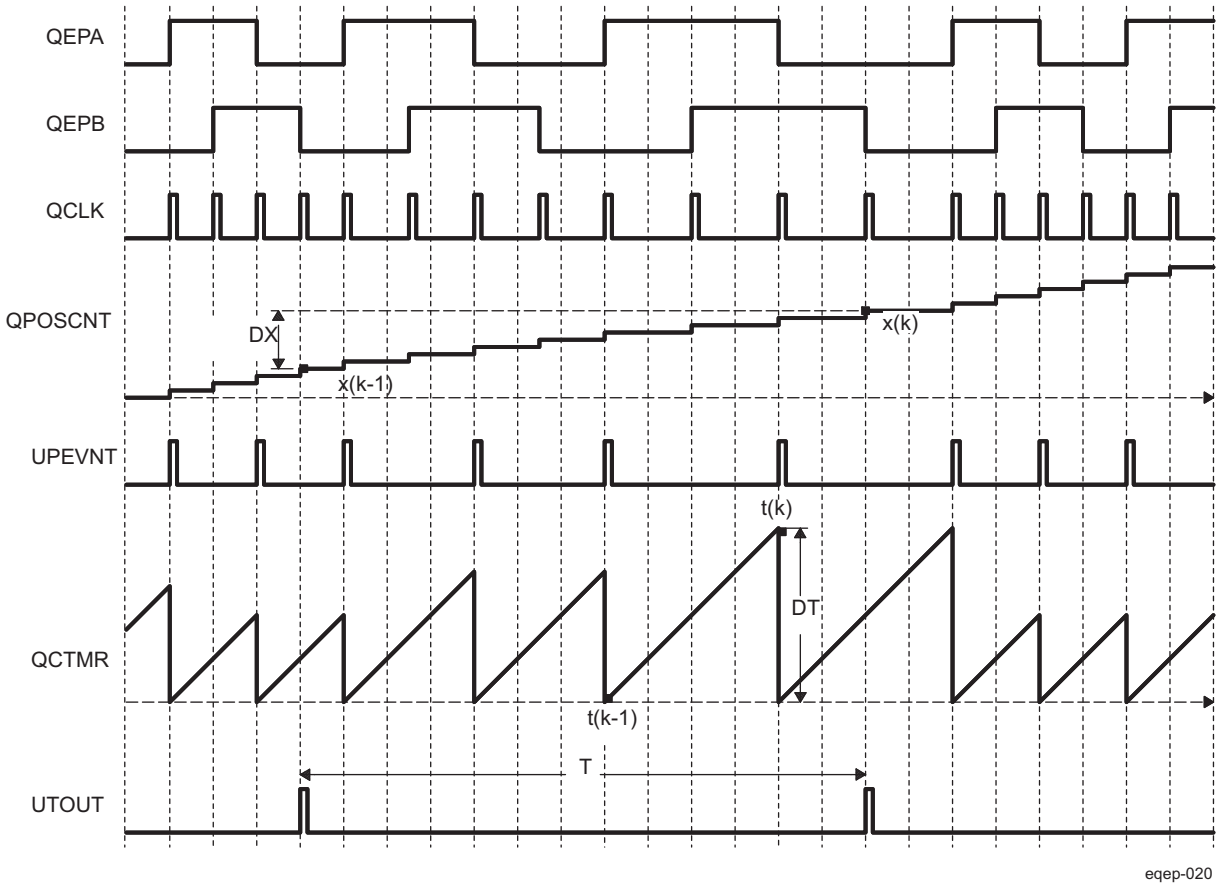


Figure 11-450. Unit Position Event for Low Speed Measurement (EQEP_QCAPCTL[UPPS] = 0010)



N - Number of quadrature periods selected using EQEP_QCAPCTL[UPPS] bits

Figure 11-451. eQEP Edge Capture Unit - Timing Details



Velocity Calculation Equations:

$$V(k) = \frac{x(k) - x(k-1)}{T} = \frac{\Delta X}{T} \quad \text{or}$$

$$V(k) = \frac{X}{t(k) - t(k-1)} = \frac{X}{\Delta T}$$

eqep_021 (8)

where

v(k): Velocity at time instant k

x(k): Position at time instant k

x(k-1): Position at time instant k - 1

T: Fixed unit time or inverse of velocity calculation rate

ΔX: Incremental position movement in unit time

X: Fixed unit position

ΔT: Incremental time elapsed for unit position movement

t(k): Time instant "k"

t(k-1): Time instant "k - 1"

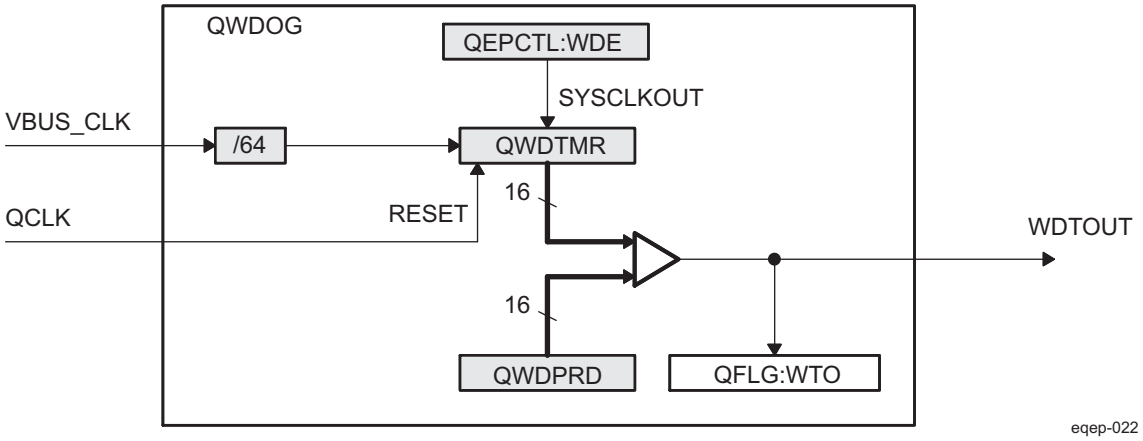
Unit time (T) and unit period (X) are configured using the EQEP_QUPRD and EQEP_QCAPCTL[UPPS] registers. Incremental position output and incremental time output is available in the EQEP_QPOSLAT and EQEP_QCPRDLAT registers.

Parameter	Relevant Register to Configure or Read the Information
T	Unit Period Register (EQEP_QUPRD)
ΔX	Incremental Position = QPOSLAT(k) - QPOSLAT(k - 1)
X	Fixed unit position defined by sensor resolution and ZCAPCTL[UPPS] bits
ΔT	Capture Period Latch (QCPRDLAT)

11.6.4.5 eQEP Watchdog

The eQEP peripheral contains a 16-bit watchdog timer that monitors the quadrature-clock to indicate proper operation of the motion-control system. The eQEP watchdog timer is clocked from SYSCLKOUT/64 and the quadrate clock event (pulse) resets the watchdog timer. If no quadrature-clock event is detected until a period match (QWDPRD = QWDTMR), then the watchdog timer will time out and the watchdog interrupt flag will be set (EQEP_QFLG[WTO]). The time-out value is programmable through the watchdog period register (EQEP_QWDPRD).

Figure 11-452. eQEP Watchdog Timer

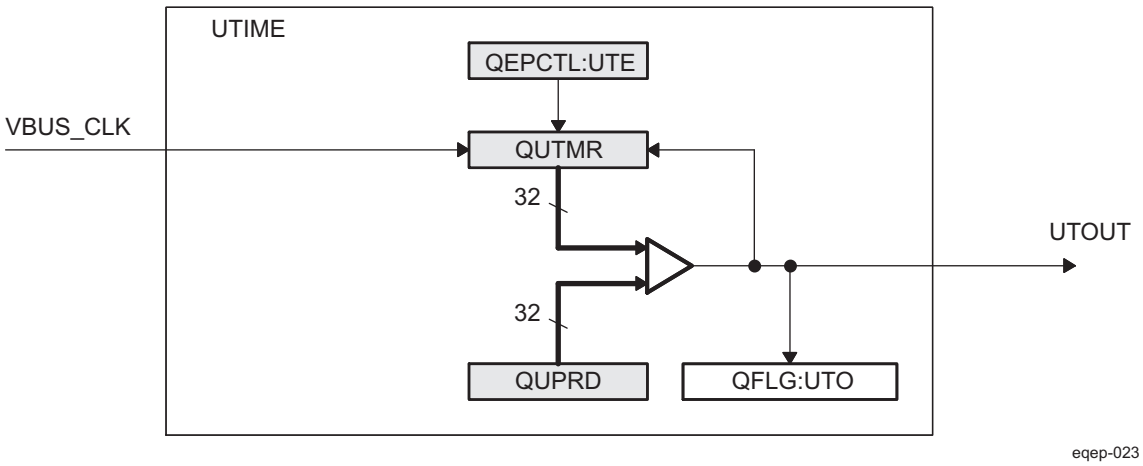


11.6.4.6 Unit Timer Base

The eQEP peripheral includes a 32-bit timer (QUTMR) that is clocked by SYSCLKOUT to generate periodic interrupts for velocity calculations. The unit time out interrupt is set (EQEP_QFLG[UTO]) when the unit timer (QUTMR) matches the unit period register (EQEP_QUPRD).

The eQEP peripheral can be configured to latch the position counter, capture timer, and capture period values on a unit time out event so that latched values are used for velocity calculation as described in Section Section 11.6.4.4.

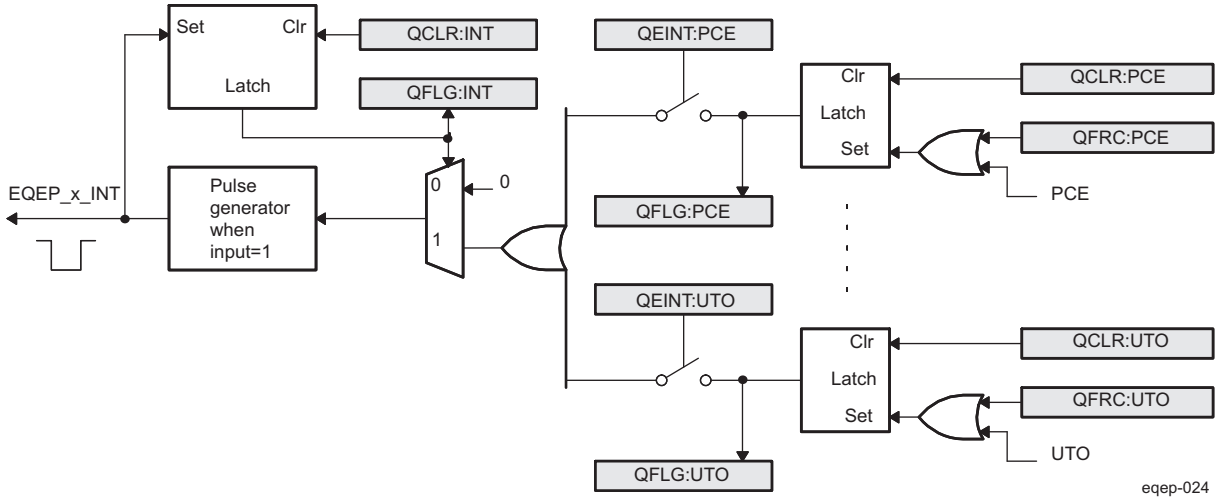
Figure 11-453. eQEP Unit Time Base



11.6.4.7 eQEP Interrupt Structure

Figure 11-454 shows how the interrupt mechanism works in the EQEP module.

Figure 11-454. EQEP Interrupt Generation



Eleven interrupt events (PCE, PHE, QDC, WTO, PCU, PCO, PCR, PCM, SEL, IEL, and UTO) can be generated. The interrupt control register (EQEP_QEINT) is used to enable/disable individual interrupt event sources. The interrupt flag register (EQEP_QFLG) indicates if any interrupt event has been latched and contains the global interrupt flag bit (INT). An interrupt pulse is generated only to the interrupt controller if any of the interrupt events is enabled, the flag bit is 1 and the INT flag bit is 0. The interrupt service routine will need to clear the global interrupt flag bit and the serviced event, via the interrupt clear register (EQEP_QCLR), before any other interrupt pulses are generated. SW can force an interrupt event by way of the interrupt force register (EQEP_QFRC), which is useful for test purposes.

11.6.4.8 Summary of eQEP Functional Registers

Table 11-929 lists the registers with their memory locations, sizes, and reset values.

Table 11-929. eQEP Control and Status Functional Registers

Offset	Acronym	Register Description	Size (×16)/ #shadow
0h	EQEP_QPOSCNT	eQEP Position Counter Register	2/0
4h	EQEP_QPOSINIT	eQEP Position Counter Initialization Register	2/0
8h	EQEP_QPOSMAX	eQEP Maximum Position Count Register	2/0
Ch	EQEP_QPOSCMP	eQEP Position-Compare Register	2/1
10h	EQEP_QPOSILAT	eQEP Index Position Latch Register	2/0
14h	EQEP_QPOSSLAT	eQEP Strobe Position Latch Register	2/0
18h	EQEP_QPOSLAT	eQEP Position Counter Latch Register	2/0
1Ch	EQEP_QUTMR	eQEP Unit Timer Register	2/0
20h	EQEP_QUPRD	eQEP Unit Period Register	2/0
24h	EQEP_QWDTMR	eQEP Watchdog Timer Register	1/0
26h	EQEP_QWDPRD	eQEP Watchdog Period Register	1/0
28h	EQEP_QDECCTL	eQEP Decoder Control Register	1/0
2Ah	EQEP_QEPCTL	eQEP Control Register	1/0
2Ch	EQEP_QCAPCTL	eQEP Capture Control Register	1/0
2Eh	EQEP_QPOSCTL	eQEP Position-Compare Control Register	1/0
30h	EQEP_QEINT	eQEP Interrupt Enable Register	1/0
32h	EQEP_QFLG	eQEP Interrupt Flag Register	1/0
34h	EQEP_QCLR	eQEP Interrupt Clear Register	1/0
36h	EQEP_QFRC	eQEP Interrupt Force Register	1/0
38h	EQEP_QEPSTS	eQEP Status Register	1/0
3Ah	EQEP_QCTMR	eQEP Capture Timer Register	1/0
3Ch	EQEP_QCPRD	eQEP Capture Period Register	1/0
3Eh	EQEP_QCTMRLAT	eQEP Capture Timer Latch Register	1/0
40h	EQEP_QCPRDLAT	eQEP Capture Period Latch Register	1/0
5Ch	EQEP_REVID	eQEP Revision ID Register	2/0

11.6.5 eQEP Registers

Table 11-931 lists the memory-mapped registers for the eQEP. All register offset addresses not listed in Table 11-931 should be considered as reserved locations and the register contents should not be modified.

This section describes the EQEP_0 instances registers.

Table 11-930. eQEP Instances

Instance	Base Address
EQEP_0	021C 0000h
EQEP_1	021C 0400h
EQEP_2	021C 0800h

Table 11-931. eQEP Registers

Offset	Acronym	Register Name	EQEP_0 Physical Address	EQEP_1 Physical Address	EQEP_2 Physical Address	Section
0h	EQEP_QPOSCNT	Position Counter Register	021C 0000h	021C 0400h	021C 0800h	Section 11.6.5.1
4h	EQEP_QPOSINIT	Initialization Position Counter Register	021C 0004h	021C 0404h	021C 0804h	Section 11.6.5.2
8h	EQEP_QPOSMAX	Maximum Position Count Register	021C 0008h	021C 0408h	021C 0808h	Section 11.6.5.3
Ch	EQEP_QPOSCMP	Position Compare Register	021C 000Ch	021C 040Ch	021C 080Ch	Section 11.6.5.4
10h	EQEP_QPOSILAT	Index Position Latch Register	021C 0010h	021C 0410h	021C 0810h	Section 11.6.5.5
14h	EQEP_QPOSSLAT	Strobe Position Latch Register	021C 0014h	021C 0414h	021C 0814h	Section 11.6.5.6
18h	EQEP_QPOSLAT	Position Latch Register	021C 0018h	021C 0418h	021C 0818h	Section 11.6.5.7
1Ch	EQEP_QUTMR	QEP Unit Timer	021C 001Ch	021C 041Ch	021C 081Ch	Section 11.6.5.8
20h	EQEP_QUPRD	QEP Unit Timer Period Register	021C 0020h	021C 0420h	021C 0820h	Section 11.6.5.9
24h	EQEP_QWDTMR	QEP Watchdog Timer	021C 0024h	021C 0424h	021C 0824h	Section 11.6.5.10
26h	EQEP_QWDPRD	QEP Watchdog Period Register	021C 0026h	021C 0426h	021C 0826h	Section 11.6.5.11
28h	EQEP_QDECCTL	Quadrature Decoder Control Register	021C 0028h	021C 0428h	021C 0828h	Section 11.6.5.12
2Ah	EQEP_QEPCTL	QEP Control Register	021C 002Ah	021C 042Ah	021C 082Ah	Section 11.6.5.13
2Ch	EQEP_QCAPCTL	Quadrature Capture Control Register	021C 002Ch	021C 042Ch	021C 082Ch	Section 11.6.5.14
2Eh	EQEP_QPOSCTL	Position Compare Control Register	021C 002Eh	021C 042Eh	021C 082Eh	Section 11.6.5.15
30h	EQEP_QEINT	QEP Interrupt Control Register	021C 0030h	021C 0430h	021C 0830h	Section 11.6.5.16
32h	EQEP_QFLG	QEP Interrupt Flag Register	021C 0032h	021C 0432h	021C 0832h	Section 11.6.5.17
34h	EQEP_QCLR	QEP Interrupt Clear Register	021C 0034h	021C 0434h	021C 0834h	Section 11.6.5.18
36h	EQEP_QFRC	QEP Interrupt Force Register	021C 0036h	021C 0436h	021C 0836h	Section 11.6.5.19
38h	EQEP_QEPSTS	QEP Status Register	021C 0038h	021C 0438h	021C 0838h	Section 11.6.5.20

Table 11-931. eQEP Registers (continued)

Offset	Acronym	Register Name	EQEP_0 Physical Address	EQEP_1 Physical Address	EQEP_2 Physical Address	Section
3Ah	EQEP_QCTMR	QEP Capture Timer Register	021C 003Ah	021C 043Ah	021C 083Ah	Section 11.6.5.21
3Ch	EQEP_QCPRD	QEP Capture Period Register	021C 003Ch	021C 043Ch	021C 083Ch	Section 11.6.5.22
3Eh	EQEP_QCTMRLAT	QEP Capture Timer Latch Register	021C 003Eh	021C 043Eh	021C 083Eh	Section 11.6.5.23
40h	EQEP_QCPRDLAT	QEP Capture Period Latch Register	021C 0040h	021C 0440h	021C 0840h	Section 11.6.5.24
5Ch	EQEP_REVID	QEP Peripheral ID Register	021C 005Ch	021C 045Ch	021C 085Ch	Section 11.6.5.25

11.6.5.1 EQEP_QPOSCNT Register (Offset = 0h) [reset = 0h]

EQEP_QPOSCNT is shown in Figure 11-455 and described in Table 11-933.

Table 11-932. EQEP_QPOSCNT Instances

Instance	Physical Address
EQEP_0	021C 0000h
EQEP_1	021C 0400h
EQEP_2	021C 0800h

Figure 11-455. EQEP_QPOSCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSCNT																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-933. EQEP_QPOSCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSCNT	R/W	0h	This 32 bit position counter register counts up/down on every eQEP pulse based on direction input. This counter acts as a position integrator whose count value is proportional to position from a give reference point.

Table 11-934. Register Call Summary for EQEP_QPOSCNT

eQEP Registers • eQEP Registers: [0] • EQEP_QEPCTL Register (Offset = 2Ah) [reset = 0h]: [0][1] • EQEP_QPOSCMP Register (Offset = Ch) [reset = 0h]: [0] • EQEP_QPOSCNT Register (Offset = 0h) [reset = 0h]: [0]
eQEP Functional Description • eQEP Position-Compare Sync Output: [0] • eQEP Position Counter Latch: [0] • Summary of eQEP Functional Registers: [0] • eQEP Position Counter Initialization: [0][1] • eQEP Edge Capture Unit: [0][1]

11.6.5.2 EQEP_QPOSINIT Register (Offset = 4h) [reset = 0h]

EQEP_QPOSINIT is shown in Figure 11-456 and described in Table 11-936.

Table 11-935. EQEP_QPOSINIT Instances

Instance	Physical Address
EQEP_0	021C 0004h
EQEP_1	021C 0404h
EQEP_2	021C 0804h

Figure 11-456. EQEP_QPOSINIT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSINIT																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-936. EQEP_QPOSINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSINIT	R/W	0h	This register contains the position value that is used to initialize the position counter based on external strobe or index event. The position counter can be initialized through software.

Table 11-937. Register Call Summary for EQEP_QPOSINIT

eQEP Registers • eQEP Registers: [0] • EQEP_QPOSINIT Register (Offset = 4h) [reset = 0h]: [0]
eQEP Functional Description • eQEP Position Counter Initialization: [0][1][2][3][4] • Summary of eQEP Functional Registers: [0]

11.6.5.3 EQEP_QPOSMAX Register (Offset = 8h) [reset = 0h]

EQEP_QPOSMAX is shown in Figure 11-457 and described in Table 11-939.

Table 11-938. EQEP_QPOSMAX Instances

Instance	Physical Address
EQEP_0	021C 0008h
EQEP_1	021C 0408h
EQEP_2	021C 0808h

Figure 11-457. EQEP_QPOSMAX Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSMAX																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-939. EQEP_QPOSMAX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSMAX	R/W	0h	This register contains the maximum position counter value.

Table 11-940. Register Call Summary for EQEP_QPOSMAX

eQEP Registers • EQEP_QPOSMAX Register (Offset = 8h) [reset = 0h]: [0] • eQEP Registers: [0]
eQEP Functional Description • eQEP Position Counter Operating Modes: [0] • eQEP Position Counter Reset on Index Event (EQEP_QEPCTL[31-12] PCRM] = 0b00): [0] • eQEP Position Counter Reset on Maximum Position (EQEP_QEPCTL[13-12] PCRM=0b01): [0] • Position Counter Reset on Unit Time out Event (EQEP_QEPCTL[13-12] PCRM = 0b11): [0] • Position Counter Reset on the First Index Event (EQEP_QEPCTL[13-12] PCRM = 0b10): [0] • Summary of eQEP Functional Registers: [0]

11.6.5.4 EQEP_QPOSCMP Register (Offset = Ch) [reset = 0h]

EQEP_QPOSCMP is shown in Figure 11-458 and described in Table 11-942.

Table 11-941. EQEP_QPOSCMP Instances

Instance	Physical Address
EQEP_0	021C 000Ch
EQEP_1	021C 040Ch
EQEP_2	021C 080Ch

Figure 11-458. EQEP_QPOSCMP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSCMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-942. EQEP_QPOSCMP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSCMP	R/W	0h	The position-compare value in this register is compared with the position counter (EQEP_QPOSCNT[31-0] QPOSCNT) to generate sync output and/or interrupt on compare match.

Table 11-943. Register Call Summary for EQEP_QPOSCMP

eQEP Registers • eQEP Registers: [0] • EQEP_QPOSCMP Register (Offset = Ch) [reset = 0h]: [0]
eQEP Functional Description • eQEP Position-Compare Sync Output: [0] • Summary of eQEP Functional Registers: [0] • eQEP Position-Compare Unit: [0][1][2]

11.6.5.5 EQEP_QPOSILAT Register (Offset = 10h) [reset = 0h]

EQEP_QPOSILAT is shown in Figure 11-459 and described in Table 11-945.

Table 11-944. EQEP_QPOSILAT Instances

Instance	Physical Address
EQEP_0	021C 0010h
EQEP_1	021C 0410h
EQEP_2	021C 0810h

Figure 11-459. EQEP_QPOSILAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSILAT																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-945. EQEP_QPOSILAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSILAT	R	0h	The position-counter value is latched into this register on an index event as defined by the EQEP_QEPCTL[5-4] IEL bits.

Table 11-946. Register Call Summary for EQEP_QPOSILAT

eQEP Registers • eQEP Registers: [0] • EQEP_QPOSILAT Register (Offset = 10h) [reset = 0h]: [0] • EQEP_QEPCTL Register (Offset = 2Ah) [reset = 0h]: [0]
eQEP Functional Description • eQEP Position Counter Latch: [0] • Index Event Latch: [0][1][2] • eQEP Position Counter Reset on Index Event (EQEP_QEPCTL[31-12] PCRM] = 0b00): [0] • Summary of eQEP Functional Registers: [0]

11.6.5.6 EQEP_QPOSSLAT Register (Offset = 14h) [reset = 0h]

EQEP_QPOSSLAT is shown in Figure 11-460 and described in Table 11-948.

Table 11-947. EQEP_QPOSSLAT Instances

Instance	Physical Address
EQEP_0	021C 0014h
EQEP_1	021C 0414h
EQEP_2	021C 0814h

Figure 11-460. EQEP_QPOSSLAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSSLAT																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-948. EQEP_QPOSSLAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSSLAT	R	0h	The position-counter value is latched into this register on strobe event as defined by the EQEP_QEPCTL[6] SEL bit.

Table 11-949. Register Call Summary for EQEP_QPOSSLAT

eQEP Registers • EQEP_QFLG Register (Offset = 32h) [reset = 0h]: [0] • eQEP Registers: [0] • EQEP_QPOSSLAT Register (Offset = 14h) [reset = 0h]: [0]
eQEP Functional Description • eQEP Position Counter Latch: [0] • eQEP Strobe Event Latch: [0][1][2] • Summary of eQEP Functional Registers: [0]

11.6.5.7 EQEP_QPOSLAT Register (Offset = 18h) [reset = 0h]

EQEP_QPOSLAT is shown in Figure 11-461 and described in Table 11-951.

Table 11-950. EQEP_QPOSLAT Instances

Instance	Physical Address
EQEP_0	021C 0018h
EQEP_1	021C 0418h
EQEP_2	021C 0818h

Figure 11-461. EQEP_QPOSLAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSLAT																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-951. EQEP_QPOSLAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSLAT	R	0h	The position-counter value is latched into this register on unit time out event.

Table 11-952. Register Call Summary for EQEP_QPOSLAT

eQEP Registers • eQEP Registers: [0] • EQEP_QEPCTL Register (Offset = 2Ah) [reset = 0h]: [0] • EQEP_QPOSLAT Register (Offset = 18h) [reset = 0h]: [0]
eQEP Functional Description • Position Counter Reset on Unit Time out Event (EQEP_QEPCTL[13-12] PCRM = 0b11): [0] • Summary of eQEP Functional Registers: [0] • eQEP Edge Capture Unit: [0][1]

11.6.5.8 EQEP_QUTMR Register (Offset = 1Ch) [reset = 0h]

EQEP_QUTMR is shown in Figure 11-462 and described in Table 11-954.

Table 11-953. EQEP_QUTMR Instances

Instance	Physical Address
EQEP_0	021C 001Ch
EQEP_1	021C 041Ch
EQEP_2	021C 081Ch

Figure 11-462. EQEP_QUTMR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUTMR																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-954. EQEP_QUTMR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QUTMR	R/W	0h	This register acts as time base for unit time event generation. When this timer value matches with unit time period value, unit time event is generated.

Table 11-955. Register Call Summary for EQEP_QUTMR

eQEP Registers • EQEP_QUTMR Register (Offset = 1Ch) [reset = 0h]: [0] • eQEP Registers: [0]
eQEP Functional Description • Summary of eQEP Functional Registers: [0]

11.6.5.9 EQEP_QUPRD Register (Offset = 20h) [reset = 0h]

EQEP_QUPRD is shown in Figure 11-463 and described in Table 11-957.

Table 11-956. EQEP_QUPRD Instances

Instance	Physical Address
EQEP_0	021C 0020h
EQEP_1	021C 0420h
EQEP_2	021C 0820h

Figure 11-463. EQEP_QUPRD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUPRD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-957. EQEP_QUPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QUPRD	R/W	0h	This register contains the period count for unit timer to generate periodic unit time events to latch the eQEP position information at periodic interval and optionally to generate interrupt.

Table 11-958. Register Call Summary for EQEP_QUPRD

eQEP Registers • eQEP Registers: [0] • EQEP_QUPRD Register (Offset = 20h) [reset = 0h]: [0]
eQEP Functional Description • Summary of eQEP Functional Registers: [0] • Unit Timer Base: [0] • eQEP Edge Capture Unit: [0][1]

11.6.5.10 EQEP_QWDTMR Register (Offset = 24h) [reset = 0h]

EQEP_QWDTMR is shown in Figure 11-464 and described in Table 11-960.

Table 11-959. EQEP_QWDTMR Instances

Instance	Physical Address
EQEP_0	021C 0024h
EQEP_1	021C 0424h
EQEP_2	021C 0824h

Figure 11-464. EQEP_QWDTMR Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QWDTMR															
R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-960. EQEP_QWDTMR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QWDTMR	R/W	0h	This register acts as time base for watch dog to detect motor stalls. When this timer value matches with watch dog period value, watch dog timeout interrupt is generated. This register is reset upon edge transition in quadrature-clock indicating the motion.

Table 11-961. Register Call Summary for EQEP_QWDTMR

eQEP Registers • eQEP Registers: [0] • EQEP_QWDTMR Register (Offset = 24h) [reset = 0h]: [0]
eQEP Functional Description • Summary of eQEP Functional Registers: [0]

11.6.5.11 EQEP_QWDPRD Register (Offset = 26h) [reset = 0h]

EQEP_QWDPRD is shown in Figure 11-465 and described in Table 11-963.

Table 11-962. EQEP_QWDPRD Instances

Instance	Physical Address
EQEP_0	021C 0026h
EQEP_1	021C 0426h
EQEP_2	021C 0826h

Figure 11-465. EQEP_QWDPRD Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QWDPRD															
R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-963. EQEP_QWDPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QWDPRD	R/W	0h	This register contains the time-out count for the eQEP peripheral watch dog timer. When the watchdog timer value matches the watchdog period value, a watchdog timeout interrupt is generated.

Table 11-964. Register Call Summary for EQEP_QWDPRD

eQEP Registers • eQEP Registers: [0] • EQEP_QWDPRD Register (Offset = 26h) [reset = 0h]: [0]
eQEP Functional Description • eQEP Watchdog: [0] • Summary of eQEP Functional Registers: [0]

11.6.5.12 EQEP_QDECCTL Register (Offset = 28h) [reset = 0h]

EQEP_QDECCTL is shown in Figure 11-466 and described in Table 11-966.

Table 11-965. EQEP_QDECCTL Instances

Instance	Physical Address
EQEP_0	021C 0028h
EQEP_1	021C 0428h
EQEP_2	021C 0828h

Figure 11-466. EQEP_QDECCTL Register

15	14	13	12	11	10	9	8
QSRC		SOEN	SPSEL	XCR	SWAP	IGATE	QAP
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
QBP	QIP	QSP	RESERVED				
R/W-0h	R/W-0h	R/W-0h	R-0h				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-966. EQEP_QDECCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	QSRC	R/W	0h	Position-counter source selection. 0h = Quadrature count mode (QCLK = iCLK, QDIR = iDIR) 1h = Direction-count mode (QCLK = xCLK, QDIR = xDIR) 2h = UP count mode for frequency measurement (QCLK = xCLK, QDIR = 1) 3h = DOWN count mode for frequency measurement (QCLK = xCLK, QDIR = 0)
13	SOEN	R/W	0h	Sync output-enable 0h = Disable position-compare sync output 1h = Enable position-compare sync output
12	SPSEL	R/W	0h	Sync output pin selection 0h = Index pin is used for sync output 1h = Strobe pin is used for sync output
11	XCR	R/W	0h	External clock rate 0h = 2x resolution: Count the rising/falling edge 1h = 1x resolution: Count the rising edge only
10	SWAP	R/W	0h	Swap quadrature clock inputs. This swaps the input to the quadrature decoder, reversing the counting direction. 0h = Quadrature-clock inputs are not swapped 1h = Quadrature-clock inputs are swapped
9	IGATE	R/W	0h	Index pulse gating option 0h = Disable gating of Index pulse 1h = Gate the index pin with strobe
8	QAP	R/W	0h	QEPA input polarity 0h = No effect 1h = Negates QEPA input
7	QBP	R/W	0h	QEPB input polarity 0h = No effect 1h = Negates QEPB input

Table 11-966. EQEP_QDECCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	QIP	R/W	0h	QEPI input polarity 0h = No effect 1h = Negates QEPI input
5	QSP	R/W	0h	QEPS input polarity 0h = No effect 1h = Negates QEPS input
4-0	RESERVED	R	0h	Reserved

Table 11-967. Register Call Summary for EQEP_QDECCTL

eQEP Registers • eQEP Registers: [0] • EQEP_QDECCTL Register (Offset = 28h) [reset = 0h]: [0] • EQEP_QEPCTL Register (Offset = 2Ah) [reset = 0h]: [0]
eQEP Functional Description • eQEP Position Counter Input Modes: [0] • Quadrature Count Mode: [0] • eQEP Position-Compare Sync Output: [0][1] • eQEP Input Polarity Selection: [0][1] • Position Counter Reset on Unit Time out Event (EQEP_QEPCTL[13-12] PCRM = 0b11): [0] • eQEP Up-Count Mode: [0] • eQEP Down-Count Mode: [0] • Summary of eQEP Functional Registers: [0]

11.6.5.13 EQEP_QEPCTL Register (Offset = 2Ah) [reset = 0h]

EQEP_QEPCTL is shown in Figure 11-467 and described in Table 11-969.

Table 11-968. EQEP_QEPCTL Instances

Instance	Physical Address
EQEP_0	021C 002Ah
EQEP_1	021C 042Ah
EQEP_2	021C 082Ah

Figure 11-467. EQEP_QEPCTL Register

15	14	13	12	11	10	9	8
FREE_SOFT		PCRM		SEI		IEI	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
SWI	SEL	IEL		PHEN	QCLM	UTE	WDE
R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-969. EQEP_QEPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	FREE_SOFT	R/W	0h	Emulation Control Bits. In the values 0 through 3 listed below, x is different for the four following behaviors. EQEP_QPOSCNT behavior, x refers to the Position counter. QWDTMR behavior, x refers to the Watchdog counter. QUTMR behavior, x refers to the Unit timer. QCTMR behavior, x refers to the Capture timer. 0h = x stops immediately. For QPOSCNT behavior, the stop is on emulation suspend. 1h = x continues to count until the rollover. 2h = x is unaffected by emulation suspend. 3h = x is unaffected by emulation suspend.
13-12	PCRM	R/W	0h	Position counter reset mode 0h = Position counter reset on an index event 1h = Position counter reset on the maximum position 2h = Position counter reset on the first index event 3h = Position counter reset on a unit time event
11-10	SEI	R/W	0h	Strobe event initialization of position counter 0h = Does nothing (action disabled) 1h = Does nothing (action disabled) 2h = Initializes the position counter on rising edge of the QEPS signal 3h = Clockwise Direction: Initializes the position counter on the rising edge of QEPS strobe. Counter Clockwise Direction: Initializes the position counter on the falling edge of QEPS strobe
9-8	IEI	R/W	0h	Index event initialization of position counter 0h = Do nothing (action disabled) 1h = Do nothing (action disabled) 2h = Initializes the position counter on the rising edge of the QEPI signal (QPOSCNT = QPOSINIT) 3h = Initializes the position counter on the falling edge of QEPI signal (QPOSCNT = QPOSINIT)
7	SWI	R/W	0h	Software initialization of position counter 0h = Do nothing (action disabled) 1h = Initialize position counter, this bit is cleared automatically

Table 11-969. EQEP_QEPCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	SEL	R/W	0h	Strobe event latch of position counter 0h = The position counter is latched on the rising edge of QEPS strobe (QPOSSLAT = POSCCNT). Latching on the falling edge can be done by inverting the strobe input using the QSP bit in the EQEP_QDECCTL register. 1h = Clockwise Direction: Position counter is latched on rising edge of QEPS strobe. Counter Clockwise Direction: Position counter is latched on falling edge of QEPS strobe.
5-4	IEL	R/W	0h	Index event latch of position counter (software index marker) 0h = Reserved 1h = Latches position counter on rising edge of the index signal 2h = Latches position counter on falling edge of the index signal 3h = Software index marker. Latches the position counter and quadrature direction flag on index event marker. The position counter is latched to the EQEP_QPOSILAT register and the direction flag is latched in the EQEP_QEPSTS[QDLF] bit. This mode is useful for software index marking.
3	PHEN	R/W	0h	Quadrature position counter enable/software reset 0h = Reset the eQEP peripheral internal operating flags/read-only registers. Control/configuration registers are not disturbed by a software reset. 1h = eQEP position counter is enabled
2	QCLM	R/W	0h	eQEP capture latch mode 0h = Latch on position counter read by CPU. Capture timer and capture period values are latched into EQEP_QCTMRLAT and EQEP_QCPRDLAT registers when CPU reads the EQEP_QPOSCNT register. 1h = Latch on unit time out. Position counter, capture timer and capture period values are latched into EQEP_QPOSLAT, EQEP_QCTMRLAT and EQEP_QCPRDLAT registers on unit time out.
1	UTE	R/W	0h	eQEP unit timer enable 0h = Disable eQEP unit timer 1h = Enable unit timer
0	WDE	R/W	0h	eQEP watchdog enable 0h = Disable the eQEP watchdog timer 1h = Enable the eQEP watchdog timer

Table 11-970. Register Call Summary for EQEP_QEPCTL

eQEP Registers • eQEP Registers: [0] • EQEP_QPOSILAT Register (Offset = 10h) [reset = 0h]: [0] • EQEP_QEPCTL Register (Offset = 2Ah) [reset = 0h]: [0] • EQEP_QPOSSLAT Register (Offset = 14h) [reset = 0h]: [0]
eQEP Functional Description • eQEP Position Counter and Control Unit (PCCU): [0] • eQEP Position Counter Reset on Index Event (EQEP_QEPCTL[31-12] PCRM] = 0b00): [0] • Index Event Latch: [0][1][2][3][4][5][6] • eQEP Strobe Event Latch: [0][1] • Position Counter Reset on the First Index Event (EQEP_QEPCTL[13-12] PCRM = 0b10): [0] • eQEP Edge Capture Unit: [0][1] • eQEP Position Counter Reset on Maximum Position (EQEP_QEPCTL[13-12] PCRM=0b01): [0] • eQEP Position Counter Initialization: [0][1][2][3] • Summary of eQEP Functional Registers: [0]

11.6.5.14 EQEP_QCAPCTL Register (Offset = 2Ch) [reset = 0h]

EQEP_QCAPCTL is shown in Figure 11-468 and described in Table 11-972.

Table 11-971. EQEP_QCAPCTL Instances

Instance	Physical Address
EQEP_0	021C 002Ch
EQEP_1	021C 042Ch
EQEP_2	021C 082Ch

Figure 11-468. EQEP_QCAPCTL Register

15	14	13	12	11	10	9	8
CEN	RESERVED						
R/W-0h	R-0h						
7	6	5	4	3	2	1	0
RESERVED	CCPS			UPPS			
R-0h	R/W-0h			R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-972. EQEP_QCAPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	CEN	R/W	0h	Enable eQEP capture 0h = eQEP capture unit is disabled 1h = eQEP capture unit is enabled
14-7	RESERVED	R	0h	Reserved
6-4	CCPS	R/W	0h	eQEP capture timer clock prescaler 0h = CAPCLK = SYSCLKOUT/1 1h = CAPCLK = SYSCLKOUT/2 2h = CAPCLK = SYSCLKOUT/4 3h = CAPCLK = SYSCLKOUT/8 4h = CAPCLK = SYSCLKOUT/16 5h = CAPCLK = SYSCLKOUT/32 6h = CAPCLK = SYSCLKOUT/64 7h = CAPCLK = SYSCLKOUT/128
3-0	UPPS	R/W	0h	Unit position event prescaler 0h = UPEVNT = QCLK/1 1h = UPEVNT = QCLK/2 2h = UPEVNT = QCLK/4 3h = UPEVNT = QCLK/8 4h = UPEVNT = QCLK/16 5h = UPEVNT = QCLK/32 6h = UPEVNT = QCLK/64 7h = UPEVNT = QCLK/128 8h = UPEVNT = QCLK/256 9h = UPEVNT = QCLK/512 Ah = UPEVNT = QCLK/1024 Bh = UPEVNT = QCLK/2048 Ch = Reserved Dh = Reserved Eh = Reserved Fh = Reserved

Table 11-973. Register Call Summary for EQEP_QCAPCTL

eQEP Registers • eQEP Registers: [0] • EQEP_QCAPCTL Register (Offset = 2Ch) [reset = 0h]: [0]
eQEP Functional Description • Summary of eQEP Functional Registers: [0] • eQEP Edge Capture Unit: [0][1][2][3]

11.6.5.15 EQEP_QPOSCTL Register (Offset = 2Eh) [reset = 0h]

EQEP_QPOSCTL is shown in Figure 11-469 and described in Table 11-975.

Table 11-974. EQEP_QPOSCTL Instances

Instance	Physical Address
EQEP_0	021C 002Eh
EQEP_1	021C 042Eh
EQEP_2	021C 082Eh

Figure 11-469. EQEP_QPOSCTL Register

15	14	13	12	11	10	9	8
PCSHDW	PCLOAD	PCPOL	PCE	PCSPW			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h			
7	6	5	4	3	2	1	0
PCSPW							
R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-975. EQEP_QPOSCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	PCSHDW	R/W	0h	Position-compare shadow enable 0h = Shadow disabled, load Immediate 1h = Shadow enabled
14	PCLOAD	R/W	0h	Position-compare shadow load mode 0h = Load on QPOSCNT = 0 1h = Load when QPOSCNT = QPOSCMP
13	PCPOL	R/W	0h	Polarity of sync output 0h = Active HIGH pulse output 1h = Active LOW pulse output
12	PCE	R/W	0h	Position-compare enable/disable 0h = Disable position compare unit 1h = Enable position compare unit
11-0	PCSPW	R/W	0h	Select-position-compare sync output pulse width ... 0h = 1 × 4 × SYSCLKOUT cycles 1h = 2 × 4 × SYSCLKOUT cycles 2h = 3 × 4 × SYSCLKOUT cycles to 4096 × 4 × SYSCLKOUT cycles FFFh = 3 × 4 × SYSCLKOUT cycles to 4096 × 4 × SYSCLKOUT cycles

Table 11-976. Register Call Summary for EQEP_QPOSCTL

eQEP Registers • eQEP Registers: [0] • EQEP_QPOSCTL Register (Offset = 2Eh) [reset = 0h]: [0]
eQEP Functional Description • eQEP Position Counter and Control Unit (PCCU): [0] • Summary of eQEP Functional Registers: [0] • eQEP Position-Compare Unit: [0][1]

11.6.5.16 EQEP_QEINT Register (Offset = 30h) [reset = 0h]

EQEP_QEINT is shown in Figure 11-470 and described in Table 11-978.

Table 11-977. EQEP_QEINT Instances

Instance	Physical Address
EQEP_0	021C 0030h
EQEP_1	021C 0430h
EQEP_2	021C 0830h

Figure 11-470. EQEP_QEINT Register

15	14	13	12	11	10	9	8
RESERVED				UTO	IEL	SEL	PCM
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-978. EQEP_QEINT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	UTO	R/W	0h	Unit time out interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
10	IEL	R/W	0h	Index event latch interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
9	SEL	R/W	0h	Strobe event latch interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
8	PCM	R/W	0h	Position-compare match interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
7	PCR	R/W	0h	Position-compare ready interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
6	PCO	R/W	0h	Position counter overflow interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
5	PCU	R/W	0h	Position counter underflow interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
4	WTO	R/W	0h	Watchdog time out interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
3	QDC	R/W	0h	Quadrature direction change interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled

Table 11-978. EQEP_QEINT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	PHE	R/W	0h	Quadrature phase error interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
1	PCE	R/W	0h	Position counter error interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
0	RESERVED	R	0h	Reserved

Table 11-979. Register Call Summary for EQEP_QEINT

eQEP Registers • eQEP Registers: [0] • EQEP_QEINT Register (Offset = 30h) [reset = 0h]: [0]
eQEP Functional Description • Summary of eQEP Functional Registers: [0] • eQEP Interrupt Structure: [0]

11.6.5.17 EQEP_QFLG Register (Offset = 32h) [reset = 0h]

EQEP_QFLG is shown in Figure 11-471 and described in Table 11-981.

Table 11-980. EQEP_QFLG Instances

Instance	Physical Address
EQEP_0	021C 0032h
EQEP_1	021C 0432h
EQEP_2	021C 0832h

Figure 11-471. EQEP_QFLG Register

15	14	13	12	11	10	9	8
RESERVED				UTO	IEL	SEL	PCM
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	INT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-981. EQEP_QFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	UTO	R	0h	Unit time out interrupt flag 0h = No interrupt generated 1h = Set by eQEP unit timer period match
10	IEL	R	0h	Index event latch interrupt flag 0h = No interrupt generated 1h = This bit is set after latching the QPOSCNT to QPOSILAT
9	SEL	R	0h	Strobe event latch interrupt flag 0h = No interrupt generated 1h = This bit is set after latching the QPOSCNT to EQEP_QPOSSLAT
8	PCM	R	0h	eQEP compare match event interrupt flag 0h = No interrupt generated 1h = This bit is set on position-compare match
7	PCR	R	0h	Position-compare ready interrupt flag 0h = No interrupt generated 1h = This bit is set after transferring the shadow register value to the active position compare register.
6	PCO	R	0h	Position counter overflow interrupt flag 0h = No interrupt generated 1h = This bit is set on position counter overflow.
5	PCU	R	0h	Position counter underflow interrupt flag 0h = No interrupt generated 1h = This bit is set on position counter underflow.
4	WTO	R	0h	Watchdog timeout interrupt flag 0h = No interrupt generated 1h = Set by watch dog timeout
3	QDC	R	0h	Quadrature direction change interrupt flag 0h = No interrupt generated 1h = This bit is set during change of direction

Table 11-981. EQEP_QFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	PHE	R	0h	Quadrature phase error interrupt flag 0h = No interrupt generated 1h = Set on simultaneous transition of QEPA and QEPB
1	PCE	R	0h	Position counter error interrupt flag 0h = No interrupt generated 1h = Position counter error
0	INT	R	0h	Global interrupt status flag 0h = No interrupt generated 1h = Interrupt was generated

Table 11-982. Register Call Summary for EQEP_QFLG

eQEP Registers • EQEP_QFLG Register (Offset = 32h) [reset = 0h]: [0] • eQEP Registers: [0]
eQEP Functional Description • Summary of eQEP Functional Registers: [0] • Index Event Latch: [0] • eQEP Strobe Event Latch: [0] • eQEP Position Counter Initialization: [0][1] • eQEP Position-Compare Unit: [0][1] • eQEP Interrupt Structure: [0] • eQEP Watchdog: [0] • Quadrature Count Mode: [0] • eQEP Position Counter Operating Modes: [0] • eQEP Position Counter Reset on Index Event (EQEP_QEPCTL[31-12] PCRM] = 0b00): [0][1] • Unit Timer Base: [0]

11.6.5.18 EQEP_QCLR Register (Offset = 34h) [reset = 0h]

EQEP_QCLR is shown in Figure 11-472 and described in Table 11-984.

Table 11-983. EQEP_QCLR Instances

Instance	Physical Address
EQEP_0	021C 0034h
EQEP_1	021C 0434h
EQEP_2	021C 0834h

Figure 11-472. EQEP_QCLR Register

15	14	13	12	11	10	9	8
RESERVED				UTO	IEL	SEL	PCM
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	INT
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-984. EQEP_QCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	UTO	R/W	0h	Clear unit time out interrupt flag 0h = No effect 1h = Clears the interrupt flag
10	IEL	R/W	0h	Clear index event latch interrupt flag 0h = No effect 1h = Clears the interrupt flag
9	SEL	R/W	0h	Clear strobe event latch interrupt flag 0h = No effect 1h = Clears the interrupt flag
8	PCM	R/W	0h	Clear eQEP compare match event interrupt flag 0h = No effect 1h = Clears the interrupt flag
7	PCR	R/W	0h	Clear position-compare ready interrupt flag 0h = No effect 1h = Clears the interrupt flag
6	PCO	R/W	0h	Clear position counter overflow interrupt flag 0h = No effect 1h = Clears the interrupt flag
5	PCU	R/W	0h	Clear position counter underflow interrupt flag 0h = No effect 1h = Clears the interrupt flag
4	WTO	R/W	0h	Clear watchdog timeout interrupt flag 0h = No effect 1h = Clears the interrupt flag
3	QDC	R/W	0h	Clear quadrature direction change interrupt flag 0h = No effect 1h = Clears the interrupt flag

Table 11-984. EQEP_QCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	PHE	R/W	0h	Clear quadrature phase error interrupt flag 0h = No effect 1h = Clears the interrupt flag
1	PCE	R/W	0h	Clear position counter error interrupt flag 0h = No effect 1h = Clears the interrupt flag
0	INT	R/W	0h	Global interrupt clear flag 0h = No effect 1h = Clears the interrupt flag and enables further interrupts to be generated if an event flags is set to 1.

Table 11-985. Register Call Summary for EQEP_QCLR

eQEP Registers • EQEP_QCLR Register (Offset = 34h) [reset = 0h]: [0] • eQEP Registers: [0]
eQEP Functional Description • Summary of eQEP Functional Registers: [0] • eQEP Interrupt Structure: [0]

11.6.5.19 EQEP_QFRC Register (Offset = 36h) [reset = 0h]

EQEP_QFRC is shown in Figure 11-473 and described in Table 11-987.

Table 11-986. EQEP_QFRC Instances

Instance	Physical Address
EQEP_0	021C 0036h
EQEP_1	021C 0436h
EQEP_2	021C 0836h

Figure 11-473. EQEP_QFRC Register

15	14	13	12	11	10	9	8
RESERVED				UTO	IEL	SEL	PCM
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-987. EQEP_QFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	UTO	R/W	0h	Force unit time out interrupt 0h = No effect 1h = Force the interrupt
10	IEL	R/W	0h	Force index event latch interrupt 0h = No effect 1h = Force the interrupt
9	SEL	R/W	0h	Force strobe event latch interrupt 0h = No effect 1h = Force the interrupt
8	PCM	R/W	0h	Force position-compare match interrupt 0h = No effect 1h = Force the interrupt
7	PCR	R/W	0h	Force position-compare ready interrupt 0h = No effect 1h = Force the interrupt
6	PCO	R/W	0h	Force position counter overflow interrupt 0h = No effect 1h = Force the interrupt
5	PCU	R/W	0h	Force position counter underflow interrupt 0h = No effect 1h = Force the interrupt
4	WTO	R/W	0h	Force watchdog time out interrupt 0h = No effect 1h = Force the interrupt
3	QDC	R/W	0h	Force quadrature direction change interrupt 0h = No effect 1h = Force the interrupt

Table 11-987. EQEP_QFRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	PHE	R/W	0h	Force quadrature phase error interrupt 0h = No effect 1h = Force the interrupt
1	PCE	R/W	0h	Force position counter error interrupt 0h = No effect 1h = Force the interrupt
0	RESERVED	R	0h	Reserved

Table 11-988. Register Call Summary for EQEP_QFRC

eQEP Registers • eQEP Registers: [0] • EQEP_QFRC Register (Offset = 36h) [reset = 0h]: [0]
eQEP Functional Description • Summary of eQEP Functional Registers: [0] • eQEP Interrupt Structure: [0]

11.6.5.20 EQEP_QEPSTS Register (Offset = 38h) [reset = 0h]

EQEP_QEPSTS is shown in Figure 11-474 and described in Table 11-990.

Table 11-989. EQEP_QEPSTS Instances

Instance	Physical Address
EQEP_0	021C 0038h
EQEP_1	021C 0438h
EQEP_2	021C 0838h

Figure 11-474. EQEP_QEPSTS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
UPEVNT	FDF	QDF	QDLF	COEF	CDEF	FIMF	PCEF
R-0h	R-0h	R-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-990. EQEP_QEPSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	UPEVNT	R	0h	Unit position event flag 0h = No unit position event detected 1h = Unit position event detected. Write 1 to clear.
6	FDF	R	0h	Direction on the first index marker. Status of the direction is latched on the first index event marker. 0h = Counter-clockwise rotation (or reverse movement) on the first index event 1h = Clockwise rotation (or forward movement) on the first index event
5	QDF	R	0h	Quadrature direction flag 0h = Counter-clockwise rotation (or reverse movement) 1h = Clockwise rotation (or forward movement)
4	QDLF	R	0h	eQEP direction latch flag. Status of direction is latched on every index event marker. 0h = Counter-clockwise rotation (or reverse movement) on index event marker 1h = Clockwise rotation (or forward movement) on index event marker
3	COEF	R/W	0h	Capture overflow error flag 0h = Sticky bit, cleared by writing 1 1h = Overflow occurred in eQEP Capture timer (QEPCTMR)
2	CDEF	R/W	0h	Capture direction error flag 0h = Sticky bit, cleared by writing 1 1h = Direction change occurred between the capture position event.
1	FIMF	R/W	0h	First index marker flag 0h = Sticky bit, cleared by writing 1 1h = Set by first occurrence of index pulse
0	PCEF	R	0h	Position counter error flag. This bit is not sticky and it is updated for every index event. 0h = No error occurred during the last index transition. 1h = Position counter error

Table 11-991. Register Call Summary for EQEP_QEPSTS

eQEP Registers • eQEP Registers: [0] • EQEP_QEPSTS Register (Offset = 38h) [reset = 0h]: [0] • EQEP_QEPCTL Register (Offset = 2Ah) [reset = 0h]: [0]
eQEP Functional Description • Quadrature Count Mode: [0] • eQEP Edge Capture Unit: [0][1][2] • eQEP Position Counter Reset on Index Event (EQEP_QEPCTL[31-12] PCRM] = 0b00): [0][1][2][3][4][5] • Index Event Latch: [0][1][2][3] • Position Counter Reset on the First Index Event (EQEP_QEPCTL[13-12] PCRM = 0b10): [0][1][2] • eQEP Position Counter Reset on Maximum Position (EQEP_QEPCTL[13-12] PCRM=0b01): [0][1][2] • Summary of eQEP Functional Registers: [0]

11.6.5.21 EQEP_QCTMR Register (Offset = 3Ah) [reset = 0h]

EQEP_QCTMR is shown in Figure 11-475 and described in Table 11-993.

Table 11-992. EQEP_QCTMR Instances

Instance	Physical Address
EQEP_0	021C 003Ah
EQEP_1	021C 043Ah
EQEP_2	021C 083Ah

Figure 11-475. EQEP_QCTMR Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QCTMR															
R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-993. EQEP_QCTMR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QCTMR	R/W	0h	This register provides time base for edge capture unit.

Table 11-994. Register Call Summary for EQEP_QCTMR

eQEP Registers • eQEP Registers: [0] • EQEP_QCTMR Register (Offset = 3Ah) [reset = 0h]: [0]
eQEP Functional Description • Summary of eQEP Functional Registers: [0] • eQEP Edge Capture Unit: [0][1]

11.6.5.22 EQEP_QCPRD Register (Offset = 3Ch) [reset = 0h]

EQEP_QCPRD is shown in Figure 11-476 and described in Table 11-996.

Table 11-995. EQEP_QCPRD Instances

Instance	Physical Address
EQEP_0	021C 003Ch
EQEP_1	021C 043Ch
EQEP_2	021C 083Ch

Figure 11-476. EQEP_QCPRD Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QCPRD															
R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-996. EQEP_QCPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QCPRD	R/W	0h	This register holds the period count value between the last successive eQEP position events.

Table 11-997. Register Call Summary for EQEP_QCPRD

eQEP Registers • eQEP Registers: [0] • EQEP_QCPRD Register (Offset = 3Ch) [reset = 0h]: [0]
eQEP Functional Description • Summary of eQEP Functional Registers: [0] • eQEP Edge Capture Unit: [0][1][2]

11.6.5.23 EQEP_QCTMRLAT Register (Offset = 3Eh) [reset = 0h]

EQEP_QCTMRLAT is shown in Figure 11-477 and described in Table 11-999.

Table 11-998. EQEP_QCTMRLAT Instances

Instance	Physical Address
EQEP_0	021C 003Eh
EQEP_1	021C 043Eh
EQEP_2	021C 083Eh

Figure 11-477. EQEP_QCTMRLAT Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QCTMRLAT															
R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-999. EQEP_QCTMRLAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QCTMRLAT	R	0h	The eQEP capture timer value can be latched into this register on two events, that is, unit timeout event, reading the eQEP position counter.

Table 11-1000. Register Call Summary for EQEP_QCTMRLAT

eQEP Registers • eQEP Registers: [0] • EQEP_QCTMRLAT Register (Offset = 3Eh) [reset = 0h]: [0] • EQEP_QEPCTL Register (Offset = 2Ah) [reset = 0h]: [0][1]
eQEP Functional Description • Summary of eQEP Functional Registers: [0] • eQEP Edge Capture Unit: [0][1]

11.6.5.24 EQEP_QCPRDLAT Register (Offset = 40h) [reset = 0h]

EQEP_QCPRDLAT is shown in Figure 11-478 and described in Table 11-1002.

Table 11-1001. EQEP_QCPRDLAT Instances

Instance	Physical Address
EQEP_0	021C 0040h
EQEP_1	021C 0440h
EQEP_2	021C 0840h

Figure 11-478. EQEP_QCPRDLAT Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QCPRDLAT															
R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1002. EQEP_QCPRDLAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QCPRDLAT	R/W	0h	eQEP capture period value can be latched into this register on two events, that is, unit timeout event, reading the eQEP position counter.

Table 11-1003. Register Call Summary for EQEP_QCPRDLAT

eQEP Registers • eQEP Registers: [0] • EQEP_QCPRDLAT Register (Offset = 40h) [reset = 0h]: [0] • EQEP_QEPCTL Register (Offset = 2Ah) [reset = 0h]: [0][1]
eQEP Functional Description • Summary of eQEP Functional Registers: [0] • eQEP Edge Capture Unit: [0][1][2]

11.6.5.25 EQEP_REVID Register (Offset = 5Ch) [reset = -h]

EQEP_REVID is shown in Figure 11-479 and described in Table 11-1005.

Table 11-1004. EQEP_REVID Instances

Instance	Physical Address
EQEP_0	021C 005Ch
EQEP_1	021C 045Ch
EQEP_2	021C 085Ch

Figure 11-479. EQEP_REVID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REVISION																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1005. EQEP_REVID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REVISION	R	-h	IP Revision

Table 11-1006. Register Call Summary for EQEP_REVID

eQEP Registers • eQEP Registers: [0] • EQEP_REVID Register (Offset = 5Ch) [reset = -h]: [0]
eQEP Functional Description • Summary of eQEP Functional Registers: [0]

11.7 General-Purpose Interface (GPIO)

This chapter describes the General-Purpose Interface (GPIO) for the device.

11.7.1 GPIO Overview

The general-purpose input/output (GPIO) peripheral provides dedicated general-purpose pins that can be configured as either inputs or outputs. When configured as an output, user can write to an internal register to control the state driven on the output pin. When configured as an input, user can obtain the state of the input by reading the state of an internal register.

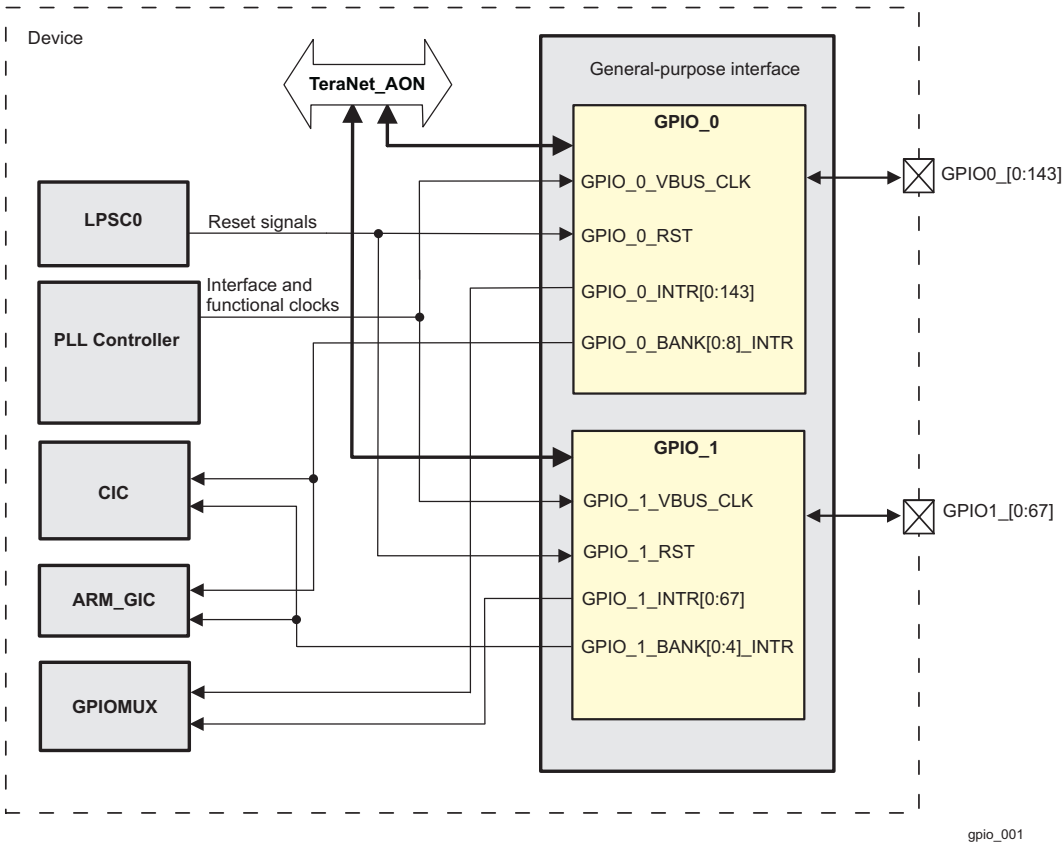
In addition, the GPIO peripheral can produce CPU interrupts and EDMA synchronization events in different interrupt/event generation modes.

The device has two instances of GPIO144 modules (GPIO_0 and GPIO_1). The GPIO pins are grouped into banks (16 pins per bank), which means that each GPIO module provides up to 144 dedicated general-purpose pins with input and output capabilities; thus, the general-purpose interface supports up to 288 (2 instances × (9 banks x 16 pins)) pins. Since GPIO1_[143:68] are reserved in this Device, general-purpose interface supports up to 212 pins.

All GPIO pins are muxed with other device pins. For details on specific muxing and for the availability of the register bits, see the device Data Manual. GPINTj (where j = bit index, j = [0:31]) are all available as synchronization events to the EDMA and as interrupt sources to the CPU.

Figure 11-480 is an overview of the general-purpose interface.

Figure 11-480. GPIO Overview



Each channel in the GPIO modules has the following features:

- Supports 9 banks of 16 GPIO signals
- Supports up to 9 banks of interrupt capable GPIOs
- Interrupts:

  - Can enable interrupts for each bank of 16 GPIO signals
  - Interrupts can be triggered by rising and/or falling edge (or neither edge = disabled), specified for each interrupt capable GPIO signal
- Set/clear functionality:
  - Software writes 1 to corresponding bit position(s) to set or to clear GPIO signal(s). This allows multiple software processes to toggle GPIO output signals without critical section protection (disable interrupts, program GPIO, re-enable interrupts, to prevent context switching to another process during GPIO programming).
- Separate Input/Output registers:
  - Output register in addition to set/clear so that if preferred by software, some GPIO output signals can be toggled by direct write to the output register(s).
  - Output register, when read in, reflects output drive status. This, in addition to the input register reflecting pin status and open-drain I/O cell, allows wired logic be implemented.

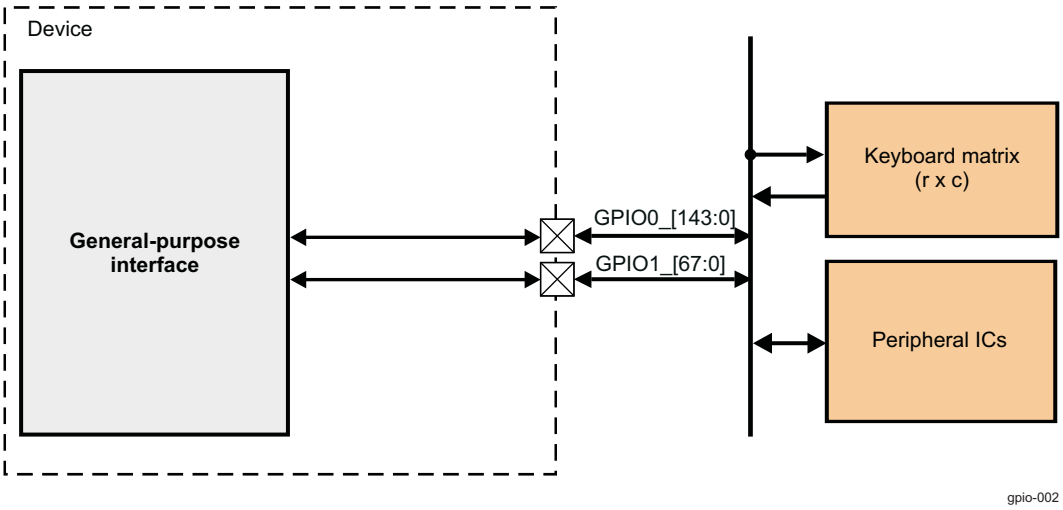
11.7.2 GPIO Environment

The general-purpose interface combines two GPIO modules for a flexible, user-programmable, general-purpose input/output (I/O) controller. The general-purpose interface implements functions that are not implemented with the dedicated controllers in the device and require simple input and/or output software-controlled signals. The GPIO allows a variety of custom connections and expands the I/O capabilities of the system to the real world.

The general-purpose interface can physically connect the device to a keyboard matrix and peripheral integrated circuits (ICs).

Figure 11-481 shows a typical application using the general-purpose interface.

Figure 11-481. GPIO Typical Application



11.7.2.1 General-Purpose Interface Signals

Table 11-1007 describes the module signals.

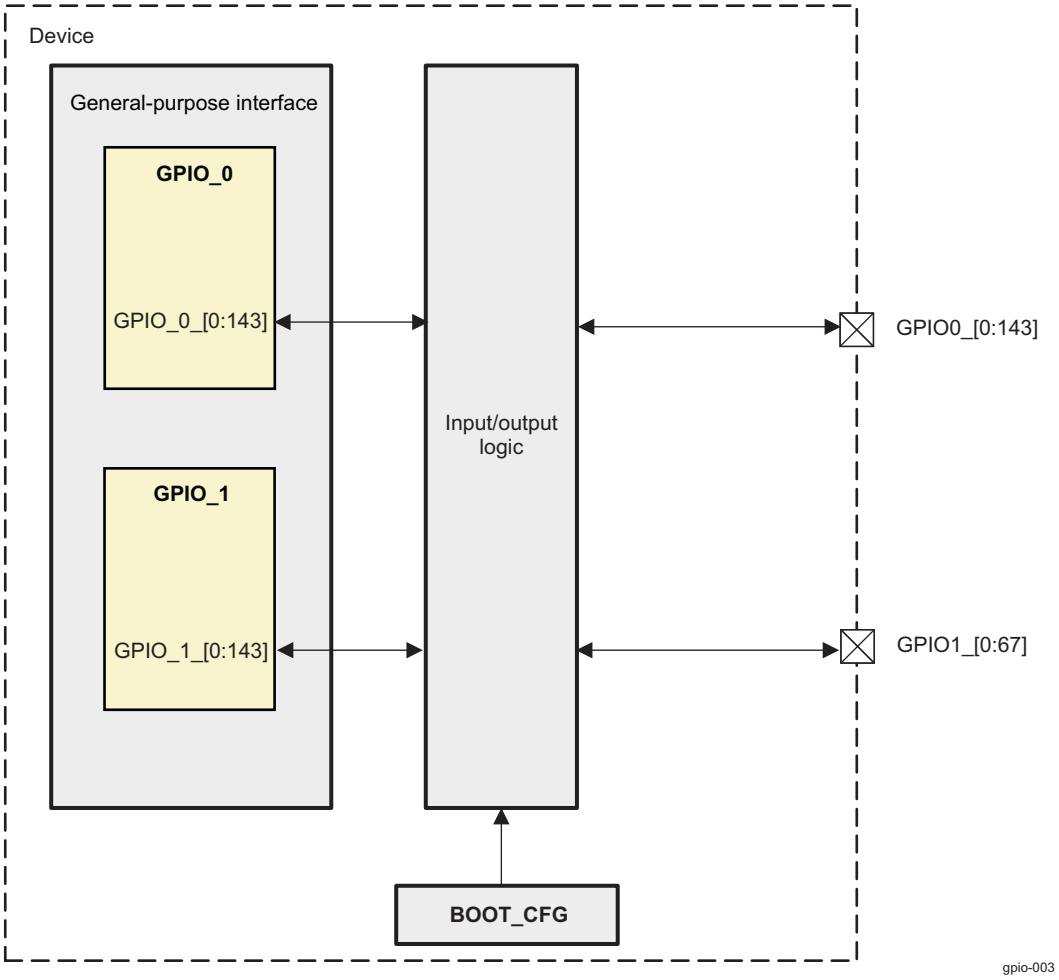
Table 11-1007. GPIO Description

Signal	I/O[1]	Description	Reset Value[2]
GPIO0_[0:143]	I/O	GPIO	HiZ
GPIO1_[0:67]	I/O	GPIO	HiZ

(1) I = Input; O = Output; I/O = Bidirectional
(2) HiZ = High Impedance

Figure 11-482 shows the signal connections of GPIO_0 and GPIO_1.

Figure 11-482. GPIO_0 and GPIO_1 Signal Connections



NOTE: For more information about GPIO signal multiplexing, see Section 5.1.3.1.1, *Pad Configuration Registers* in Section 5.1, *BOOT_CFG*.

11.7.3 GPIO Integration

This section describes module integration in the device, including information about clocks, resets, and hardware requests.

Figure 11-483 shows this module integration.

Figure 11-483. GPIO Integration

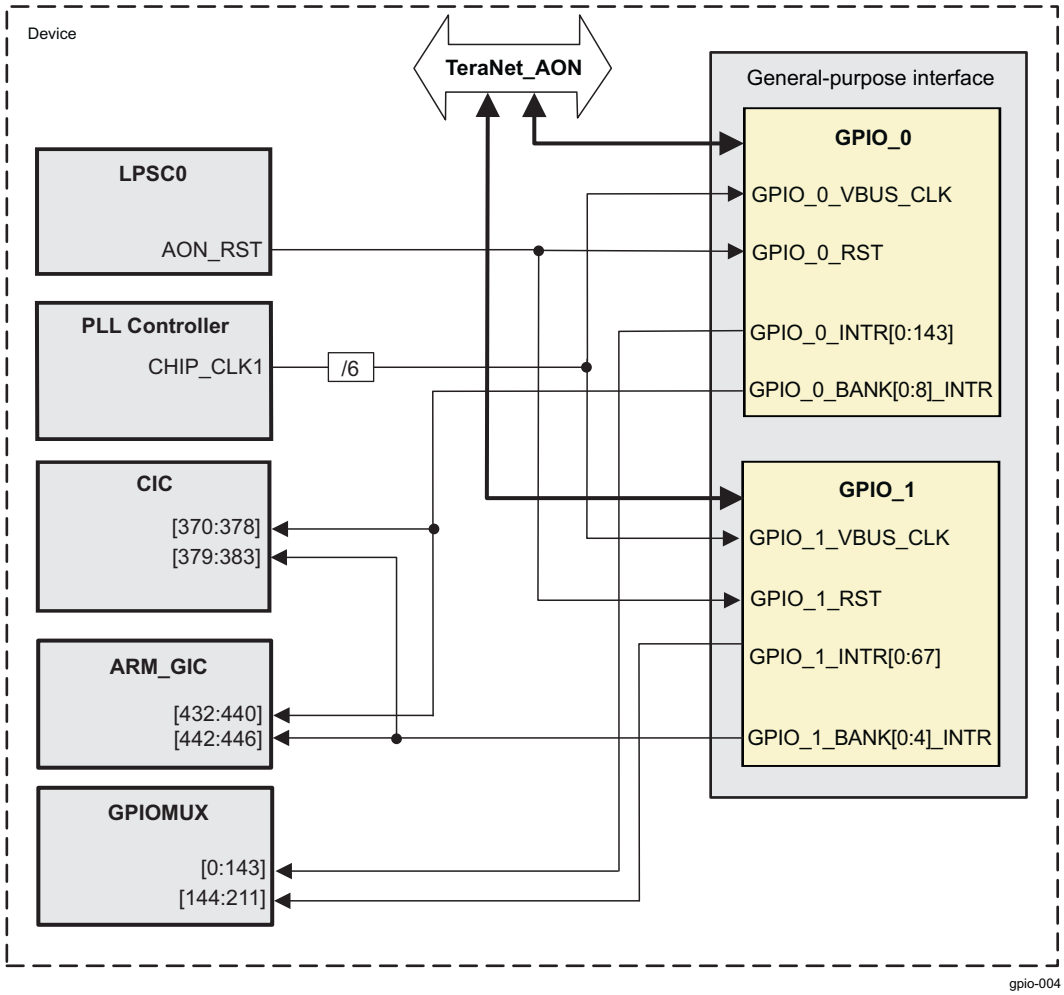


Table 11-1008 through Table 11-1010 summarize the integration of the module in the device.

Table 11-1008. GPIO Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
GPIO_0	PD0	LPSC0	TeraNet_AON
GPIO_1	PD0	LPSC0	TeraNet_AON

Table 11-1009. GPIO Clocks and Resets

Clocks				
Module Instance	Destination Signal	Source Signal	Source	Description
GPIO_0	GPIO_0_VBUS_CLK	CHIP_CLK1/6	PLL Controller	GPIO_0 Interface and Functional clock
GPIO_1	GPIO_1_VBUS_CLK	CHIP_CLK1/6	PLL Controller	GPIO_1 Interface and Functional clock

Table 11-1009. GPIO Clocks and Resets (continued)

Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
GPIO_0	GPIO_0_RST	AON_RST	LPSC0	GPIO_0 reset signal
GPIO_1	GPIO_1_RST	AON_RST	LPSC0	GPIO_1 reset signal

Table 11-1010. GPIO Hardware Requests

Interrupt Requests					
Module Instance	**Event Name**	**Mapped To Input Event [Number]**			**Description**
		ARM GIC	**CIC**	**GPIOMUX**[1]	
GPIO_0	GPIO_0_BANK0_INTR	[432]	[370]	-	GPIO_0 bank0 interrupt request
	GPIO_0_BANK1_INTR	[433]	[371]	-	GPIO_0 bank1 interrupt request
	GPIO_0_BANK2_INTR	[434]	[372]	-	GPIO_0 bank2 interrupt request
	GPIO_0_BANK3_INTR	[435]	[373]	-	GPIO_0 bank3 interrupt request
	GPIO_0_BANK4_INTR	[436]	[374]	-	GPIO_0 bank4 interrupt request
	GPIO_0_BANK5_INTR	[437]	[375]	-	GPIO_0 bank5 interrupt request
	GPIO_0_BANK6_INTR	[438]	[376]	-	GPIO_0 bank6 interrupt request
	GPIO_0_BANK7_INTR	[439]	[377]	-	GPIO_0 bank7 interrupt request
	GPIO_0_BANK8_INTR	[440]	[378]	-	GPIO_0 bank8 interrupt request
	GPIO_0_INTR[0:143]	-	-	[0:143]	GPIO_0 pins[0:143] interrupt request
GPIO_1	GPIO_1_BANK0_INTR	[442]	[379]	-	GPIO_1 bank0 interrupt request
	GPIO_1_BANK1_INTR	[443]	[380]	-	GPIO_1 bank1 interrupt request
	GPIO_1_BANK2_INTR	[444]	[381]	-	GPIO_1 bank2 interrupt request
	GPIO_1_BANK3_INTR	[445]	[382]	-	GPIO_1 bank3 interrupt request
	GPIO_1_BANK4_INTR	[446]	[383]	-	GPIO_1 bank4 interrupt request
	GPIO_1_INTR[0:67]	-	-	[144:211]	GPIO_1 pins[0:67] interrupt request

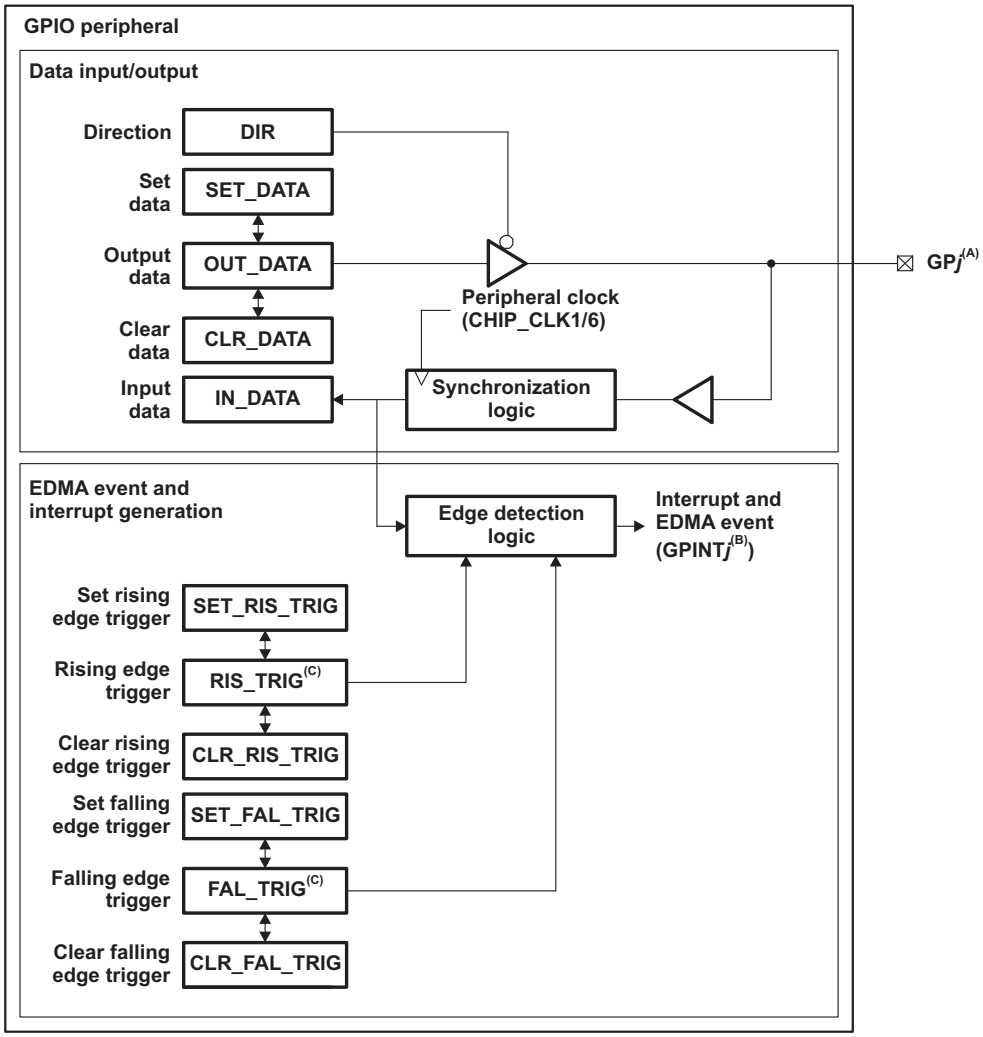
(1) For GPIOMUX output event mapping to the device interrupt/DMA controllers, see the corresponding interrupt/DMA mapping tables.
For GPIO input event mapping, see Table 11-1012.
For GPIOMUX description, see Section 5.1.3.1.7, *Event Mux Control Registers*.

11.7.4 GPIO Functional Description

11.7.4.1 GPIO Block Diagram

Figure 11-484 shows the general-purpose interface block diagram.

Figure 11-484. GPIO Block Diagram



A Some of the GPj pins are muxed with other device signals. For details, see the device Data Manual.

B All GPINTj can be used as CPU interrupts and synchronization events to the EDMA.

C The RIS_TRIG and FAL_TRIG registers are internal to the GPIO module and are not visible to the CPU.

11.7.4.2 GPIO Function

Each GPIO pin (GPj) can be independently configured as either an input or an output using the GPIO direction registers. The GPIO direction register (DIR) specifies the direction of each GPIO signal. Logic 0 indicates the GPIO pin is configured as output, and logic 1 indicates input.

When configured as output, writing a 1 to a bit in the set data register drives the corresponding GPj to a logic-high state. Writing a 1 to a bit in the clear data register drives the corresponding GPj to a logic-low state. The output state of each GPj can also be directly controlled by writing to the output data register.

For example, to set GP8 to a logic-high state, the software can perform one of the following:

- Write 100h to the GPIO_SET_DATA01 register

- Read in GPIO_OUT_DATA01 register, change the eighth bit to 1, and write the new value back to GPIO_OUT_DATA01

To set GP8 to a logic-low state, the software can perform one of the following:

- Write 100h to the GPIO_CLR_DATA01 register
- Read in GPIO_OUT_DATA01 register, change the eighth bit to 0, and write the new value back to GPIO_OUT_DATA01

Note that writing a 0 to bits in the set data and clear data registers does not affect the GPIO pin state.

Also, for GPIO pins configured as input, writing to the set data, clear data, or output data registers does not affect the pin state.

For a GPIO pin configured as input, reading the input data register (IN_DATA) will return the pin state. Reading the SET_DATA register or the CLR_DATA data register will return the value in OUT_DATA, not the actual pin state. The pin state is available by reading the input data register. Note that when the direction is configured as input, the output state is determined by software's programming set/clear/output registers, and may not agree with the pin state, which is driven by an external device.

11.7.4.3 Interrupt and Event Generation

Each GPIO pin (GPj) can be configured to generate a CPU interrupt (GPINTj) and a synchronization event to the EDMA (GPINTj). Configuration is on per-bank basis. Each bit of the BANK_INTR_EN parameter dictates YES/NO option for each bank. Bit 0 controls bank 0, bit 1 controls bank 1, and so on.

The interrupt and EDMA event can be generated on the rising-edge, falling-edge, or on both edges of the GPIO signal. The edge detection logic is synchronized to the GPIO peripheral clock.

The direction of the GPIO pin does not need to be input when using the pin to generate the interrupt and EDMA event. When the GPIO pin is configured as input, transitions on the pin trigger interrupts and EDMA events. When the GPIO pin is configured as output, software can toggle the GPIO output register to change the pin state and in turn trigger the interrupt and EDMA event.

Note that the direction of the pin need not be input for interrupt generation to work. When the GPINT pin is configured as input, transitions on the pin trigger interrupts. When the GPINT pin is configured as output, firmware can toggle the GPIO output register to change the pin state, and in turn trigger interrupts.

Each interrupt output of GPINT signal are available at module boundary. Each group of 16 GPINT signals also has their masked interrupt outputs ORed together to generate per bank interrupt, available at module boundary. The idea is to either connect individual interrupts or per bank interrupts to the system interrupt controller.

11.7.4.3.1 Interrupt Enable (per Bank)

GPIO_BINTEN register provides interrupt enable/disable feature for each bank of 16 GPINT signals.

11.7.4.3.2 Trigger Configuration (per Bit)

Two internal registers, RIS_TRIG and FAL_TRIG, specify which edge of the GPj signal generates an interrupt and EDMA event. Each bit in these two registers corresponds to a GPj pin. Table 11-1011 describes the CPU interrupt and EDMA event generation of GPj pin based on the bit settings of the RIS_TRIG and FAL_TRIG registers.

Table 11-1011. GPIO Interrupt and EDMA Event Configuration Options

RIS_TRIG Bit n	FAL_TRIG Bit n	CPU Interrupt and EDMA Event Generation
0	0	GPINTj interrupt and EDMA event is disabled
0	1	GPINTj interrupt and EDMA event is triggered on falling edge of GPj signal
1	0	GPINTj interrupt and EDMA event is triggered on rising edge of GPj signal
1	1	GPINTj interrupt and EDMA event is triggered on both rising and falling edge of GPj signal

RIS_TRIG and FAL_TRIG are not directly accessible or visible to the CPU. These registers are accessed indirectly through four registers: SET_RIS_TRIG, CLR_RIS_TRIG, SET_FAL_TRIG, and CLR_FAL_TRIG. Writing 1 to a bit on the SET_RIS_TRIG register sets the corresponding bit on the RIS_TRIG register. Writing 1 to a bit of CLR_RIS_TRIG register clears the corresponding bit on the RIS_TRIG register. Writing to SET_FAL_TRIG and CLR_FAL_TRIG works the same way on the FAL_TRIG register.

Reading the SET_RIS_TRIG or CLR_RIS_TRIG register returns the value of RIS_TRIG register. Reading from SET_FAL_TRIG and CLR_FAL_TRIG register returns the value of FAL_TRIG register.

To use the GPIO pins as sources for CPU interrupts and EDMA events, bit 0 in the bank interrupt enable register (GPIO_BINTEN) must be set to 1.

11.7.4.3.3 Interrupt Status and Clear (per Bit)

INTSTAT registers provide interrupt status upon reading, and interrupt clear feature upon writing 1 to the corresponding bit position(s). Upon receiving an interrupt, the ISR can examine the interrupt status and clear the processed interrupts.

NOTE: GPIO module generates an interrupt pulse on the individual GPINT interrupt in response to each occurrence of the specified edge condition. Therefore, for GPINT signals having their interrupts routed directly to the interrupt controller, it is not necessary to clear the status bits in this module. The interrupt status and clear register is a facility for the per-bank interrupt connection.

11.7.4.4 GPIO Interrupt Connectivity

Because this device muxes all GPIO signals with other functional signals, the availability of any particular GPIO and hence the usability of its associated interrupt will change based on the use case. The large number of possible GPIO interrupt sources makes it impractical to route all interrupt events to each processor. Since most applications don't typically require a large number of GPIO interrupts, the interrupt uncertainty is resolved by mapping all GPIO interrupts to a series of event muxes (GPIOMUX). These muxes allow any one of the available GPIO interrupts (per pin) to be selected and passed on as an event to the various processor interrupt controllers and DMA controllers. Event selection is controlled through associated BOOT_CFG module registers EVENT_MUXCTL0/11. Table 11-1012 summarizes the GPIOMUX input event mapping.

There is a use case to use some of the GPIO pins as interrupt sources for the DSP core (connected either directly or through CPINTC) and Arm core (connected via GIC). Also, some of the GPIO pins can be used as wake-up sources for the PMMC, so they are directly connected to the PMMC interrupt inputs.

One of the GPIO pins has a potential use case as an Arm reset input and should therefore be routed as highest priority interrupt in the GIC and mapped to nFIQ in software.

Table 11-1012. GPIOMUX Input Event Mapping

Input Event	Event Name	Description
0	GPIO_0_INTR0	GPIO0 pin 0 interrupt
1	GPIO_0_INTR1	GPIO0 pin 1 interrupt
2	GPIO_0_INTR2	GPIO0 pin 2 interrupt
3	GPIO_0_INTR3	GPIO0 pin 3 interrupt
4	GPIO_0_INTR4	GPIO0 pin 4 interrupt
5	GPIO_0_INTR5	GPIO0 pin 5 interrupt
6	GPIO_0_INTR6	GPIO0 pin 6 interrupt
7	GPIO_0_INTR7	GPIO0 pin 7 interrupt
8	GPIO_0_INTR8	GPIO0 pin 8 interrupt
9	GPIO_0_INTR9	GPIO0 pin 9 interrupt
10	GPIO_0_INTR10	GPIO0 pin 10 interrupt
11	GPIO_0_INTR11	GPIO0 pin 11 interrupt

Table 11-1012. GPIOMUX Input Event Mapping (continued)

Input Event	Event Name	Description
12	GPIO_0_INTR12	GPIO0 pin 12 interrupt
13	GPIO_0_INTR13	GPIO0 pin 13 interrupt
14	GPIO_0_INTR14	GPIO0 pin 14 interrupt
15	GPIO_0_INTR15	GPIO0 pin 15 interrupt
16	GPIO_0_INTR16	GPIO0 pin 16 interrupt
17	GPIO_0_INTR17	GPIO0 pin 17 interrupt
18	GPIO_0_INTR18	GPIO0 pin 18 interrupt
19	GPIO_0_INTR19	GPIO0 pin 19 interrupt
20	GPIO_0_INTR20	GPIO0 pin 20 interrupt
21	GPIO_0_INTR21	GPIO0 pin 21 interrupt
22	GPIO_0_INTR22	GPIO0 pin 22 interrupt
23	GPIO_0_INTR23	GPIO0 pin 23 interrupt
24	GPIO_0_INTR24	GPIO0 pin 24 interrupt
25	GPIO_0_INTR25	GPIO0 pin 25 interrupt
26	GPIO_0_INTR26	GPIO0 pin 26 interrupt
27	GPIO_0_INTR27	GPIO0 pin 27 interrupt
28	GPIO_0_INTR28	GPIO0 pin 28 interrupt
29	GPIO_0_INTR29	GPIO0 pin 29 interrupt
30	GPIO_0_INTR30	GPIO0 pin 30 interrupt
31	GPIO_0_INTR31	GPIO0 pin 31 interrupt
32	GPIO_0_INTR32	GPIO0 pin 32 interrupt
33	GPIO_0_INTR33	GPIO0 pin 33 interrupt
34	GPIO_0_INTR34	GPIO0 pin 34 interrupt
35	GPIO_0_INTR35	GPIO0 pin 35 interrupt
36	GPIO_0_INTR36	GPIO0 pin 36 interrupt
37	GPIO_0_INTR37	GPIO0 pin 37 interrupt
38	GPIO_0_INTR38	GPIO0 pin 38 interrupt
39	GPIO_0_INTR39	GPIO0 pin 39 interrupt
40	GPIO_0_INTR40	GPIO0 pin 40 interrupt
41	GPIO_0_INTR41	GPIO0 pin 41 interrupt
42	GPIO_0_INTR42	GPIO0 pin 42 interrupt
43	GPIO_0_INTR43	GPIO0 pin 43 interrupt
44	GPIO_0_INTR44	GPIO0 pin 44 interrupt
45	GPIO_0_INTR45	GPIO0 pin 45 interrupt
46	GPIO_0_INTR46	GPIO0 pin 46 interrupt
47	GPIO_0_INTR47	GPIO0 pin 47 interrupt
48	GPIO_0_INTR48	GPIO0 pin 48 interrupt
49	GPIO_0_INTR49	GPIO0 pin 49 interrupt
50	GPIO_0_INTR50	GPIO0 pin 50 interrupt
51	GPIO_0_INTR51	GPIO0 pin 51 interrupt
52	GPIO_0_INTR52	GPIO0 pin 52 interrupt
53	GPIO_0_INTR53	GPIO0 pin 53 interrupt
54	GPIO_0_INTR54	GPIO0 pin 54 interrupt
55	GPIO_0_INTR55	GPIO0 pin 55 interrupt
56	GPIO_0_INTR56	GPIO0 pin 56 interrupt
57	GPIO_0_INTR57	GPIO0 pin 57 interrupt
58	GPIO_0_INTR58	GPIO0 pin 58 interrupt

Table 11-1012. GPIOMUX Input Event Mapping (continued)

Input Event	Event Name	Description
59	GPIO_0_INTR59	GPIO0 pin 59 interrupt
60	GPIO_0_INTR60	GPIO0 pin 60 interrupt
61	GPIO_0_INTR61	GPIO0 pin 61 interrupt
62	GPIO_0_INTR62	GPIO0 pin 62 interrupt
63	GPIO_0_INTR63	GPIO0 pin 63 interrupt
64	GPIO_0_INTR64	GPIO0 pin 64 interrupt
65	GPIO_0_INTR65	GPIO0 pin 65 interrupt
66	GPIO_0_INTR66	GPIO0 pin 66 interrupt
67	GPIO_0_INTR67	GPIO0 pin 67 interrupt
68	GPIO_0_INTR68	GPIO0 pin 68 interrupt
69	GPIO_0_INTR69	GPIO0 pin 69 interrupt
70	GPIO_0_INTR70	GPIO0 pin 70 interrupt
71	GPIO_0_INTR71	GPIO0 pin 71 interrupt
72	GPIO_0_INTR72	GPIO0 pin 72 interrupt
73	GPIO_0_INTR73	GPIO0 pin 73 interrupt
74	GPIO_0_INTR74	GPIO0 pin 74 interrupt
75	GPIO_0_INTR75	GPIO0 pin 75 interrupt
76	GPIO_0_INTR76	GPIO0 pin 76 interrupt
77	GPIO_0_INTR77	GPIO0 pin 77 interrupt
78	GPIO_0_INTR78	GPIO0 pin 78 interrupt
79	GPIO_0_INTR79	GPIO0 pin 79 interrupt
80	GPIO_0_INTR80	GPIO0 pin 80 interrupt
81	GPIO_0_INTR81	GPIO0 pin 81 interrupt
82	GPIO_0_INTR82	GPIO0 pin 82 interrupt
83	GPIO_0_INTR83	GPIO0 pin 83 interrupt
84	GPIO_0_INTR84	GPIO0 pin 84 interrupt
85	GPIO_0_INTR85	GPIO0 pin 85 interrupt
86	GPIO_0_INTR86	GPIO0 pin 86 interrupt
87	GPIO_0_INTR87	GPIO0 pin 87 interrupt
88	GPIO_0_INTR88	GPIO0 pin 88 interrupt
89	GPIO_0_INTR89	GPIO0 pin 89 interrupt
90	GPIO_0_INTR90	GPIO0 pin 90 interrupt
91	GPIO_0_INTR91	GPIO0 pin 91 interrupt
92	GPIO_0_INTR92	GPIO0 pin 92 interrupt
93	GPIO_0_INTR93	GPIO0 pin 93 interrupt
94	GPIO_0_INTR94	GPIO0 pin 94 interrupt
95	GPIO_0_INTR95	GPIO0 pin 95 interrupt
96	GPIO_0_INTR96	GPIO0 pin 96 interrupt
97	GPIO_0_INTR97	GPIO0 pin 97 interrupt
98	GPIO_0_INTR98	GPIO0 pin 98 interrupt
99	GPIO_0_INTR99	GPIO0 pin 99 interrupt
100	GPIO_0_INTR100	GPIO0 pin 100 interrupt
101	GPIO_0_INTR101	GPIO0 pin 101 interrupt
102	GPIO_0_INTR102	GPIO0 pin 102 interrupt
103	GPIO_0_INTR103	GPIO0 pin 103 interrupt
104	GPIO_0_INTR104	GPIO0 pin 104 interrupt
105	GPIO_0_INTR105	GPIO0 pin 105 interrupt

Table 11-1012. GPIOMUX Input Event Mapping (continued)

Input Event	Event Name	Description
106	GPIO_0_INTR106	GPIO0 pin 106 interrupt
107	GPIO_0_INTR107	GPIO0 pin 107 interrupt
108	GPIO_0_INTR108	GPIO0 pin 108 interrupt
109	GPIO_0_INTR109	GPIO0 pin 109 interrupt
110	GPIO_0_INTR110	GPIO0 pin 110 interrupt
111	GPIO_0_INTR111	GPIO0 pin 111 interrupt
112	GPIO_0_INTR112	GPIO0 pin 112 interrupt
113	GPIO_0_INTR113	GPIO0 pin 113 interrupt
114	GPIO_0_INTR114	GPIO0 pin 114 interrupt
115	GPIO_0_INTR115	GPIO0 pin 115 interrupt
116	GPIO_0_INTR116	GPIO0 pin 116 interrupt
117	GPIO_0_INTR117	GPIO0 pin 117 interrupt
118	GPIO_0_INTR118	GPIO0 pin 118 interrupt
119	GPIO_0_INTR119	GPIO0 pin 119 interrupt
120	GPIO_0_INTR120	GPIO0 pin 120 interrupt
121	GPIO_0_INTR121	GPIO0 pin 121 interrupt
122	GPIO_0_INTR122	GPIO0 pin 122 interrupt
123	GPIO_0_INTR123	GPIO0 pin 123 interrupt
124	GPIO_0_INTR124	GPIO0 pin 124 interrupt
125	GPIO_0_INTR125	GPIO0 pin 125 interrupt
126	GPIO_0_INTR126	GPIO0 pin 126 interrupt
127	GPIO_0_INTR127	GPIO0 pin 127 interrupt
128	GPIO_0_INTR128	GPIO0 pin 128 interrupt
129	GPIO_0_INTR129	GPIO0 pin 129 interrupt
130	GPIO_0_INTR130	GPIO0 pin 130 interrupt
131	GPIO_0_INTR131	GPIO0 pin 131 interrupt
132	GPIO_0_INTR132	GPIO0 pin 132 interrupt
133	GPIO_0_INTR133	GPIO0 pin 133 interrupt
134	GPIO_0_INTR134	GPIO0 pin 134 interrupt
135	GPIO_0_INTR135	GPIO0 pin 135 interrupt
136	GPIO_0_INTR136	GPIO0 pin 136 interrupt
137	GPIO_0_INTR137	GPIO0 pin 137 interrupt
138	GPIO_0_INTR138	GPIO0 pin 138 interrupt
139	GPIO_0_INTR139	GPIO0 pin 139 interrupt
140	GPIO_0_INTR140	GPIO0 pin 140 interrupt
141	GPIO_0_INTR141	GPIO0 pin 141 interrupt
142	GPIO_0_INTR142	GPIO0 pin 142 interrupt
143	GPIO_0_INTR143	GPIO0 pin 143 interrupt
144	GPIO_1_INTR0	GPIO1 pin 0 interrupt
145	GPIO_1_INTR1	GPIO1 pin 1 interrupt
146	GPIO_1_INTR2	GPIO1 pin 2 interrupt
147	GPIO_1_INTR3	GPIO1 pin 3 interrupt
148	GPIO_1_INTR4	GPIO1 pin 4 interrupt
149	GPIO_1_INTR5	GPIO1 pin 5 interrupt
150	GPIO_1_INTR6	GPIO1 pin 6 interrupt
151	GPIO_1_INTR7	GPIO1 pin 7 interrupt
152	GPIO_1_INTR8	GPIO1 pin 8 interrupt

Table 11-1012. GPIOMUX Input Event Mapping (continued)

Input Event	Event Name	Description
153	GPIO_1_INTR9	GPIO1 pin 9 interrupt
154	GPIO_1_INTR10	GPIO1 pin 10 interrupt
155	GPIO_1_INTR11	GPIO1 pin 11 interrupt
156	GPIO_1_INTR12	GPIO1 pin 12 interrupt
157	GPIO_1_INTR13	GPIO1 pin 13 interrupt
158	GPIO_1_INTR14	GPIO1 pin 14 interrupt
159	GPIO_1_INTR15	GPIO1 pin 15 interrupt
160	GPIO_1_INTR16	GPIO1 pin 16 interrupt
161	GPIO_1_INTR17	GPIO1 pin 17 interrupt
162	GPIO_1_INTR18	GPIO1 pin 18 interrupt
163	GPIO_1_INTR19	GPIO1 pin 19 interrupt
164	GPIO_1_INTR20	GPIO1 pin 20 interrupt
165	GPIO_1_INTR21	GPIO1 pin 21 interrupt
166	GPIO_1_INTR22	GPIO1 pin 22 interrupt
167	GPIO_1_INTR23	GPIO1 pin 23 interrupt
168	GPIO_1_INTR24	GPIO1 pin 24 interrupt
169	GPIO_1_INTR25	GPIO1 pin 25 interrupt
170	GPIO_1_INTR26	GPIO1 pin 26 interrupt
171	GPIO_1_INTR27	GPIO1 pin 27 interrupt
172	GPIO_1_INTR28	GPIO1 pin 28 interrupt
173	GPIO_1_INTR29	GPIO1 pin 29 interrupt
174	GPIO_1_INTR30	GPIO1 pin 30 interrupt
175	GPIO_1_INTR31	GPIO1 pin 31 interrupt
176	GPIO_1_INTR32	GPIO1 pin 32 interrupt
177	GPIO_1_INTR33	GPIO1 pin 33 interrupt
178	GPIO_1_INTR34	GPIO1 pin 34 interrupt
179	GPIO_1_INTR35	GPIO1 pin 35 interrupt
180	GPIO_1_INTR36	GPIO1 pin 36 interrupt
181	GPIO_1_INTR37	GPIO1 pin 37 interrupt
182	GPIO_1_INTR38	GPIO1 pin 38 interrupt
183	GPIO_1_INTR39	GPIO1 pin 39 interrupt
184	GPIO_1_INTR40	GPIO1 pin 40 interrupt
185	GPIO_1_INTR41	GPIO1 pin 41 interrupt
186	GPIO_1_INTR42	GPIO1 pin 42 interrupt
187	GPIO_1_INTR43	GPIO1 pin 43 interrupt
188	GPIO_1_INTR44	GPIO1 pin 44 interrupt
189	GPIO_1_INTR45	GPIO1 pin 45 interrupt
190	GPIO_1_INTR46	GPIO1 pin 46 interrupt
191	GPIO_1_INTR47	GPIO1 pin 47 interrupt
192	GPIO_1_INTR48	GPIO1 pin 48 interrupt
193	GPIO_1_INTR49	GPIO1 pin 49 interrupt
194	GPIO_1_INTR50	GPIO1 pin 50 interrupt
195	GPIO_1_INTR51	GPIO1 pin 51 interrupt
196	GPIO_1_INTR52	GPIO1 pin 52 interrupt
197	GPIO_1_INTR53	GPIO1 pin 53 interrupt
198	GPIO_1_INTR54	GPIO1 pin 54 interrupt
199	GPIO_1_INTR55	GPIO1 pin 55 interrupt

Table 11-1012. GPIOMUX Input Event Mapping (continued)

Input Event	Event Name	Description
200	GPIO_1_INTR56	GPIO1 pin 56 interrupt
201	GPIO_1_INTR57	GPIO1 pin 57 interrupt
202	GPIO_1_INTR58	GPIO1 pin 58 interrupt
203	GPIO_1_INTR59	GPIO1 pin 59 interrupt
204	GPIO_1_INTR60	GPIO1 pin 60 interrupt
205	GPIO_1_INTR61	GPIO1 pin 61 interrupt
206	GPIO_1_INTR62	GPIO1 pin 62 interrupt
207	GPIO_1_INTR63	GPIO1 pin 63 interrupt
208	GPIO_1_INTR64	GPIO1 pin 64 interrupt
209	GPIO_1_INTR65	GPIO1 pin 65 interrupt
210	GPIO_1_INTR66	GPIO1 pin 66 interrupt
211	GPIO_1_INTR67	GPIO1 pin 67 interrupt
212	RESERVED	Reserved
213	RESERVED	Reserved
214	RESERVED	Reserved
215	RESERVED	Reserved
216	RESERVED	Reserved
217	RESERVED	Reserved
218	RESERVED	Reserved
219	RESERVED	Reserved
220	RESERVED	Reserved
221	RESERVED	Reserved
222	RESERVED	Reserved
223	RESERVED	Reserved
224	ASRC_STREAM_IN_0_EVT	ASRC Stream 0 input event
225	ASRC_STREAM_IN_1_EVT	ASRC Stream 1 input event
226	ASRC_STREAM_IN_2_EVT	ASRC Stream 2 input event
227	ASRC_STREAM_IN_3_EVT	ASRC Stream 3 input event
228	ASRC_STREAM_IN_4_EVT	ASRC Stream 4 input event
229	ASRC_STREAM_IN_5_EVT	ASRC Stream 5 input event
230	ASRC_STREAM_IN_6_EVT	ASRC Stream 6 input event
231	ASRC_STREAM_IN_7_EVT	ASRC Stream 7 input event
232	ASRC_STREAM_IN_8_EVT	ASRC Stream 8 input event
233	ASRC_STREAM_IN_9_EVT	ASRC Stream 9 input event
234	ASRC_STREAM_IN_10_EVT	ASRC Stream 10 input event
235	ASRC_STREAM_IN_11_EVT	ASRC Stream 11 input event
236	ASRC_STREAM_IN_12_EVT	ASRC Stream 12 input event
237	ASRC_STREAM_IN_13_EVT	ASRC Stream 13 input event
238	ASRC_STREAM_IN_14_EVT	ASRC Stream 14 input event
239	ASRC_STREAM_IN_15_EVT	ASRC Stream 15 input event
240	ASRC_STREAM_OUT_0_EVT	ASRC Stream 0 output event
241	ASRC_STREAM_OUT_1_EVT	ASRC Stream 1 output event
242	ASRC_STREAM_OUT_2_EVT	ASRC Stream 2 output event
243	ASRC_STREAM_OUT_3_EVT	ASRC Stream 3 output event
244	ASRC_STREAM_OUT_4_EVT	ASRC Stream 4 output event
245	ASRC_STREAM_OUT_5_EVT	ASRC Stream 5 output event
246	ASRC_STREAM_OUT_6_EVT	ASRC Stream 6 output event

Table 11-1012. GPIOMUX Input Event Mapping (continued)

Input Event	Event Name	Description
247	ASRC_STREAM_OUT_7_EVT	ASRC Stream 7 output event
248	ASRC_STREAM_OUT_8_EVT	ASRC Stream 8 output event
249	ASRC_STREAM_OUT_9_EVT	ASRC Stream 9 output event
250	ASRC_STREAM_OUT_10_EVT	ASRC Stream 10 output event
251	ASRC_STREAM_OUT_11_EVT	ASRC Stream 11 output event
252	ASRC_STREAM_OUT_12_EVT	ASRC Stream 12 output event
253	ASRC_STREAM_OUT_13_EVT	ASRC Stream 13 output event
254	ASRC_STREAM_OUT_14_EVT	ASRC Stream 14 output event
255	ASRC_STREAM_OUT_15_EVT	ASRC Stream 15 output event

Refer to Section 5.1, *Control Module (BOOT_CFG)*, for more details on the GPIO interrupt muxes and connectivity.

11.7.4.5 Emulation Halt Operation

The GPIO peripheral is not affected by emulation halts.

11.7.5 GPIO Programming Guide

11.7.5.1 GPIO Low-Level Programming Models

11.7.5.1.1 Global Initialization

11.7.5.1.1.1 Surrounding Modules Global Initialization

This section identifies the requirements for initializing the surrounding modules when the general-purpose interface module is to be used for the first time after a device reset. This initialization of surrounding modules is based on the environment and integration of the general-purpose interface. For more information, see Section 11.7.2, *GPIO Environment*, and Section 11.7.3, *GPIO Integration*.

Table 11-1013. Global Initialization of Surrounding Modules

Surrounding Modules	Comments
LPSC0	Module reset must be enabled. For more information about the module configuration, see Section 5.2, *Power Management*.
PLL Controller	Interface and functional clocks must be enabled. For more information about the module configuration, see Section 5.4.5.3, *PLL Controller*.
ARM_GIC	ARM_GIC configuration must be done to enable the interrupts from the general-purpose interface module. See Section 6.1, *Arm Cortex-A15 Subsystem*.
CIC	CIC configuration must be done to enable the interrupts from the general-purpose interface module. See Chapter 9, *Interupts*.
GPIOMUX	GPIOMUX configuration must be done to enable the interrupts from the general-purpose interface module. See Section 5.1.3.1.7, *Event Mux Control Registers*.

11.7.5.1.1.2 GPIO Module Global Initialization

This procedure initializes the general-purpose Interface module after a power-on reset (POR) or software reset.

Table 11-1014. GPIO Global Initialization

Step	Register/Bit Field/Programming Model	Value
Configure GPIO channels as input or output of the corresponding bank	DIR	-h
Interrupt requests configuration		
Configure detection events	SET_RIS_TRIG and/or SET_FAL_TRIG	-h
Clear interrupt status	INTSTAT	FFFFh
Enable interrupts for desired banks [0:8]	GPIO_BINTEN[8:0]	-h

11.7.5.1.2 GPIO Operational Modes Configuration

11.7.5.1.2.1 GPIO Read Input Register

Table 11-1015. GPIO Read Input Register

Step	Register/Bit Field/Programming Model	Value
Read interrupt status of the corresponding bank	INTSTAT	-h
Read input register value	IN_DATA	-h
Clear interrupt status	INTSTAT	FFFF FFFFh

11.7.5.1.2.2 GPIO Set Bit Function

Table 11-1016. GPIO Set Bit Function

Step	Register/Bit Field/Programming Model	Value
Write 1h to set desired bit(s) in SET_DATA register.	SET_DATA	-h

11.7.5.1.2.3 GPIO Clear Bit Function

Table 11-1017. GPIO Clear Bit Function

Step	Register/Bit Field/Programming Model	Value
Write 1h to clear desired bit(s) in CLR_DATA register.	CLR_DATA	-h

11.7.6 GPIO Registers

11.7.6.1 GPIO Instance Summary

Table 11-1018 summarizes the general-purpose interface instance.

Table 11-1019 provides the register summary and associated offset addresses for the GPIO internal registers.

Table 11-1018. GPIO Instances

Instance	Base Address
GPIO_0	0260 3000h
GPIO_1	0260 A000h

Table 11-1019. GPIO Registers

Offset	Acronym	Register Name	GPIO_0 Physical Address	GPIO_1 Physical Address	Section
0h	GPIO_PID	Peripheral Identification Register	0260 3000h	0260 A000h	Section 11.7.6.2
8h	GPIO_BINTEN	Interrupt Per-Bank Enable Register	0260 3008h	0260 A008h	Section 11.7.6.3
10h	GPIO_DIR01	Direction 0 and 1 Register	0260 3010h	0260 A010h	Section 11.7.6.4
14h	GPIO_OUT_DATA01	Output Data 0 and 1 Register	0260 3014h	0260 A014h	Section 11.7.6.5
18h	GPIO_SET_DATA01	Set Data 0 and 1 Register	0260 3018h	0260 A018h	Section 11.7.6.6
1Ch	GPIO_CLR_DATA01	Clear Data 0 and 1 Register	0260 301Ch	0260 A01Ch	Section 11.7.6.7
20h	GPIO_IN_DATA01	Input Data 0 and 1 Register	0260 3020h	0260 A020h	Section 11.7.6.8
24h	GPIO_SET_RIS_TRIG01	Set Rising Edge Interrupt 0 and 1 Register	0260 3024h	0260 A024h	Section 11.7.6.9
28h	GPIO_CLR_RIS_TRIG01	Clear Rising Edge Interrupt 0 and 1 Register	0260 3028h	0260 A028h	Section 11.7.6.10
2Ch	GPIO_SET_FAL_TRIG01	Set Falling Edge Interrupt 0 and 1 Register	0260 302Ch	0260 A02Ch	Section 11.7.6.11
30h	GPIO_CLR_FAL_TRIG01	Clear Falling Edge Interrupt 0 and 1 Register	0260 3030h	0260 A030h	Section 11.7.6.12
34h	GPIO_INTSTAT01	GPIO Interrupt status 0 and 1 Register	0260 3034h	0260 A034h	Section 11.7.6.13
38h	GPIO_DIR23	Direction 2 and 3 Register	0260 3038h	0260 A038h	Section 11.7.6.14
3Ch	GPIO_OUT_DATA23	Output Data 2 and 3 Register	0260 303Ch	0260 A03Ch	Section 11.7.6.15
40h	GPIO_SET_DATA23	Set Data 2 and 3 Register	0260 3040h	0260 A040h	Section 11.7.6.16
44h	GPIO_CLR_DATA23	Clear Data 2 and 3 Register	0260 3044h	0260 A044h	Section 11.7.6.17
48h	GPIO_IN_DATA23	Input Data 2 and 3 Register	0260 3048h	0260 A048h	Section 11.7.6.18
4Ch	GPIO_SET_RIS_TRIG23	Set Rising Edge Interrupt 2 and 3 Register	0260 304Ch	0260 A04Ch	Section 11.7.6.19
50h	GPIO_CLR_RIS_TRIG23	Clear Rising Edge Interrupt 2 and 3 Register	0260 3050h	0260 A050h	Section 11.7.6.20
54h	GPIO_SET_FAL_TRIG23	Set Falling Edge Interrupt 2 and 3 Register	0260 3054h	0260 A054h	Section 11.7.6.21
58h	GPIO_CLR_FAL_TRIG23	Clear Falling Edge Interrupt 2 and 3 Register	0260 3058h	0260 A058h	Section 11.7.6.22
5Ch	GPIO_INTSTAT23	GPIO Interrupt status 2 and 3 Register	0260 305Ch	0260 A05Ch	Section 11.7.6.23
60h	GPIO_DIR45	Direction 4 and 5 Register	0260 3060h	0260 A060h	Section 11.7.6.24
64h	GPIO_OUT_DATA45	Output Data 4 and 5 Register	0260 3064h	0260 A064h	Section 11.7.6.25
68h	GPIO_SET_DATA45	Set Data 4 and 5 Register	0260 3068h	0260 A068h	Section 11.7.6.26
6Ch	GPIO_CLR_DATA45	Clear Data 4 and 5 Register	0260 306Ch	0260 A06Ch	Section 11.7.6.27
70h	GPIO_IN_DATA45	Input Data 4 and 5 Register	0260 3070h	0260 A070h	Section 11.7.6.28

Table 11-1019. GPIO Registers (continued)

Offset	Acronym	Register Name	GPIO_0 Physical Address	GPIO_1 Physical Address	Section
74h	GPIO_SET_RIS_TRIG45	Set Rising Edge Interrupt 4 and 5 Register	0260 3074h	0260 A074h	Section 11.7.6.29
78h	GPIO_CLR_RIS_TRIG45	Clear Rising Edge Interrupt 4 and 5 Register	0260 3078h	0260 A078h	Section 11.7.6.30
7Ch	GPIO_SET_FAL_TRIG45	Set Falling Edge Interrupt 4 and 5 Register	0260 307Ch	0260 A07Ch	Section 11.7.6.31
80h	GPIO_CLR_FAL_TRIG45	Clear Falling Edge Interrupt 4 and 5 Register	0260 3080h	0260 A080h	Section 11.7.6.32
84h	GPIO_INTSTAT45	GPIO Interrupt status 4 and 5 Register	0260 3084h	0260 A084h	Section 11.7.6.33
88h	GPIO_DIR67	Direction 6 and 7 Register	0260 3088h	-	Section 11.7.6.34
8Ch	GPIO_OUT_DATA67	Output Data 6 and 7 Register	0260 308Ch	-	Section 11.7.6.35
90h	GPIO_SET_DATA67	Set Data 6 and 7 Register	0260 3090h	-	Section 11.7.6.36
94h	GPIO_CLR_DATA67	Clear Data 6 and 7 Register	0260 3094h	-	Section 11.7.6.37
98h	GPIO_IN_DATA67	Input Data 6 and 7 Register	0260 3098h	-	Section 11.7.6.38
9Ch	GPIO_SET_RIS_TRIG67	Set Rising Edge Interrupt 6 and 7 Register	0260 309Ch	-	Section 11.7.6.39
A0h	GPIO_CLR_RIS_TRIG67	Clear Rising Edge Interrupt 6 and 7 Register	0260 30A0h	-	Section 11.7.6.40
A4h	GPIO_SET_FAL_TRIG67	Set Falling Edge Interrupt 6 and 7 Register	0260 30A4h	-	Section 11.7.6.41
A8h	GPIO_CLR_FAL_TRIG67	Clear Falling Edge Interrupt 6 and 7 Register	0260 30A8h	-	Section 11.7.6.42
ACh	GPIO_INTSTAT67	GPIO Interrupt status 6 and 7 Register	0260 30ACh	-	Section 11.7.6.43
B0h	GPIO_DIR8	Direction 8 Register	0260 30B0h	-	Section 11.7.6.44
B4h	GPIO_OUT_DATA8	Output Data 8 Register	0260 30B4h	-	Section 11.7.6.45
B8h	GPIO_SET_DATA8	Set Data 8 Register	0260 30B8h	-	Section 11.7.6.46
BCh	GPIO_CLR_DATA8	Clear Data 8 Register	0260 30BCh	-	Section 11.7.6.47
C0h	GPIO_IN_DATA8	Input Data 8 Register	0260 30C0h	-	Section 11.7.6.48
C4h	GPIO_SET_RIS_TRIG8	Set Rising Edge Interrupt 8 Register	0260 30C4h	-	Section 11.7.6.49
C8h	GPIO_CLR_RIS_TRIG8	Clear Rising Edge Interrupt 8 Register	0260 30C8h	-	Section 11.7.6.50
CCh	GPIO_SET_FAL_TRIG8	Set Falling Edge Interrupt 8 Register	0260 30CCh	-	Section 11.7.6.51
D0h	GPIO_CLR_FAL_TRIG8	Clear Falling Edge Interrupt 8 Register	0260 30D0h	-	Section 11.7.6.52
D4h	GPIO_INTSTAT8	GPIO Interrupt status 8 Register	0260 30D4h	-	Section 11.7.6.53

11.7.6.2 GPIO_PID Register (Offset = 0h) [reset = 4483 1105h]

Figure 11-485 shows the Peripheral Identification register and Table 11-1021 describes the fields.

Table 11-1020. GPIO_PID Instances

Instance	Physical Address
GPIO_0	0260 3000h
GPIO_1	0260 A000h

Figure 11-485. GPIO_PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-4483 1105h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1021. GPIO_PID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4483 1105h	TI internal data. Identifies revision of peripheral.

Table 11-1022. Register Call Summary for GPIO_PID

GPIO Registers
• GPIO Instance Summary: [0]

11.7.6.3 GPIO_BINTEN Register (Offset = 8h) [reset = 0h]

To use the GPIO pins as sources for CPU interrupts and EDMA events, bitfield[8-0] in the bank interrupt enable register (GPIO_BINTEN) must be set.

Figure 11-486 shows the Interrupt Per-Bank Enable register and Table 11-1024 describes the fields.

Table 11-1023. GPIO_BINTEN Instances

Instance	Physical Address
GPIO_0	0260 3008h
GPIO_1	0260 A008h

Figure 11-486. GPIO_BINTEN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							EN								
R-0h																							RW-0h								

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 11-1024. GPIO_BINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
8-0	EN	RW	0h	Enables GPIO pins in each bank as interrupt sources to the CPU. • 0 = Disables GPIO interrupts. • 1 = Enables GPIO interrupts.

Table 11-1025. Register Call Summary for GPIO_BINTEN

GPIO Functional Description • Trigger Configuration (per Bit): [0] • Interrupt Enable (per Bank): [0]
GPIO Programming Guide • GPIO Module Global Initialization: [0]
GPIO Registers • GPIO_BINTEN Register (Offset = 8h) [reset = 0h]: [0] • GPIO Instance Summary: [0]

11.7.6.4 GPIO_DIR01 Register (Offset = 10h) [reset = FFFF FFFFh]

The GPIO direction 0 and 1 register (GPIO_DIR01, GPIO[0:31] pins) determines if a given Bank0/1 GPIO pin is an input or an output. By default, all the GPIO pins are configured as input pins.

When GPIO pins are configured as output pins, the GPIO output buffer drives the GPIO pin. If it is necessary to place the GPIO output buffer in a high-impedance state, the GPIO pin must be configured as an input pin (DIRj = 1). At reset, GPIO pins default to input mode.

Figure 11-487 shows the Direction 0 and 1 register and Table 11-1027 describes the fields.

Table 11-1026. GPIO_DIR01 Instances

Instance	Physical Address
GPIO_0	0260 3010h
GPIO_1	0260 A010h

Figure 11-487. GPIO_DIR01 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIR																															
RW-FFFF FFFFh																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1027. GPIO_DIR01 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DIR	RW	FFFF FFFFh	Controls the direction of the GPIOj pin, j = 0 to 31. • 0 = GPIOj pin is configured as output pin. • 1 = GPIOj pin is configured as input pin. **Bits 0-15 correspond to GPIO0-15 of BANK0 and bits 16-31 correspond to GPIO0-15 of BANK1.**

Table 11-1028. Register Call Summary for GPIO_DIR01

GPIO Registers
- GPIO_DIR01 Register (Offset = 10h) [reset = FFFF FFFFh]: [0]
- GPIO Instance Summary: [0]

11.7.6.5 GPIO_OUT_DATA01 Register (Offset = 14h) [reset = 0h]

The GPIO Output Data 0 and 1 register (GPIO_OUT_DATA01, GPIO[0:31] pins) indicates the value to be driven on a given Bank0/1 GPIO output pin.

Figure 11-488 shows the Output Data 0 and 1 register and Table 11-1030 describes the fields.

Table 11-1029. GPIO_OUT_DATA01 Instances

Instance	Physical Address
GPIO_0	0260 3014h
GPIO_1	0260 A014h

Figure 11-488. GPIO_OUT_DATA01 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUT																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1030. GPIO_OUT_DATA01 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	OUT	RW	0h	Controls the drive state of the corresponding GPIOj pin, j = 0 to 31. These bits do not affect the state of the pin when the pin is configured as an input. Reading these bits returns the value of this register, not the state of the pin. **Bits 0-15 correspond to GPIO0-15 of BANK0 and bits 16-31 correspond to GPIO0-15 of BANK1.**

Table 11-1031. Register Call Summary for GPIO_OUT_DATA01

GPIO Functional Description • GPIO Function: [0][1][2][3]
GPIO Registers • GPIO_OUT_DATA01 Register (Offset = 14h) [reset = 0h]: [0] • GPIO Instance Summary: [0]

11.7.6.6 GPIO_SET_DATA01 Register (Offset = 18h) [reset = 0h]

The GPIO Set Data 0 and 1 register (GPIO_SET_DATA01, GPIO[0:31] pins) provides an alternate means of driving Bank0/1 GPIO outputs high. Writing a 1 to a bit of the Set Data 0 and 1 register sets the corresponding bit in the Output Data 0 and 1 register. Writing a 0 has no effect. Reading the Set Data 0 and 1 register returns the contents of Output Data 0 and 1 register.

Figure 11-489 shows the Set Data register and Table 11-1033 describes the fields.

Table 11-1032. GPIO_SET_DATA01 Instances

Instance	Physical Address
GPIO_0	0260 3018h
GPIO_1	0260 A018h

Figure 11-489. GPIO_SET_DATA01 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SET																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1033. GPIO_SET_DATA01 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SET	RW	0h	Writing 1 sets the corresponding bit in the Output Data 0 and 1 register. Reading this register returns the contents of the Output Data 0 and 1 register. Writing a 0 has no effect. • 0 = No effect. • 1 = Sets the corresponding bit in Output Data 0 and 1 register. **Bits 0-15 correspond to GPIO0-15 of BANK0 and bits 16-31 correspond to GPIO0-15 of BANK1.**

Table 11-1034. Register Call Summary for GPIO_SET_DATA01

GPIO Functional Description • GPIO Function: [0]
GPIO Registers • GPIO_SET_DATA01 Register (Offset = 18h) [reset = 0h]: [0] • GPIO Instance Summary: [0]

11.7.6.7 GPIO_CLR_DATA01 Register (Offset = 1Ch) [reset = 0h]

The GPIO Clear Data 0 and 1 register (GPIO_CLR_DATA01, GPIO[0:31] pins) provides an alternate means of driving Bank0/1 GPIO outputs low. Writing a 1 to a bit of the Clear Data 0 and 1 register clears the corresponding bit in the Output Data 0 and 1 register. Writing a 0 has no effect. Reading the Clear Data 0 and 1 register returns the contents of Output Data 0 and 1 register.

Figure 11-490 shows the Clear Data 0 and 1 register and Table 11-1036 describes the fields.

Table 11-1035. GPIO_CLR_DATA01 Instances

Instance	Physical Address
GPIO_0	0260 301Ch
GPIO_1	0260 A01Ch

Figure 11-490. GPIO_CLR_DATA01 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLR																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1036. GPIO_CLR_DATA01 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLR	RW	0h	Writing 1 clears the corresponding bit in the Output Data 0 and 1 register. Reading this register returns the contents of the Output Data 0 and 1 register. Writing a 0 has no effect. • 0 = No effect. • 1 = Clears the corresponding bit in Output Data 0 and 1 register. **Bits 0-15 correspond to GPIO0-15 of BANK0 and bits 16-31 correspond to GPIO0-15 of BANK1.**

Table 11-1037. Register Call Summary for GPIO_CLR_DATA01

GPIO Functional Description • GPIO Function: [0]
GPIO Registers • GPIO_CLR_DATA01 Register (Offset = 1Ch) [reset = 0h]: [0] • GPIO Instance Summary: [0]

11.7.6.8 GPIO_IN_DATA01 Register (Offset = 20h) [reset = 0h]

The GPIO Input Data 0 and 1 register (GPIO_IN_DATA01, GPIO[0:31] pins) reflects the state of the Bank0/1 GPIO pins. When read, the Input Data register 0 and 1 returns the state of the GPIO pins regardless of the state of the corresponding bits in the Direction and the Output Data 0 and 1 registers.

Figure 11-491 shows the Input Data 0 and 1 register and Table 11-1039 describes the fields.

Table 11-1038. GPIO_IN_DATA01 Instances

Instance	Physical Address
GPIO_0	0260 3020h
GPIO_1	0260 A020h

Figure 11-491. GPIO_IN_DATA01 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IN																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1039. GPIO_IN_DATA01 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IN	R	0h	Returns the status of the corresponding GPIOj pin, j = 0 to 31. **Bits 0-15 correspond to GPIO0-15 of BANK0 and bits 16-31 correspond to GPIO0-15 of BANK1.**

Table 11-1040. Register Call Summary for GPIO_IN_DATA01

GPIO Registers
- GPIO_IN_DATA01 Register (Offset = 20h) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.9 GPIO_SET_RIS_TRIG01 Register (Offset = 24h) [reset = 0h]

The GPIO rising trigger 0 and 1 register (RIS_TRIG01, GPIO[0:31] pins) configures the edge detection logic to trigger Bank0/1 GPIO interrupts and EDMA events on the rising edge of GPIO signals. Setting a bit to 1 in RIS_TRIG01 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the rising edge of GPIOj. RIS_TRIG01 is not directly accessible by the CPU; it must be configured using the GPIO set rising trigger and clear rising trigger registers.

Writing a 1 to a bit of GPIO_SET_RIS_TRIG01 sets the corresponding bit in RIS_TRIG01. Writing a 0 has no effect. Reading GPIO_SET_RIS_TRIG01 returns the value in RIS_TRIG01.

Figure 11-492 shows the Set Rising Edge Interrupt 0 and 1 register Table 11-1042 describes the fields.

Table 11-1041. GPIO_SET_RIS_TRIG01 Instances

Instance	Physical Address
GPIO_0	0260 3024h
GPIO_1	0260 A024h

Figure 11-492. GPIO_SET_RIS_TRIG01 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETRIS																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1042. GPIO_SET_RIS_TRIG01 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SETRIS	RW	0h	Writing a 1 enables the rising edge detection for the corresponding GPIOj pin, j = 0 to 31. Reading this register returns the state of the RIS_TRIG01 register. • 0 = No effect. • 1 = Sets the corresponding bit in RIS_TRIG01. **Bits 0-15 correspond to GPIO0-15 of BANK0 and bits 16-31 correspond to GPIO0-15 of BANK1.**

Table 11-1043. Register Call Summary for GPIO_SET_RIS_TRIG01

GPIO Registers
- GPIO_SET_RIS_TRIG01 Register (Offset = 24h) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.10 GPIO_CLR_RIS_TRIG01 Register (Offset = 28h) [reset = 0h]

The GPIO rising trigger 0 and 1 register (RIS_TRIG01, GPIO[0:31] pins) configures the edge detection logic to trigger Bank0/1 GPIO interrupts and EDMA events on the rising edge of GPIO signals. Setting a bit to 1 in RIS_TRIG causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the rising edge of GPIOj. RIS_TRIG01 is not directly accessible by the CPU; it must be configured using the GPIO set rising trigger and clear rising trigger registers.

Writing a 1 to a bit of GPIO_CLR_RIS_TRIG01 clears the corresponding bit in RIS_TRIG01. Writing a 0 has no effect. Reading GPIO_CLR_RIS_TRIG01 returns the value in RIS_TRIG01.

Figure 11-493 shows the Clear Rising Edge Interrupt 0 and 1 register and Table 11-1045 describes the fields.

Table 11-1044. GPIO_CLR_RIS_TRIG01 Instances

Instance	Physical Address
GPIO_0	0260 3028h
GPIO_1	0260 A028h

Figure 11-493. GPIO_CLR_RIS_TRIG01 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLRRIS																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1045. GPIO_CLR_RIS_TRIG01 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLRRIS	RW	0h	Writing a 1 disables the rising edge detection for the corresponding GPIOj pin, j = 0 to 31. Reading this register returns the state of the RIS_TRIG01 register. • 0 = No effect. • 1 = Clears the corresponding bit in RIS_TRIG01 **Bits 0-15 correspond to GPIO0-15 of BANK0 and bits 16-31 correspond to GPIO0-15 of BANK1.**

Table 11-1046. Register Call Summary for GPIO_CLR_RIS_TRIG01

GPIO Registers
- GPIO_CLR_RIS_TRIG01 Register (Offset = 28h) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.11 GPIO_SET_FAL_TRIG01 Register (Offset = 2Ch) [reset = 0h]

The GPIO falling trigger 0 and 1 register (FAL_TRIG01, GPIO[0:31] pins) configures the edge detection logic to trigger Bank0/1 GPIO interrupts and EDMA events on the falling edge of GPIO signals. Setting a bit to 1 in FAL_TRIG01 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the falling edge of GPIOj. FAL_TRIG01 is not directly accessible by the CPU; it must be configured using the GPIO set falling trigger 0 and 1 and clear falling trigger 0 and 1 registers.

Writing a 1 to a bit of GPIO_SET_FAL_TRIG01 sets the corresponding bit in FAL_TRIG01. Writing a 0 has no effect. Reading GPIO_SET_FAL_TRIG01 returns the value in FAL_TRIG01.

Figure 11-494 shows the Set Falling Edge Interrupt 0 and 1 register and Table 11-1048 describes the fields.

Table 11-1047. GPIO_SET_FAL_TRIG01 Instances

Instance	Physical Address
GPIO_0	0260 302Ch
GPIO_1	0260 A02Ch

Figure 11-494. GPIO_SET_FAL_TRIG01 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETFAL																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1048. GPIO_SET_FAL_TRIG01 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SETFAL	RW	0h	Writing a 1 enables the falling edge detection for the corresponding GPIOj pin, j = 0 to 31. Reading this register returns the state of the FAL_TRIG01 register. • 0 = No effect. • 1 = Sets the corresponding bit in FAL_TRIG01. **Bits 0-15 correspond to GPIO0-15 of BANK0 and bits 16-31 correspond to GPIO0-15 of BANK1.**

Table 11-1049. Register Call Summary for GPIO_SET_FAL_TRIG01

GPIO Registers
- GPIO_SET_FAL_TRIG01 Register (Offset = 2Ch) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.12 GPIO_CLR_FAL_TRIG01 Register (Offset = 30h) [reset = 0h]

The GPIO falling trigger 0 and 1 register (FAL_TRIG01, GPIO[0:31] pins) configures the edge detection logic to trigger Bank0/1 GPIO interrupts and EDMA events on the falling edge of GPIO signals. Setting a bit to 1 in FAL_TRIG01 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the falling edge of GPIOj. FAL_TRIG01 is not directly accessible by the CPU; it must be configured using the GPIO set falling trigger 0 and 1 and clear falling trigger 0 and 1 registers.

Writing a 1 to a bit of GPIO_CLR_FAL_TRIG01 clears the corresponding bit in FAL_TRIG01. Writing a 0 has no effect. Reading GPIO_CLR_FAL_TRIG01 returns the value in FAL_TRIG01.

Figure 11-496 shows the Clear Falling Edge Interrupt 0 and 1 register and Table 11-1054 describes the fields.

Table 11-1050. GPIO_CLR_FAL_TRIG01 Instances

Instance	Physical Address
GPIO_0	0260 3030h
GPIO_1	0260 A030h

Figure 11-495. GPIO_CLR_FAL_TRIG01 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLRFAL																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1051. GPIO_CLR_FAL_TRIG01 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLRFAL	RW	0h	Writing a 1 disables the falling edge detection for the corresponding GPIOj pin, j = 0 to 31. Reading this register returns the state of the FAL_TRIG01 register. • 0 = No effect. • 1 = Clears the corresponding bit in FAL_TRIG01. **Bits 0-15 correspond to GPIO0-15 of BANK0 and bits 16-31 correspond to GPIO0-15 of BANK1.**

Table 11-1052. Register Call Summary for GPIO_CLR_FAL_TRIG01

GPIO Registers
- GPIO_CLR_FAL_TRIG01 Register (Offset = 30h) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.13 GPIO_INTSTAT01 Register (Offset = 34h) [reset = 0h]

The GPIO_INTSTAT01 (GPIO[0:31] pins) register provides interrupt status upon reading, and interrupt clear feature upon writing 1 to the corresponding Bank0/1 bit position(s).

The interrupt status and clear feature is for the per bank interrupt connection scheme. Upon receiving an interrupt, the ISR can examine the interrupt status and clear the processed interrupts.

Note that the GPIO module generates an interrupt pulse on the individual GPINT interrupt in response to each occurrence of the specified edge condition. Therefore, for GPINT signals having their interrupts routed directly to the interrupt controller, it is not necessary to clear the status bits in this module. The interrupt status and clear register is a facility for the per-bank interrupt connection.

Also note, for the per-bank interrupt connection, ISR software should make sure that no triggering events escape detection.

Figure 11-496 shows the Interrupt Status 0 and 1 Register and Table 11-1054 describes the fields.

Table 11-1053. GPIO_INTSTAT01 Instances

Instance	Physical Address
GPIO_0	0260 3034h
GPIO_1	0260 A034h

Figure 11-496. GPIO_INTSTAT01 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STAT																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1054. GPIO_INTSTAT01 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STAT	RW	0h	Status of GPIO bank 0 and 1 interrupts. • Reading 1 = Interrupt occurred. • Reading 0 = Interrupt hasn't occurred since last clearing. • Writing 1 = Clears the corresponding interrupt status. **Bits 0-15 correspond to GPIO0-15 of BANK0 and bits 16-31 correspond to GPIO0-15 of BANK1.**

Table 11-1055. Register Call Summary for GPIO_INTSTAT01

GPIO Registers
- GPIO_INTSTAT01 Register (Offset = 34h) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.14 GPIO_DIR23 Register (Offset = 38h) [reset = FFFF FFFFh]

The GPIO direction 2 and 3 register (GPIO_DIR23, GPIO[32:63] pins) determines if a given Bank2/3 GPIO pin is an input or an output. By default, all the GPIO pins are configured as input pins.

When GPIO pins are configured as output pins, the GPIO output buffer drives the GPIO pin. If it is necessary to place the GPIO output buffer in a high-impedance state, the GPIO pin must be configured as an input pin (DIRj = 0). At reset, GPIO pins default to input mode.

Figure 11-497 shows the Direction 2 and 3 register and Table 11-1057 describes the fields.

Table 11-1056. GPIO_DIR23 Instances

Instance	Physical Address
GPIO_0	0260 3038h
GPIO_1	0260 A038h

Figure 11-497. GPIO_DIR23 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIR																															
RW-FFFF FFFFh																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1057. GPIO_DIR23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DIR	RW	FFFF FFFFh	Controls the direction of the GPIOj pins, j = 32 to 63. • 0 = GPIOj pin is configured as output pin. • 1 = GPIOj pin is configured as input pin. **Bits 0-15 correspond to GPIO0-15 of BANK2 and bits 16-31 correspond to GPIO0-15 of BANK3.**

Table 11-1058. Register Call Summary for GPIO_DIR23

GPIO Registers
- GPIO_DIR23 Register (Offset = 38h) [reset = FFFF FFFFh]: [0]
- GPIO Instance Summary: [0]

11.7.6.15 GPIO_OUT_DATA23 Register (Offset = 3Ch) [reset = 0h]

The GPIO Output Data 2 and 3 register (GPIO_OUT_DATA23, GPIO[32:63] pins) indicates the value to be driven on a given Bank2/3 GPIO output pin.

Figure 11-498 shows the Output Data 2 and 3 register and Table 11-1060 describes the fields.

Table 11-1059. GPIO_OUT_DATA23 Instances

Instance	Physical Address
GPIO_0	0260 303Ch
GPIO_1	0260 A03Ch

Figure 11-498. GPIO_OUT_DATA23 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUT																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1060. GPIO_OUT_DATA23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	OUT	RW	0h	Controls the drive state of the corresponding GPIOj pin, j = 32 to 63. These bits do not affect the state of the pin when the pin is configured as an input. Reading these bits returns the value of this register, not the state of the pin. **Bits 0-15 correspond to GPIO0-15 of BANK2 and bits 16-31 correspond to GPIO0-15 of BANK3.**

Table 11-1061. Register Call Summary for GPIO_OUT_DATA23

GPIO Registers
- GPIO_OUT_DATA23 Register (Offset = 3Ch) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.16 GPIO_SET_DATA23 Register (Offset = 40h) [reset = 0h]

The GPIO Set Data 2 and 3 register (GPIO_SET_DATA23, GPIO[32:63] pins) provides an alternate means of driving Bank2/3 GPIO outputs high. Writing a 1 to a bit of the Set Data 2 and 3 register sets the corresponding bit in the Output Data 2 and 3 register. Writing a 0 has no effect. Reading the Set Data 2 and 3 register returns the contents of Output Data 2 and 3 register.

Figure 11-499 shows the Set Data 2 and 3 register and Table 11-1063 describes the fields.

Table 11-1062. GPIO_SET_DATA23 Instances

Instance	Physical Address
GPIO_0	0260 3040h
GPIO_1	0260 A040h

Figure 11-499. GPIO_SET_DATA23 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SET																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1063. GPIO_SET_DATA23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SET	RW	0h	Writing 1 sets the corresponding bit in the Output Data 2 and 3 register. Reading this register returns the contents of the Output Data 2 and 3 register. Writing a 0 has no effect. • 0 = No effect. • 1 = Sets the corresponding bit in Output Data 2 and 3 register. **Bits 0-15 correspond to GPIO0-15 of BANK2 and bits 16-31 correspond to GPIO0-15 of BANK3.**

Table 11-1064. Register Call Summary for GPIO_SET_DATA23

GPIO Registers
- GPIO_SET_DATA23 Register (Offset = 40h) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.17 GPIO_CLR_DATA23 Register (Offset = 44h) [reset = 0h]

The GPIO Clear Data 2 and 3 register (GPIO_CLR_DATA23, GPIO[32:63] pins) provides an alternate means of driving Bank2/3 GPIO outputs low. Writing a 1 to a bit of the Clear Data 2 and 3 register clears the corresponding bit in the Output Data 2 and 3 register. Writing a 0 has no effect. Reading the Clear Data 2 and 3 register returns the contents of Output Data 2 and 3 register.

Figure 11-500 shows the Clear Data 2 and 3 register and Table 11-1066 describes the fields.

Table 11-1065. GPIO_CLR_DATA23 Instances

Instance	Physical Address
GPIO_0	0260 3044h
GPIO_1	0260 A044h

Figure 11-500. GPIO_CLR_DATA23 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLR																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1066. GPIO_CLR_DATA23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLR	RW	0h	Writing 1 clears the corresponding bit in the Output Data 2 and 3 register. Reading this register returns the contents of the Output Data 2 and 3 register. Writing a 0 has no effect. • 0 = No effect. • 1 = Clears the corresponding bit in Output Data 2 and 3 register. **Bits 0-15 correspond to GPIO0-15 of BANK2 and bits 16-31 correspond to GPIO0-15 of BANK3.**

Table 11-1067. Register Call Summary for GPIO_CLR_DATA23

GPIO Registers
- GPIO_CLR_DATA23 Register (Offset = 44h) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.18 GPIO_IN_DATA23 Register (Offset = 48h) [reset = 0h]

The GPIO Input Data 2 and 3 register (GPIO_IN_DATA23, GPIO[32:63] pins) reflects the state of the Bank2/3 GPIO pins. When read, the Input Data register returns the state of the GPIO pins regardless of the state of the corresponding bits in the Direction 2 and 3 and the Output Data 2 and 3 registers.

Figure 11-501 shows the Input Data 2 and 3 register and Table 11-1069 describes the fields.

Table 11-1068. GPIO_IN_DATA23 Instances

Instance	Physical Address
GPIO_0	0260 3048h
GPIO_1	0260 A048h

Figure 11-501. GPIO_IN_DATA23 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IN																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1069. GPIO_IN_DATA23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IN	R	0h	Returns the status of the corresponding GPIOj pin, j = 32 to 63. **Bits 0-15 correspond to GPIO0-15 of BANK2 and bits 16-31 correspond to GPIO0-15 of BANK3.**

Table 11-1070. Register Call Summary for GPIO_IN_DATA23

GPIO Registers
- GPIO_IN_DATA23 Register (Offset = 48h) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.19 GPIO_SET_RIS_TRIG23 Register (Offset = 4Ch) [reset = 0h]

The GPIO rising trigger 2 and 3 register (RIS_TRIG23, GPIO[32:63] pins) configures the edge detection logic to trigger Bank2/3 GPIO interrupts and EDMA events on the rising edge of GPIO signals. Setting a bit to 1 in RIS_TRIG23 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the rising edge of GPIO RIS_TRIG23 is not directly accessible by the CPU; it must be configured using the GPIO set rising trigger and clear rising trigger registers.

Writing a 1 to a bit of GPIO_SET_RIS_TRIG23 sets the corresponding bit in RIS_TRIG23. Writing a 0 has no effect. Reading GPIO_SET_RIS_TRIG23 returns the value in RIS_TRIG23.

Figure 11-502 shows the Set Rising Edge Interrupt 2 and 3 register Table 11-1072 describes the fields.

Table 11-1071. GPIO_SET_RIS_TRIG23 Instances

Instance	Physical Address
GPIO_0	0260 304Ch
GPIO_1	0260 A04Ch

Figure 11-502. GPIO_SET_RIS_TRIG23 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETRIS																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1072. GPIO_SET_RIS_TRIG23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SETRIS	RW	0h	Writing a 1 enables the rising edge detection for the corresponding GPIOj pin, j = 32 to 63. Reading this register returns the state of the RIS_TRIG23 register. • 0 = No effect. • 1 = Sets the corresponding bit in RIS_TRIG23. **Bits 0-15 correspond to GPIO0-15 of BANK2 and bits 16-31 correspond to GPIO0-15 of BANK3.**

Table 11-1073. Register Call Summary for GPIO_SET_RIS_TRIG23

GPIO Registers
- GPIO_SET_RIS_TRIG23 Register (Offset = 4Ch) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.20 GPIO_CLR_RIS_TRIG23 Register (Offset = 50h) [reset = 0h]

The GPIO rising trigger 2 and 3 register (RIS_TRIG23, GPIO[32:63] pins) configures the edge detection logic to trigger Bank2/3 GPIO interrupts and EDMA events on the rising edge of GPIO signals. Setting a bit to 1 in RIS_TRIG23 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the rising edge of GPIOj. RIS_TRIG23 is not directly accessible by the CPU; it must be configured using the GPIO set rising trigger and clear rising trigger registers.

Writing a 1 to a bit of GPIO_CLR_RIS_TRIG23 clears the corresponding bit in RIS_TRIG23. Writing a 0 has no effect. Reading GPIO_CLR_RIS_TRIG23 returns the value in RIS_TRIG23.

Figure 11-503 shows the Clear Rising Edge Interrupt 2 and 3 register and Table 11-1075 describes the fields.

Table 11-1074. GPIO_CLR_RIS_TRIG23 Instances

Instance	Physical Address
GPIO_0	0260 3050h
GPIO_1	0260 A050h

Figure 11-503. GPIO_CLR_RIS_TRIG23 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLRRIS																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1075. GPIO_CLR_RIS_TRIG23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLRRIS	RW	0h	Writing a 1 disables the rising edge detection for the corresponding GPIOj pin, j = 32 to 63. Reading this register returns the state of the RIS_TRIG23 register. • 0 = No effect. • 1 = Clears the corresponding bit in RIS_TRIG23. **Bits 0-15 correspond to GPIO0-15 of BANK2 and bits 16-31 correspond to GPIO0-15 of BANK3.**

Table 11-1076. Register Call Summary for GPIO_CLR_RIS_TRIG23

GPIO Registers
- GPIO_CLR_RIS_TRIG23 Register (Offset = 50h) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.21 GPIO_SET_FAL_TRIG23 Register (Offset = 54h) [reset = 0h]

The GPIO falling trigger 2 and 3 register (FAL_TRIG23, GPIO[32:63] pins) configures the edge detection logic to trigger Bank2/3 GPIO interrupts and EDMA events on the falling edge of GPIO signals. Setting a bit to 1 in FAL_TRIG23 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the falling edge of GPIOj. FAL_TRIG23 is not directly accessible by the CPU; it must be configured using the GPIO set falling trigger 2 and 3 and clear falling trigger 2 and 3 registers.

Writing a 1 to a bit of GPIO_SET_FAL_TRIG23 sets the corresponding bit in FAL_TRIG23. Writing a 0 has no effect. Reading GPIO_SET_FAL_TRIG23 returns the value in FAL_TRIG23.

Figure 11-504 shows the Set Falling Edge Interrupt 2 and 3 register and Table 11-1078 describes the fields.

Table 11-1077. GPIO_SET_FAL_TRIG23 Instances

Instance	Physical Address
GPIO_0	0260 3054h
GPIO_1	0260 A054h

Figure 11-504. GPIO_SET_FAL_TRIG23 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETFAL																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1078. GPIO_SET_FAL_TRIG23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SETFAL	RW	0h	Writing a 1 enables the falling edge detection for the corresponding GPIOj pin, j = 32 to 63. Reading this register returns the state of the FAL_TRIG23 register. • 0 = No effect. • 1 = Sets the corresponding bit in FAL_TRIG23. **Bits 0-15 correspond to GPIO0-15 of BANK2 and bits 16-31 correspond to GPIO0-15 of BANK3.**

Table 11-1079. Register Call Summary for GPIO_SET_FAL_TRIG23

GPIO Registers
- GPIO_SET_FAL_TRIG23 Register (Offset = 54h) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.22 GPIO_CLR_FAL_TRIG23 Register (Offset = 58h) [reset = 0h]

The GPIO falling trigger 2 and 3 register (FAL_TRIG23, GPIO[32:63] pins) configures the edge detection logic to trigger Bank2/3 GPIO interrupts and EDMA events on the falling edge of GPIO signals. Setting a bit to 1 in FAL_TRIG causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the falling edge of GPIOj. FAL_TRIG23 is not directly accessible by the CPU; it must be configured using the GPIO set falling trigger 2 and 3 and clear falling trigger 2 and 3 registers.

Writing a 1 to a bit of GPIO_CLR_FAL_TRIG23 clears the corresponding bit in FAL_TRIG23. Writing a 0 has no effect. Reading GPIO_CLR_FAL_TRIG23 returns the value in FAL_TRIG23.

Figure 11-506 shows the Clear Falling Edge Interrupt 2 and 3 register and Table 11-1084 describes the fields.

Table 11-1080. GPIO_CLR_FAL_TRIG23 Instances

Instance	Physical Address
GPIO_0	0260 3058h
GPIO_1	0260 A058h

Figure 11-505. GPIO_CLR_FAL_TRIG23 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLRFAL																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1081. GPIO_CLR_FAL_TRIG23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLRFAL	RW	0h	Writing a 1 disables the falling edge detection for the corresponding GPIOj pin, j = 32 to 63. Reading this register returns the state of the FAL_TRIG23 register. • 0 = No effect. • 1 = Clears the corresponding bit in FAL_TRIG23. **Bits 0-15 correspond to GPIO0-15 of BANK2 and bits 16-31 correspond to GPIO0-15 of BANK3.**

Table 11-1082. Register Call Summary for GPIO_CLR_FAL_TRIG23

GPIO Registers
- GPIO_CLR_FAL_TRIG23 Register (Offset = 58h) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.23 GPIO_INTSTAT23 Register (Offset = 5Ch) [reset = 0h]

The GPIO_INTSTAT23 (GPIO[32:63] pins) register provides interrupt status upon reading, and interrupt clear feature upon writing 1 to the corresponding Bank2/3 bit position(s).

The interrupt status and clear feature is for the per bank interrupt connection scheme. Upon receiving an interrupt, the ISR can examine the interrupt status and clear the processed interrupts.

Note that the GPIO module generates an interrupt pulse on the individual GPINT interrupt in response to each occurrence of the specified edge condition. Therefore, for GPINT signals having their interrupts routed directly to the interrupt controller, it is not necessary to clear the status bits in this module. The interrupt status and clear register is a facility for the per-bank interrupt connection.

Also note, for the per-bank interrupt connection, ISR software should make sure that no triggering events escape detection.

Figure 11-506 shows the Interrupt Status Register 2 and 3 and Table 11-1084 describes the fields.

Table 11-1083. GPIO_INTSTAT23 Instances

Instance	Physical Address
GPIO_0	0260 305Ch
GPIO_1	0260 A05Ch

Figure 11-506. GPIO_INTSTAT23 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STAT																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1084. GPIO_INTSTAT23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STAT	RW	0h	Status of GPIO bank 2 and 3 interrupts. • Reading 1 = Interrupt occurred. • Reading 0 = Interrupt hasn't occurred since last clearing. • Writing 1 = Clears the corresponding interrupt status. **Bits 0-15 correspond to GPIO0-15 of BANK2 and bits 16-31 correspond to GPIO0-15 of BANK3.**

Table 11-1085. Register Call Summary for GPIO_INTSTAT23

GPIO Registers
- GPIO_INTSTAT23 Register (Offset = 5Ch) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.24 GPIO_DIR45 Register (Offset = 60h) [reset = FFFFFFFFh]

The GPIO direction 4 and 5 register (GPIO_DIR45, GPIO[64:95] pins) determines if a given Bank4/5 GPIO pin is an input or an output. By default, all the GPIO pins are configured as input pins.

When GPIO pins are configured as output pins, the GPIO output buffer drives the GPIO pin. If it is necessary to place the GPIO output buffer in a high-impedance state, the GPIO pin must be configured as an input pin (DIRj = 0). At reset, GPIO pins default to input mode.

Figure 11-507 shows the Direction 4 and 5 register and Table 11-1087 describes the fields.

Table 11-1086. GPIO_DIR45 Instances

Instance	Physical Address
GPIO_0	0260 3060h
GPIO_1	0260 A060h

Figure 11-507. GPIO_DIR45 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIR																															
RW-FFFFFFFFh																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1087. GPIO_DIR45 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DIR	RW	FFFFFFFFh	Controls the direction of the GPIOj pin, j = 64 to 95. • 0 = GPIOj pin is configured as output pin. • 1 = GPIOj pin is configured as input pin. **Bits 0-15 correspond to GPIO0-15 of BANK4 and bits 16-31 correspond to GPIO0-15 of BANK5.**

Table 11-1088. Register Call Summary for GPIO_DIR45

GPIO Registers
- GPIO_DIR45 Register (Offset = 60h) [reset = FFFFFFFFh]: [0]
- GPIO Instance Summary: [0]

11.7.6.25 GPIO_OUT_DATA45 Register (Offset = 64h) [reset = 0h]

The GPIO Output Data 4 and 5 register (GPIO_OUT_DATA45, GPIO[64:95] pins) indicates the value to be driven on a given Bank4/5 GPIO output pin.

Figure 11-508 shows the Output Data 4 and 5 register and Table 11-1090 describes the fields.

Table 11-1089. GPIO_OUT_DATA45 Instances

Instance	Physical Address
GPIO_0	0260 3064h
GPIO_1	0260 A064h

Figure 11-508. GPIO_OUT_DATA45 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUT																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1090. GPIO_OUT_DATA45 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	OUT	RW	0h	Controls the drive state of the corresponding GPIOj pin, j = 64 to 95. These bits do not affect the state of the pin when the pin is configured as an input. Reading these bits returns the value of this register, not the state of the pin. **Bits 0-15 correspond to GPIO0-15 of BANK4 and bits 16-31 correspond to GPIO0-15 of BANK5.**

Table 11-1091. Register Call Summary for GPIO_OUT_DATA45

GPIO Registers
- GPIO_OUT_DATA45 Register (Offset = 64h) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.26 GPIO_SET_DATA45 Register (Offset = 68h) [reset = 0h]

The GPIO Set Data 4 and 5 register (GPIO_SET_DATA45, GPIO[64:95] pins) provides an alternate means of driving Bank4/5 GPIO outputs high. Writing a 1 to a bit of the Set Data register sets the corresponding bit in the Output Data 4 and 5 register. Writing a 0 has no effect. Reading the Set Data 4 and 5 register returns the contents of Output Data 4 and 5 register.

Figure 11-509 shows the Set Data 4 and 5 register and Table 11-1093 describes the fields.

Table 11-1092. GPIO_SET_DATA45 Instances

Instance	Physical Address
GPIO_0	0260 3068h
GPIO_1	0260 A068h

Figure 11-509. GPIO_SET_DATA45 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SET																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1093. GPIO_SET_DATA45 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SET	RW	0h	Writing 1 sets the corresponding bit in the Output Data 4 and 5 register. Reading this register returns the contents of the Output Data 4 and 5 register. Writing a 0 has no effect. • 0 = No effect. • 1 = Sets the corresponding bit in Output Data 4 and 5 register. **Bits 0-15 correspond to GPIO0-15 of BANK4 and bits 16-31 correspond to GPIO0-15 of BANK5.**

Table 11-1094. Register Call Summary for GPIO_SET_DATA45

GPIO Registers
- GPIO_SET_DATA45 Register (Offset = 68h) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.27 GPIO_CLR_DATA45 Register (Offset = 6Ch) [reset = 0h]

The GPIO Clear Data 4 and 5 register (GPIO_CLR_DATA45, GPIO[64:95] pins) provides an alternate means of driving Bank4/5 GPIO outputs low. Writing a 1 to a bit of the Clear Data 4 and 5 register clears the corresponding bit in the Output Data4 and 5 register. Writing a 0 has no effect. Reading the Clear Data 4 and 5 register returns the contents of Output Data 4 and 5 register.

Figure 11-510 shows the Clear Data 4 and 5 register and Table 11-1096 describes the fields.

Table 11-1095. GPIO_CLR_DATA45 Instances

Instance	Physical Address
GPIO_0	0260 306Ch
GPIO_1	0260 A06Ch

Figure 11-510. GPIO_CLR_DATA45 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLR																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1096. GPIO_CLR_DATA45 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLR	RW	0h	Writing 1 clears the corresponding bit in the Output Data 4 and 5 register. Reading this register returns the contents of the Output Data 4 and 5 register. Writing a 0 has no effect. • 0 = No effect. • 1 = Clears the corresponding bit in Output Data 4 and 5 register. **Bits 0-15 correspond to GPIO0-15 of BANK4 and bits 16-31 correspond to GPIO0-15 of BANK5.**

Table 11-1097. Register Call Summary for GPIO_CLR_DATA45

GPIO Registers
- GPIO_CLR_DATA45 Register (Offset = 6Ch) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.28 GPIO_IN_DATA45 Register (Offset = 70h) [reset = 0h]

The GPIO Input Data 4 and 5 register (GPIO_IN_DATA45, GPIO[64:95] pins) reflects the state of the Bank4/5 GPIO pins. When read, the Input Data 4 and 5 register returns the state of the GPIO pins regardless of the state of the corresponding bits in the Direction 4 and 5 and the Output Data 4 and 5 registers.

Figure 11-511 shows the Input Data 4 and 5 register and Table 11-1099 describes the fields.

Table 11-1098. GPIO_IN_DATA45 Instances

Instance	Physical Address
GPIO_0	0260 3070h
GPIO_1	0260 A070h

Figure 11-511. GPIO_IN_DATA45 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IN																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1099. GPIO_IN_DATA45 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IN	R	0h	Returns the status of the corresponding GPIOj pin, j = 64 to 95. **Bits 0-15 correspond to GPIO0-15 of BANK4 and bits 16-31 correspond to GPIO0-15 of BANK5.**

Table 11-1100. Register Call Summary for GPIO_IN_DATA45

GPIO Registers
- GPIO_IN_DATA45 Register (Offset = 70h) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.29 GPIO_SET_RIS_TRIG45 Register (Offset = 74h) [reset = 0h]

The GPIO rising trigger 4 and 5 register (RIS_TRIG45, GPIO[64:95] pins) configures the edge detection logic to trigger Bank4/5 GPIO interrupts and EDMA events on the rising edge of GPIO signals. Setting a bit to 1 in RIS_TRIG45 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the rising edge of GPIOj. RIS_TRIG45 is not directly accessible by the CPU; it must be configured using the GPIO set rising trigger and clear rising trigger registers.

Writing a 1 to a bit of GPIO_SET_RIS_TRIG45 sets the corresponding bit in RIS_TRIG45. Writing a 0 has no effect. Reading GPIO_SET_RIS_TRIG45 returns the value in RIS_TRIG45.

Figure 11-512 shows the Set Rising Edge Interrupt 4 and 5 register Table 11-1102 describes the fields.

Table 11-1101. GPIO_SET_RIS_TRIG45 Instances

Instance	Physical Address
GPIO_0	0260 3074h
GPIO_1	0260 A074h

Figure 11-512. GPIO_SET_RIS_TRIG45 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETRIS																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1102. GPIO_SET_RIS_TRIG45 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SETRIS	RW	0h	Writing a 1 enables the rising edge detection for the corresponding GPIOj pin, j = 64 to 95. Reading this register returns the state of the RIS_TRIG45 register. • 0 = No effect. • 1 = Sets the corresponding bit in RIS_TRIG45. **Bits 0-15 correspond to GPIO0-15 of BANK4 and bits 16-31 correspond to GPIO0-15 of BANK5.**

Table 11-1103. Register Call Summary for GPIO_SET_RIS_TRIG45

GPIO Registers
- GPIO_SET_RIS_TRIG45 Register (Offset = 74h) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.30 GPIO_CLR_RIS_TRIG45 Register (Offset = 78h) [reset = 0h]

The GPIO rising trigger 4 and 5 register (RIS_TRIG45, GPIO[64:95] pins) configures the edge detection logic to trigger Bank4/5 GPIO interrupts and EDMA events on the rising edge of GPIO signals. Setting a bit to 1 in RIS_TRIG45 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the rising edge of GPIOj. RIS_TRIG45 is not directly accessible by the CPU; it must be configured using the GPIO set rising trigger and clear rising trigger registers.

Writing a 1 to a bit of GPIO_CLR_RIS_TRIG45 clears the corresponding bit in RIS_TRIG45. Writing a 0 has no effect. Reading GPIO_CLR_RIS_TRIG45 returns the value in RIS_TRIG45.

Figure 11-513 shows the Clear Rising Edge Interrupt 4 and 5 register and Table 11-1105 describes the fields.

Table 11-1104. GPIO_CLR_RIS_TRIG45 Instances

Instance	Physical Address
GPIO_0	0260 3078h
GPIO_1	0260 A078h

Figure 11-513. GPIO_CLR_RIS_TRIG45 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLRRIS																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1105. GPIO_CLR_RIS_TRIG45 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLRRIS	RW	0h	Writing a 1 disables the rising edge detection for the corresponding GPIOj pin, j = 64 to 95. Reading this register returns the state of the RIS_TRIG45 register. • 0 = No effect. • 1 = Clears the corresponding bit in RIS_TRIG45. **Bits 0-15 correspond to GPIO0-15 of BANK4 and bits 16-31 correspond to GPIO0-15 of BANK5.**

Table 11-1106. Register Call Summary for GPIO_CLR_RIS_TRIG45

GPIO Registers
- GPIO_CLR_RIS_TRIG45 Register (Offset = 78h) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.31 GPIO_SET_FAL_TRIG45 Register (Offset = 7Ch) [reset = 0h]

The GPIO falling trigger 4 and 5 register (FAL_TRIG45, GPIO[64:95] pins) configures the edge detection logic to trigger Bank4/5 GPIO interrupts and EDMA events on the falling edge of GPIO signals. Setting a bit to 1 in FAL_TRIG45 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the falling edge of GPIOj. FAL_TRIG45 is not directly accessible by the CPU; it must be configured using the GPIO set falling trigger 4 and 5 and clear falling trigger 4 and 5 registers.

Writing a 1 to a bit of GPIO_SET_FAL_TRIG45 sets the corresponding bit in FAL_TRI45G. Writing a 0 has no effect. Reading GPIO_SET_FAL_TRIG45 returns the value in FAL_TRIG45.

Figure 11-514 shows the Set Falling Edge Interrupt 4 and 5 register and Table 11-1108 describes the fields.

Table 11-1107. GPIO_SET_FAL_TRIG45 Instances

Instance	Physical Address
GPIO_0	0260 307Ch
GPIO_1	0260 A07Ch

Figure 11-514. GPIO_SET_FAL_TRIG45 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETFAL																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1108. GPIO_SET_FAL_TRIG45 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SETFAL	RW	0h	Writing a 1 enables the falling edge detection for the corresponding GPIOj pin, j = 64 to 95. Reading this register returns the state of the FAL_TRIG45 register. • 0 = No effect. • 1 = Sets the corresponding bit in FAL_TRIG45. **Bits 0-15 correspond to GPIO0-15 of BANK4 and bits 16-31 correspond to GPIO0-15 of BANK5.**

Table 11-1109. Register Call Summary for GPIO_SET_FAL_TRIG45

GPIO Registers
- GPIO_SET_FAL_TRIG45 Register (Offset = 7Ch) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.32 GPIO_CLR_FAL_TRIG45 Register (Offset = 80h) [reset = 0h]

The GPIO falling trigger 4 and 5 register (FAL_TRIG45, GPIO[64:95] pins) configures the edge detection logic to trigger Bank4/5 GPIO interrupts and EDMA events on the falling edge of GPIO signals. Setting a bit to 1 in FAL_TRIG45 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the falling edge of GPIO. FAL_TRIG45 is not directly accessible by the CPU; it must be configured using the GPIO set falling trigger 4 and 5 and clear falling trigger 4 and 5 registers.

Writing a 1 to a bit of GPIO_CLR_FAL_TRIG45 clears the corresponding bit in FAL_TRIG45. Writing a 0 has no effect. Reading GPIO_CLR_FAL_TRIG45 returns the value in FAL_TRIG45.

Figure 11-516 shows the Clear Falling Edge Interrupt 4 and 5 register and Table 11-1114 describes the fields.

Table 11-1110. GPIO_CLR_FAL_TRIG45 Instances

Instance	Physical Address
GPIO_0	0260 3080h
GPIO_1	0260 A080h

Figure 11-515. GPIO_CLR_FAL_TRIG45 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLRFAL																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1111. GPIO_CLR_FAL_TRIG45 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLRFAL	RW	0h	Writing a 1 disables the falling edge detection for the corresponding GPIOj pin, j = 64 to 95. Reading this register returns the state of the FAL_TRIG45 register. • 0 =No effect. • 1 = Clears the corresponding bit in FAL_TRIG45. **Bits 0-15 correspond to GPIO0-15 of BANK4 and bits 16-31 correspond to GPIO0-15 of BANK5.**

Table 11-1112. Register Call Summary for GPIO_CLR_FAL_TRIG45

GPIO Registers
- GPIO_CLR_FAL_TRIG45 Register (Offset = 80h) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.33 GPIO_INTSTAT45 Register (Offset = 84h) [reset = 0h]

The GPIO_INTSTAT45 (GPIO[64:95] pins) register provides interrupt status upon reading, and interrupt clear feature upon writing 1 to the corresponding Bank4/5 bit position(s).

The interrupt status and clear feature is for the per bank interrupt connection scheme. Upon receiving an interrupt, the ISR can examine the interrupt status and clear the processed interrupts.

Note that the GPIO module generates an interrupt pulse on the individual GPINT interrupt in response to each occurrence of the specified edge condition. Therefore, for GPINT signals having their interrupts routed directly to the interrupt controller, it is not necessary to clear the status bits in this module. The interrupt status and clear register is a facility for the per-bank interrupt connection.

Also note, for the per-bank interrupt connection, ISR software should make sure that no triggering events escape detection.

Figure 11-516 shows the Interrupt Status Register 4 and 5 and Table 11-1114 describes the fields.

Table 11-1113. GPIO_INTSTAT45 Instances

Instance	Physical Address
GPIO_0	0260 3084h
GPIO_1	0260 A084h

Figure 11-516. GPIO_INTSTAT45 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STAT																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1114. GPIO_INTSTAT45 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STAT	RW	0h	Status of GPIO bank 4 and 5 interrupts. • Reading 1 = Interrupt occurred. • Reading 0 = Interrupt hasn't occurred since last clearing. • Writing 1 = Clears the corresponding interrupt status. **Bits 0-15 correspond to GPIO0-15 of BANK4 and bits 16-31 correspond to GPIO0-15 of BANK5.**

Table 11-1115. Register Call Summary for GPIO_INTSTAT45

GPIO Registers
- GPIO_INTSTAT45 Register (Offset = 84h) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.34 GPIO_DIR67 Register (Offset = 88h) [reset = FFFF FFFFh]

The GPIO direction 6 and 7register (GPIO_DIR67, GPIO[96:127] pins) determines if a given Bank6/7 GPIO pin is an input or an output. By default, all the GPIO pins are configured as input pins.

When GPIO pins are configured as output pins, the GPIO output buffer drives the GPIO pin. If it is necessary to place the GPIO output buffer in a high-impedance state, the GPIO pin must be configured as an input pin (DIRj = 0). At reset, GPIO pins default to input mode.

Figure 11-517 shows the Direction 6 and 7 register and Table 11-1117 describes the fields.

Table 11-1116. GPIO_DIR67 Instances

Instance	Physical Address
GPIO_0	0260 3088h

Figure 11-517. GPIO_DIR67 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIR																															
RW-FFFF FFFFh																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1117. GPIO_DIR67 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DIR	RW	FFFF FFFFh	Controls the direction of the GPIOj pin, j = 96 to 127. • 0 = GPIOj pin is configured as output pin. • 1 = GPIOj pin is configured as input pin. **Bits 0-15 correspond to GPIO0-15 of BANK6 and bits 16-31 correspond to GPIO0-15 of BANK7.**

Table 11-1118. Register Call Summary for GPIO_DIR67

GPIO Registers
- GPIO_DIR67 Register (Offset = 88h) [reset = FFFF FFFFh]: [0]
- GPIO Instance Summary: [0]

11.7.6.35 GPIO_OUT_DATA67 Register (Offset = 8Ch) [reset = 0h]

The GPIO Output Data 6 and 7 register (GPIO_OUT_DATA67, GPIO[96:127] pins) indicates the value to be driven on a given Bank6/7 GPIO output pin.

Figure 11-518 shows the Output Data 6 and 7 register and Table 11-1120 describes the fields.

Table 11-1119. GPIO_OUT_DATA67 Instances

Instance	Physical Address
GPIO_0	0260 308Ch

Figure 11-518. GPIO_OUT_DATA67 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUT																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1120. GPIO_OUT_DATA67 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	OUT	RW	0h	Controls the drive state of the corresponding GPIOj pin, j = 96 to 127. These bits do not affect the state of the pin when the pin is configured as an input. Reading these bits returns the value of this register, not the state of the pin. **Bits 0-15 correspond to GPIO0-15 of BANK6 and bits 16-31 correspond to GPIO0-15 of BANK7.**

Table 11-1121. Register Call Summary for GPIO_OUT_DATA67

GPIO Registers
- GPIO_OUT_DATA67 Register (Offset = 8Ch) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.36 GPIO_SET_DATA67 Register (Offset = 90h) [reset = 0h]

The GPIO Set Data 6 and 7 register (GPIO_SET_DATA67, GPIO[96:127] pins) provides an alternate means of driving Bank6/7 GPIO outputs high. Writing a 1 to a bit of the Set Data 6 and 7 register sets the corresponding bit in the Output Data 6 and 7 register. Writing a 0 has no effect. Reading the Set Data 6 and 7 register returns the contents of Output Data 6 and 7 register.

Figure 11-519 shows the Set Data 6 and 7 register and Table 11-1123 describes the fields.

Table 11-1122. GPIO_SET_DATA67 Instances

Instance	Physical Address
GPIO_0	0260 3090h

Figure 11-519. GPIO_SET_DATA67 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SET																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1123. GPIO_SET_DATA67 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SET	RW	0h	Writing 1 sets the corresponding bit in the Output Data 6 and 7 register. Reading this register returns the contents of the Output Data 6 and 7 register. Writing a 0 has no effect. • 0 = No effect. • 1 = Sets the corresponding bit in Output Data 6 and 7 register. **Bits 0-15 correspond to GPIO0-15 of BANK6 and bits 16-31 correspond to GPIO0-15 of BANK7.**

Table 11-1124. Register Call Summary for GPIO_SET_DATA67

GPIO Registers
- GPIO_SET_DATA67 Register (Offset = 90h) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.37 GPIO_CLR_DATA67 Register (Offset = 94h) [reset = 0h]

The GPIO Clear Data 6 and 7 register (GPIO_CLR_DATA67, GPIO[96:127] pins) provides an alternate means of driving Bank6/7 GPIO outputs low. Writing a 1 to a bit of the Clear Data 6 and 7 register clears the corresponding bit in the Output Data 6 and 7 register. Writing a 0 has no effect. Reading the Clear Data 6 and 7 register returns the contents of Output Data 6 and 7register.

Figure 11-520 shows the Clear Data 6 and 7 register and Table 11-1126 describes the fields.

Table 11-1125. GPIO_CLR_DATA67 Instances

Instance	Physical Address
GPIO_0	0260 3094h

Figure 11-520. GPIO_CLR_DATA67 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLR																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1126. GPIO_CLR_DATA67 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLR	RW	0h	Writing 1 clears the corresponding bit in the Output Data 6 and 7 register. Reading this register returns the contents of the Output Data 6 and 7 register. Writing a 0 has no effect. • 0 = No effect. • 1 = Clears the corresponding bit in Output Data 6 and 7 register. **Bits 0-15 correspond to GPIO0-15 of BANK6 and bits 16-31 correspond to GPIO0-15 of BANK7.**

Table 11-1127. Register Call Summary for GPIO_CLR_DATA67

GPIO Registers
- GPIO_CLR_DATA67 Register (Offset = 94h) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.38 GPIO_IN_DATA67 Register (Offset = 98h) [reset = 0h]

The GPIO Input Data 6 and 7 register (GPIO_IN_DATA67, GPIO[96:127] pins) reflects the state of the Bank6/7 GPIO pins. When read, the Input Data register returns the state of the GPIO pins regardless of the state of the corresponding bits in the Direction and the Output Data 6 and 7 registers.

Figure 11-521 shows the Input Data 6 and 7 register and Table 11-1129 describes the fields.

Table 11-1128. GPIO_IN_DATA67 Instances

Instance	Physical Address
GPIO_0	0260 3098h

Figure 11-521. GPIO_IN_DATA67 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IN																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1129. GPIO_IN_DATA67 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IN	R	0h	Returns the status of the corresponding GPIOj pin, j = 96 to 127. **Bits 0-15 correspond to GPIO0-15 of BANK6 and bits 16-31 correspond to GPIO0-15 of BANK7.**

Table 11-1130. Register Call Summary for GPIO_IN_DATA67

GPIO Registers
- GPIO_IN_DATA67 Register (Offset = 98h) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.39 GPIO_SET_RIS_TRIG67 Register (Offset = 9Ch) [reset = 0h]

The GPIO rising trigger register (RIS_TRIG67, GPIO[96:127] pins) configures the edge detection logic to trigger Bank6/7 GPIO interrupts and EDMA events on the rising edge of GPIO signals. Setting a bit to 1 in RIS_TRIG67 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the rising edge of GPIOj. RIS_TRIG67 is not directly accessible by the CPU; it must be configured using the GPIO set rising trigger and clear rising trigger registers.

Writing a 1 to a bit of GPIO_SET_RIS_TRIG67 sets the corresponding bit in RIS_TRIG67. Writing a 0 has no effect. Reading GPIO_SET_RIS_TRIG67 returns the value in RIS_TRIG67.

Figure 11-522 shows the Set Rising Edge Interrupt 6 and 7 register Table 11-1132 describes the fields.

Table 11-1131. GPIO_SET_RIS_TRIG67 Instances

Instance	Physical Address
GPIO_0	0260 309Ch

Figure 11-522. GPIO_SET_RIS_TRIG67 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETRIS																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1132. GPIO_SET_RIS_TRIG67 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SETRIS	RW	0h	Writing a 1 enables the rising edge detection for the corresponding GPIOj pin, j = 96 to 127. Reading this register returns the state of the RIS_TRIG67 register. • 0 = No effect. • 1 = Sets the corresponding bit in RIS_TRIG67. **Bits 0-15 correspond to GPIO0-15 of BANK6 and bits 16-31 correspond to GPIO0-15 of BANK7.**

Table 11-1133. Register Call Summary for GPIO_SET_RIS_TRIG67

GPIO Registers
- GPIO_SET_RIS_TRIG67 Register (Offset = 9Ch) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.40 GPIO_CLR_RIS_TRIG67 Register (Offset = A0h) [reset = 0h]

The GPIO rising trigger 6 and 7 register (RIS_TRIG67, GPIO[96:127] pins) configures the edge detection logic to trigger Bank6/7 GPIO interrupts and EDMA events on the rising edge of GPIO signals. Setting a bit to 1 in RIS_TRIG67 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the rising edge of GPIOj. RIS_TRIG67 is not directly accessible by the CPU; it must be configured using the GPIO set rising trigger and clear rising trigger registers.

Writing a 1 to a bit of GPIO_CLR_RIS_TRIG67 clears the corresponding bit in RIS_TRIG67. Writing a 0 has no effect. Reading GPIO_CLR_RIS_TRIG67 returns the value in RIS_TRIG67.

Figure 11-523 shows the Clear Rising Edge Interrupt 6 and 7 register and Table 11-1135 describes the fields.

Table 11-1134. GPIO_CLR_RIS_TRIG67 Instances

Instance	Physical Address
GPIO_0	0260 30A0h

Figure 11-523. GPIO_CLR_RIS_TRIG67 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLRRIS																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1135. GPIO_CLR_RIS_TRIG67 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLRRIS	RW	0h	Writing a 1 disables the rising edge detection for the corresponding GPIOj pin, j = 96 to 127. Reading this register returns the state of the RIS_TRIG67 register. • 0 = No effect. • 1 = Clears the corresponding bit in RIS_TRIG67. **Bits 0-15 correspond to GPIO0-15 of BANK6 and bits 16-31 correspond to GPIO0-15 of BANK7.**

Table 11-1136. Register Call Summary for GPIO_CLR_RIS_TRIG67

GPIO Registers
- GPIO_CLR_RIS_TRIG67 Register (Offset = A0h) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.41 GPIO_SET_FAL_TRIG67 Register (Offset = A4h) [reset = 0h]

The GPIO falling trigger 6 and 7 register (FAL_TRIG67, GPIO[96:127] pins) configures the edge detection logic to trigger Bank6/7 GPIO interrupts and EDMA events on the falling edge of GPIO signals. Setting a bit to 1 in FAL_TRIG67 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the falling edge of GPIOj. FAL_TRIG67 is not directly accessible by the CPU; it must be configured using the GPIO set falling trigger 6 and 7 and clear falling trigger 6 and 7 registers.

Writing a 1 to a bit of GPIO_SET_FAL_TRIG67 sets the corresponding bit in FAL_TRIG67. Writing a 0 has no effect. Reading GPIO_SET_FAL_TRIG67 returns the value in FAL_TRIG67.

Figure 11-524 shows the Set Falling Edge Interrupt 6 and 7 register and Table 11-1138 describes the fields.

Table 11-1137. GPIO_SET_FAL_TRIG67 Instances

Instance	Physical Address
GPIO_0	0260 30A4h

Figure 11-524. GPIO_SET_FAL_TRIG67 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETFAL																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1138. GPIO_SET_FAL_TRIG67 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SETFAL	RW	0h	Writing a 1 enables the falling edge detection for the corresponding GPIOj pin, j = 96 to 127. Reading this register returns the state of the FAL_TRIG67 register. • 0 = No effect. • 1 = Sets the corresponding bit in FAL_TRIG67. **Bits 0-15 correspond to GPIO0-15 of BANK6 and bits 16-31 correspond to GPIO0-15 of BANK7.**

Table 11-1139. Register Call Summary for GPIO_SET_FAL_TRIG67

GPIO Registers
- GPIO_SET_FAL_TRIG67 Register (Offset = A4h) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.42 GPIO_CLR_FAL_TRIG67 Register (Offset = A8h) [reset = 0h]

The GPIO falling trigger 6 and 7 register (FAL_TRIG67, GPIO[96:127] pins) configures the edge detection logic to trigger Bank6/7 GPIO interrupts and EDMA events on the falling edge of GPIO signals. Setting a bit to 1 in FAL_TRIG67 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the falling edge of GPIOj. FAL_TRIG67 is not directly accessible by the CPU; it must be configured using the GPIO set falling trigger 6 and 7 and clear falling trigger 6 and 7 registers.

Writing a 1 to a bit of GPIO_CLR_FAL_TRIG67 clears the corresponding bit in FAL_TRIG67. Writing a 0 has no effect. Reading GPIO_CLR_FAL_TRIG67 returns the value in FAL_TRIG67.

Figure 11-526 shows the Clear Falling Edge Interrupt 6 and 7 register and Table 11-1144 describes the fields.

Table 11-1140. GPIO_CLR_FAL_TRIG67 Instances

Instance	Physical Address
GPIO_0	0260 30A8h

Figure 11-525. GPIO_CLR_FAL_TRIG67 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLRFAL																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1141. GPIO_CLR_FAL_TRIG67 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLRFAL	RW	0h	Writing a 1disables the falling edge detection for the corresponding GPIOj pin, j = 96 to 127. Reading this register returns the state of the FAL_TRIG67 register. • 0 =No effect. • 1 = Clears the corresponding bit in FAL_TRIG67. **Bits 0-15 correspond to GPIO0-15 of BANK6 and bits 16-31 correspond to GPIO0-15 of BANK7.**

Table 11-1142. Register Call Summary for GPIO_CLR_FAL_TRIG67

GPIO Registers
- GPIO_CLR_FAL_TRIG67 Register (Offset = A8h) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.43 GPIO_INTSTAT67 Register (Offset = ACh) [reset = 0h]

The GPIO_INTSTAT67 (GPIO[96:127] pins) register provides interrupt status upon reading, and interrupt clear feature upon writing 1 to the corresponding Bank6/7 bit position(s).

The interrupt status and clear feature is for the per bank interrupt connection scheme. Upon receiving an interrupt, the ISR can examine the interrupt status and clear the processed interrupts.

Note that the GPIO module generates an interrupt pulse on the individual GPINT interrupt in response to each occurrence of the specified edge condition. Therefore, for GPINT signals having their interrupts routed directly to the interrupt controller, it is not necessary to clear the status bits in this module. The interrupt status and clear register is a facility for the per-bank interrupt connection.

Also note, for the per-bank interrupt connection, ISR software should make sure that no triggering events escape detection.

Figure 11-526 shows the Interrupt Status Register 6 and 7 and Table 11-1144 describes the fields.

Table 11-1143. GPIO_INTSTAT67 Instances

Instance	Physical Address
GPIO_0	0260 30ACh

Figure 11-526. GPIO_INTSTAT67 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STAT																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-1144. GPIO_INTSTAT67 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STAT	RW	0h	Status of GPIO bank 6 and 7 interrupts. • Reading 1 = interrupt occurred. • Reading 0 = interrupt hasn't occurred since last clearing. • Writing 1 = Clears the corresponding interrupt status. **Bits 0-15 correspond to GPIO0-15 of BANK6 and bits 16-31 correspond to GPIO0-15 of BANK7.**

Table 11-1145. Register Call Summary for GPIO_INTSTAT67

GPIO Registers
- GPIO_INTSTAT67 Register (Offset = ACh) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.44 GPIO_DIR8 Register (Offset = B0h) [reset = FFFFh]

The GPIO direction 8 register (GPIO_DIR8, GPIO[128:143] pins) determines if a given Bank8 GPIO pin is an input or an output. By default, all the GPIO pins are configured as input pins.

When GPIO pins are configured as output pins, the GPIO output buffer drives the GPIO pin. If it is necessary to place the GPIO output buffer in a high-impedance state, the GPIO pin must be configured as an input pin (DIRj = 0). At reset, GPIO pins default to input mode.

Figure 11-527 shows the Direction 8 register and Table 11-1147 describes the fields.

Table 11-1146. GPIO_DIR8 Instances

Instance	Physical Address
GPIO_0	0260 30B0h

Figure 11-527. GPIO_DIR8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																DIR															
R-0h																RW-FFFFh															

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 11-1147. GPIO_DIR8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	RESERVED
15-0	DIR	RW	FFFFh	Controls the direction of the GPIOj pin, j = 128 to 143. • 0 = GPIOj pin is configured as output pin. • 1 = GPIOj pin is configured as input pin. **Bits 0-15 correspond to GPIO0-15 of BANK8.**

Table 11-1148. Register Call Summary for GPIO_DIR8

GPIO Registers
- GPIO_DIR8 Register (Offset = B0h) [reset = FFFFh]: [0]
- GPIO Instance Summary: [0]

11.7.6.45 GPIO_OUT_DATA8 Register (Offset = B4h) [reset = 0h]

The GPIO Output Data register (GPIO_OUT_DATA8, GPIO[128:143] pins) indicates the value to be driven on a given Bank8 GPIO output pin.

Figure 11-528 shows the Output Data 8 register and Table 11-1150 describes the fields.

Table 11-1149. GPIO_OUT_DATA8 Instances

Instance	Physical Address
GPIO_0	0260 30B4h

Figure 11-528. GPIO_OUT_DATA8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																OUT															
R-0h																RW-0h															

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 11-1150. GPIO_OUT_DATA8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	RESERVED
15-0	OUT	RW	0h	Controls the drive state of the corresponding GPIOj pin, j = 128 to 143. These bits do not affect the state of the pin when the pin is configured as an input. Reading these bits returns the value of this register, not the state of the pin. **Bits 0-15 correspond to GPIO0-15 of BANK8.**

Table 11-1151. Register Call Summary for GPIO_OUT_DATA8

GPIO Registers
- GPIO_OUT_DATA8 Register (Offset = B4h) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.46 GPIO_SET_DATA8 Register (Offset = B8h) [reset = 0h]

The GPIO Set Data 8 register (GPIO_SET_DATA8, GPIO[128:143] pins) provides an alternate means of driving Bank8 GPIO outputs high. Writing a 1 to a bit of the Set Data 8 register sets the corresponding bit in the Output Data 8 register. Writing a 0 has no effect. Reading the Set Data 8 register returns the contents of Output Data 8 register.

Figure 11-529 shows the Set Data 8 register and Table 11-1153 describes the fields.

Table 11-1152. GPIO_SET_DATA8 Instances

Instance	Physical Address
GPIO_0	0260 30B8h

Figure 11-529. GPIO_SET_DATA8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																SET															
R-0h																RW-0h															

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 11-1153. GPIO_SET_DATA8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	RESERVED
15-0	SET	RW	0h	Writing 1 sets the corresponding bit in the Output Data 8 register. Reading this register returns the contents of the Output Data 8 register. Writing a 0 has no effect. • 0 = No effect. • 1 = Sets the corresponding bit in Output Data 8 register. **Bits 0-15 correspond to GPIO0-15 of BANK8.**

Table 11-1154. Register Call Summary for GPIO_SET_DATA8

GPIO Registers
- GPIO_SET_DATA8 Register (Offset = B8h) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.47 GPIO_CLR_DATA8 Register (Offset = BCh) [reset = 0h]

The GPIO Clear Data 8 register (GPIO_CLR_DATA8, GPIO[128:143] pins) provides an alternate means of driving Bank8 GPIO outputs low. Writing a 1 to a bit of the Clear Data 8 register clears the corresponding bit in the Output Data 8 register. Writing a 0 has no effect. Reading the Clear Data 8 register returns the contents of Output Data 8 register.

Figure 11-530 shows the Clear Data 8 register and Table 11-1156 describes the fields.

Table 11-1155. GPIO_CLR_DATA8 Instances

Instance	Physical Address
GPIO_0	0260 30BCh

Figure 11-530. GPIO_CLR_DATA8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CLR															
R-0h																RW-0h															

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 11-1156. GPIO_CLR_DATA8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	RESERVED
15-0	CLR	RW	0h	Writing 1 clears the corresponding bit in the Output Data 8 register. Reading this register returns the contents of the Output Data 8 register. Writing a 0 has no effect. • 0 = No effect. • 1 = Clears the corresponding bit in Output Data 8 register. **Bits 0-15 correspond to GPIO0-15 of BANK8.**

Table 11-1157. Register Call Summary for GPIO_CLR_DATA8

GPIO Registers
- GPIO_CLR_DATA8 Register (Offset = BCh) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.48 GPIO_IN_DATA8 Register (Offset = C0h) [reset = 0h]

The GPIO Input Data 8 register (GPIO_IN_DATA8, GPIO[128:143] pins) reflects the state of the Bank8 GPIO pins. When read, the Input Data 8 register returns the state of the GPIO pins regardless of the state of the corresponding bits in the Direction 8 and the Output Data 8 registers.

Figure 11-531 shows the Input Data 8 register and Table 11-1159 describes the fields.

Table 11-1158. GPIO_IN_DATA8 Instances

Instance	Physical Address
GPIO_0	0260 30C0h

Figure 11-531. GPIO_IN_DATA8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																IN															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1159. GPIO_IN_DATA8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	RESERVED
15-0	IN	R	0h	Returns the status of the corresponding GPIOj pin, j = 128 to 143. **Bits 0-15 correspond to GPIO0-15 of BANK8.**

Table 11-1160. Register Call Summary for GPIO_IN_DATA8

GPIO Registers
- GPIO_IN_DATA8 Register (Offset = C0h) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.7.6.49 GPIO_SET_RIS_TRIG8 Register (Offset = C4h) [reset = 0h]

The GPIO rising trigger 8 register (RIS_TRIG8, GPIO[128:143] pins) configures the edge detection logic to trigger Bank8 GPIO interrupts and EDMA events on the rising edge of GPIO signals. Setting a bit to 1 in RIS_TRIG8 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the rising edge of GPIOj. RIS_TRIG8 is not directly accessible by the CPU; it must be configured using the GPIO set rising trigger 8 and clear rising trigger 8 registers.

Writing a 1 to a bit of GPIO_SET_RIS_TRIG8 sets the corresponding bit in RIS_TRIG8. Writing a 0 has no effect. Reading GPIO_SET_RIS_TRIG8 returns the value in RIS_TRIG8.

Figure 11-532 shows the Set Rising Edge Interrupt 8 register Table 11-1162 describes the fields.

Table 11-1161. GPIO_SET_RIS_TRIG8 Instances

Instance	Physical Address
GPIO_0	0260 30C4h

Figure 11-532. GPIO_SET_RIS_TRIG8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																SETRIS															
R-0h																RW-0h															

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 11-1162. GPIO_SET_RIS_TRIG8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	RESERVED
15-0	SETRIS	RW	0h	Writing a 1 enables the rising edge detection for the corresponding GPIOj pin, j = 128 to 143. Reading this register returns the state of the RIS_TRIG8 register. • 0 = No effect. • 1 = Sets the corresponding bit in RIS_TRIG8. **Bits 0-15 correspond to GPIO0-15 of BANK8.**

Table 11-1163. Register Call Summary for GPIO_SET_RIS_TRIG8

GPIO Registers
- GPIO_SET_RIS_TRIG8 Register (Offset = C4h) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.50 GPIO_CLR_RIS_TRIG8 Register (Offset = C8h) [reset = 0h]

The GPIO rising trigger 8 register (RIS_TRIG8, GPIO[128:143] pins) configures the edge detection logic to trigger Bank8 GPIO interrupts and EDMA events on the rising edge of GPIO signals. Setting a bit to 1 in RIS_TRIG8 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the rising edge of GPIOj. RIS_TRIG8 is not directly accessible by the CPU; it must be configured using the GPIO set rising trigger 8 and clear rising trigger 8 registers.

Writing a 1 to a bit of GPIO_CLR_RIS_TRIG8 clears the corresponding bit in RIS_TRIG8. Writing a 0 has no effect. Reading GPIO_CLR_RIS_TRIG8 returns the value in RIS_TRIG8.

Figure 11-533 shows the Clear Rising Edge Interrupt 8 register and Table 11-1165 describes the fields.

Table 11-1164. GPIO_CLR_RIS_TRIG8 Instances

Instance	Physical Address
GPIO_0	0260 30C8h

Figure 11-533. GPIO_CLR_RIS_TRIG8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CLRRIS															
R-0h																RW-0h															

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 11-1165. GPIO_CLR_RIS_TRIG8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	RESERVED
15-0	CLRRIS	RW	0h	Writing a 1 disables the rising edge detection for the corresponding GPIOj pin, j = 128 to 143. Reading this register returns the state of the RIS_TRIG8 register. • 0 = No effect. • 1 = Clears the corresponding bit in RIS_TRIG8. **Bits 0-15 correspond to GPIO0-15 of BANK8.**

Table 11-1166. Register Call Summary for GPIO_CLR_RIS_TRIG8

GPIO Registers
- GPIO_CLR_RIS_TRIG8 Register (Offset = C8h) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.51 GPIO_SET_FAL_TRIG8 Register (Offset = CCh) [reset = 0h]

The GPIO falling trigger 8 register (FAL_TRIG8, GPIO[128:143] pins) configures the edge detection logic to trigger Bank8 GPIO interrupts and EDMA events on the falling edge of GPIO signals. Setting a bit to 1 in FAL_TRIG8 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the falling edge of GPIOj. FAL_TRIG8 is not directly accessible by the CPU; it must be configured using the GPIO set falling trigger 8 and clear falling trigger 8 registers.

Writing a 1 to a bit of GPIO_SET_FAL_TRIG8 sets the corresponding bit in FAL_TRIG8. Writing a 0 has no effect. Reading GPIO_SET_FAL_TRIG8 returns the value in FAL_TRIG8.

Figure 11-534 shows the Set Falling Edge Interrupt 8 register and Table 11-1168 describes the fields.

Table 11-1167. GPIO_SET_FAL_TRIG8 Instances

Instance	Physical Address
GPIO_0	0260 30CCh

Figure 11-534. GPIO_SET_FAL_TRIG8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																SETFAL															
R-0h																RW-0h															

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 11-1168. GPIO_SET_FAL_TRIG8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	RESERVED
15-0	SETFAL	RW	0h	Writing a 1 enables the falling edge detection for the corresponding GPIOj pin, j = 128 to 143. Reading this register returns the state of the FAL_TRIG8 register. • 0 = No effect. • 1 = Sets the corresponding bit in FAL_TRIG8. **Bits 0-15 correspond to GPIO0-15 of BANK8.**

Table 11-1169. Register Call Summary for GPIO_SET_FAL_TRIG8

GPIO Registers
- GPIO_SET_FAL_TRIG8 Register (Offset = CCh) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.52 GPIO_CLR_FAL_TRIG8 Register (Offset = D0h) [reset = 0h]

The GPIO falling trigger 8 register (FAL_TRIG8, GPIO[128:143] pins) configures the edge detection logic to trigger Bank8 GPIO interrupts and EDMA events on the falling edge of GPIO signals. Setting a bit to 1 in FAL_TRIG8 causes the corresponding GPIO interrupt and EDMA event (GPINTj) to be generated on the falling edge of GPIOj. FAL_TRIG8 is not directly accessible by the CPU; it must be configured using the GPIO set falling trigger 8 and clear falling trigger 8 registers.

Writing a 1 to a bit of GPIO_CLR_FAL_TRIG8 clears the corresponding bit in FAL_TRIG8. Writing a 0 has no effect. Reading GPIO_CLR_FAL_TRIG8 returns the value in FAL_TRIG8.

Figure 11-536 shows the Clear Falling Edge Interrupt 8 register and Table 11-1174 describes the fields.

Table 11-1170. GPIO_CLR_FAL_TRIG8 Instances

Instance	Physical Address
GPIO_0	0260 30D0h

Figure 11-535. GPIO_CLR_FAL_TRIG8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CLRFAL															
R-0h																RW-0h															

LEGEND: R = Read Only; RW = Read/Write; *-n* = value after reset

Table 11-1171. GPIO_CLR_FAL_TRIG8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	RESERVED
15-0	CLRFAL	RW	0h	Writing a 1 disables the falling edge detection for the corresponding GPIOj pin, j = 128 to 143. Reading this register returns the state of the FAL_TRIG8 register. • 0 =No effect. • 1 = Clears the corresponding bit in FAL_TRIG8. **Bits 0-15 correspond to GPIO0-15 of BANK8.**

Table 11-1172. Register Call Summary for GPIO_CLR_FAL_TRIG8

GPIO Registers
- GPIO_CLR_FAL_TRIG8 Register (Offset = D0h) [reset = 0h]: [0][1]
- GPIO Instance Summary: [0]

11.7.6.53 GPIO_INTSTAT8 Register (Offset = D4h) [reset = 0h]

The GPIO_INTSTAT8 (GPIO[128:143] pins) register provides interrupt status upon reading, and interrupt clear feature upon writing 1 to the corresponding Bank8 bit position(s).

The interrupt status and clear feature is for the per bank interrupt connection scheme. Upon receiving an interrupt, the ISR can examine the interrupt status and clear the processed interrupts.

Note that the GPIO module generates an interrupt pulse on the individual GPINT interrupt in response to each occurrence of the specified edge condition. Therefore, for GPINT signals having their interrupts routed directly to the interrupt controller, it is not necessary to clear the status bits in this module. The interrupt status and clear register is a facility for the per-bank interrupt connection.

Also note, for the per-bank interrupt connection, ISR software should make sure that no triggering events escape detection.

Figure 11-536 shows the Interrupt Status 8 Register and Table 11-1174 describes the fields.

Table 11-1173. GPIO_INTSTAT8 Instances

Instance	Physical Address
GPIO_0	0260 30D4h

Figure 11-536. GPIO_INTSTAT8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																STAT															
R-0h																RW-0h															

LEGEND: R = Read Only; RW = Read/Write; -*n* = value after reset

Table 11-1174. GPIO_INTSTAT8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	RESERVED
15-0	STAT	RW	0h	Status of GPIO bank 8 interrupts. • Reading 1 = Interrupt occurred. • Reading 0 = Interrupt hasn't occurred since last clearing. • Writing 1 = Clears the corresponding interrupt status. **Bits 0-15 correspond to GPIO0-15 of BANK8.**

Table 11-1175. Register Call Summary for GPIO_INTSTAT8

GPIO Registers
- GPIO_INTSTAT8 Register (Offset = D4h) [reset = 0h]: [0]
- GPIO Instance Summary: [0]

11.8 Inter-IC Module (I2C)

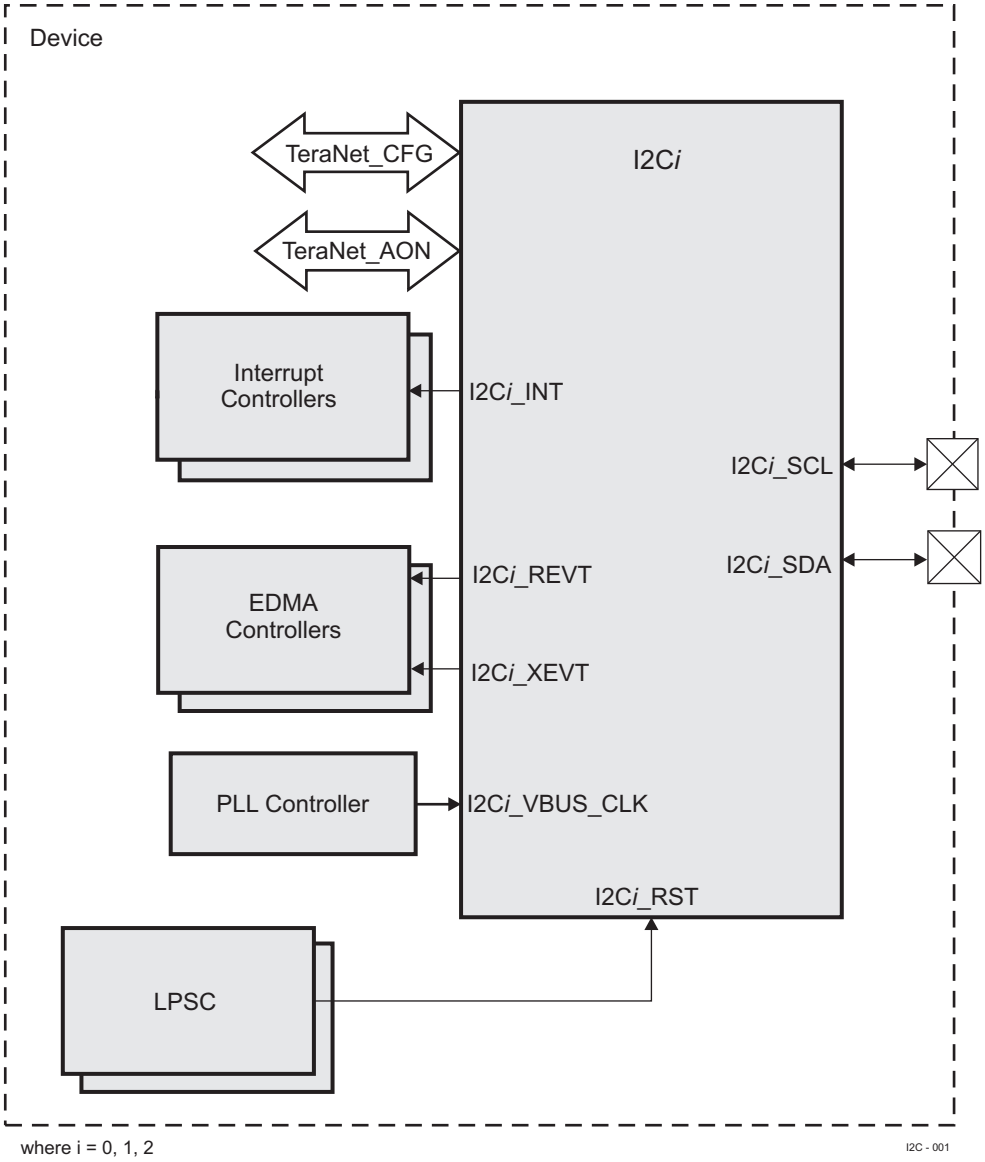
This section describes the inter-integrated circuit (I2C) module in the device.

11.8.1 I2C Overview

The multi-master inter-integrated circuit (I2C) peripheral provides an interface between the device and any I^2C-bus-compatible device that is connected via the I^2C serial bus. External components attached to the I^2C bus can serially transmit/receive up to 8-bit data to/from the device through the two-wire I^2C interface.

Figure 11-537 shows an overview of the I2C modules in the device.

Figure 11-537. I2C Modules Overview



11.8.1.1 Features

Each I2C module has the following features:

- Compliance with the Philips Semiconductors I^2C-bus specification (version 2.1):
  - Supports standard mode (up to 100 kbps) and fast mode (up to 400 kbps)
  - Support for byte format transfer
  - 7-bit addressing mode
  - General call
  - START byte mode
  - Support for multiple master-transmitters and slave-receivers mode
  - Support for multiple slave-transmitters and master-receivers mode
  - Combined master transmit/receive and receive/transmit mode
- 2 to 7 bit format transfer
- Free data format mode
- One read DMA event and one write DMA event that can be used by the DMA
- Seven interrupts that can be used by the CPU
- Module enable/disable capability

I2C module unsupported features:

- GPIO mode
- High-speed (HS) mode
- 10-bit device addressing mode

11.8.1.2 Industry Standard(s) Compliance Statement

The I2C module is compliant with the Philips Semiconductors Inter-IC bus (I^2C-bus) specification version 2.1.

11.8.2 I2C Environment

This section describes the I²C application fields from an environment point of view (external connections).

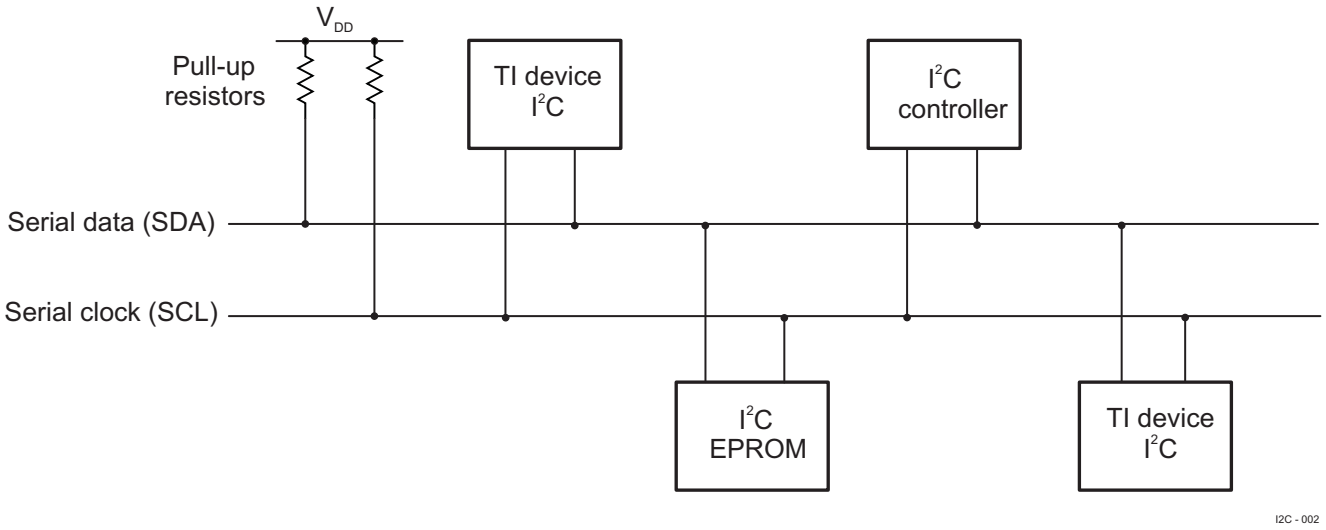
11.8.2.1 I2C Typical Application

The I²C bus is a multi-master bus. The I2C module supports the multi-master mode that allows more than one device capable of controlling the bus to be connected to it. Each I2C device is recognized by a unique address and can operate as either transmitter or receiver depending on the function of the device. In addition to being a transmitter or receiver, devices connected to the I²C bus also can be considered as master or slave when performing data transfers.

Note that a master device is the device that initiates a data transfer on the bus and generates the clock signals to permit that transfer. During this transfer, any device addressed by this master is considered a slave.

Figure 11-538 shows an example of multiple I2C modules connected for a two-way transfer from one device to other devices.

Figure 11-538. Multiple I2C Modules Connected



11.8.2.2 I2C Signal Descriptions

Data is communicated to devices interfacing with the I²C via the serial data line (SDA) and the serial clock line (SCL). These two pins carry information between the device and other devices connected to the I²C-bus. Both SDA and SCL are bi-directional pins. Each one must be connected to a positive supply voltage via a pull-up resistor. When the bus is free, both pins are high. The driver of these two pins has an open-drain configuration to perform the required wired-AND function.

Table 11-1176 lists the pins associated with the I²C interface.

Table 11-1176. I2C Input/Output Signals

Interface Name	Device Level Signal	I/O[(1)]	Description
I2C_0	I2C0_SCL	I/O	I²C serial clock line. Open-drain output buffer. Requires external pull-up resistor (Rp).
	I2C0_SDA	I/O	I²C serial data line. Open-drain output buffer. Requires external pull-up resistor (Rp).
I2C_1	I2C1_SCL	I/O	I²C serial clock line. Open-drain output buffer. Requires external pull-up resistor (Rp).
	I2C1_SDA	I/O	I²C serial data line. Open-drain output buffer. Requires external pull-up resistor (Rp).
I2C_2	I2C2_SCL	I/O	I²C serial clock line. Open-drain output buffer. Requires external pull-up resistor (Rp).
	I2C2_SDA	I/O	I²C serial data line. Open-drain output buffer. Requires external pull-up resistor (Rp).

(1) I = Input; O = Output; I/O = Bidirectional

For additional timing and electrical specifications for these pins, see the device Data Manual.

NOTE: One clock pulse is generated by the master device for each data bit transferred. Due to a variety of different technology devices that can be connected to the I^2C-bus, the levels of logic 0 (low) and logic 1 (high) are not fixed and depend on the associated power supply level. For details, see the device Data Manual.

11.8.3 I2C Integration

This section describes module integration in the device, including information about clocks, resets, and hardware requests.

Figure 11-539 shows the integration of the three I2C modules in the device.

Figure 11-539. I2C Integration

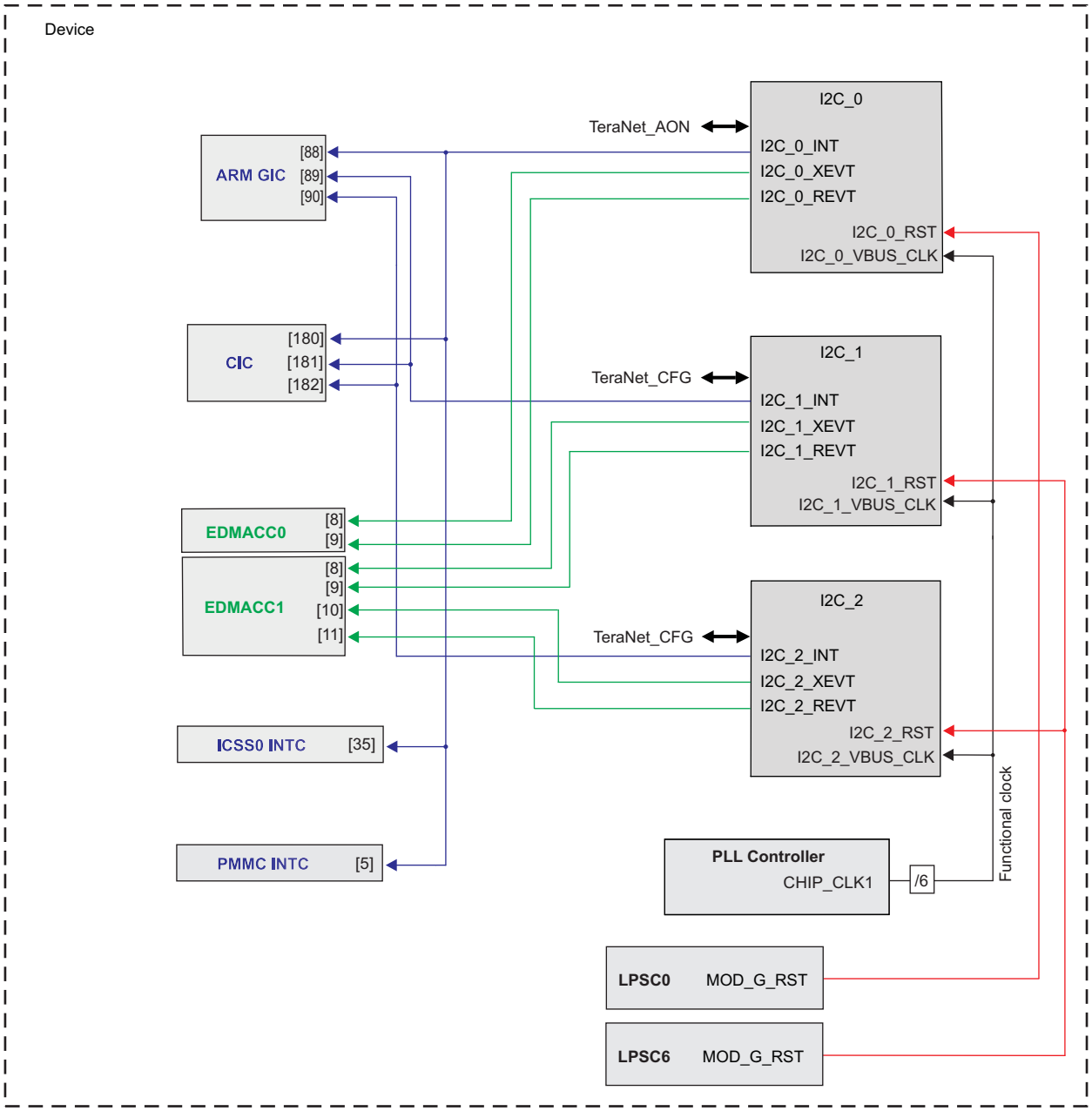


Table 11-1177 through Table 11-1179 summarize the integration of the module in the device.

Table 11-1177. I2C Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
I2C_0	PD0	LPSC0	TeraNet_AON
I2C_1	PD5	LPSC6	TeraNet_CFG
I2C_2	PD5	LPSC6	TeraNet_CFG

Table 11-1178. I2C Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
I2C_0	I2C_0_VBUS_CLK	CHIP_CLK1	PLL Controller	I2C_0 Interface and Functional clock
I2C_1	I2C_1_VBUS_CLK	CHIP_CLK1	PLL Controller	I2C_1 Interface and Functional clock
I2C_2	I2C_2_VBUS_CLK	CHIP_CLK1	PLL Controller	I2C_2 Interface and Functional clock
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
I2C_0	I2C_0_RST	MOD_G_RST	LPSC0	I2C_0 reset
I2C_1	I2C_1_RST	MOD_G_RST	LPSC6	I2C_1 reset
I2C_2	I2C_2_RST	MOD_G_RST	LPSC6	I2C_2 reset

Table 11-1179. I2C Hardware Requests

Interrupt Requests						
Module Instance	**Event Name**	**Mapped To Input Event [Number]**				**Description**
		ARM GIC	**CIC**	**ICSS0 INTC**	**PMMC INTC**	
I2C_0	I2C_0_INT	[88]	[180]	[35]	[5]	I2C_0 interrupt
I2C_1	I2C_1_INT	[89]	[181]	-	-	I2C_1 interrupt
I2C_2	I2C_2_INT	[90]	[182]	-	-	I2C_2 interrupt
DMA Requests						
Module Instance	**Event Name**	**Mapped To Input Event [Number]**				**Description**
		EDMACC0		**EDMACC1**		
I2C_0	I2C_0_XEVT	[8]		-		I2C_0 transmit event
	I2C_0_REVT	[9]		-		I2C_0 receive event
I2C_1	I2C_1_XEVT	-		[8]		I2C_1 transmit event
	I2C_1_REVT	-		[9]		I2C_1 receive event
I2C_2	I2C_2_XEVT	-		[10]		I2C_2 transmit event
	I2C_2_REVT	-		[11]		I2C_2 receive event

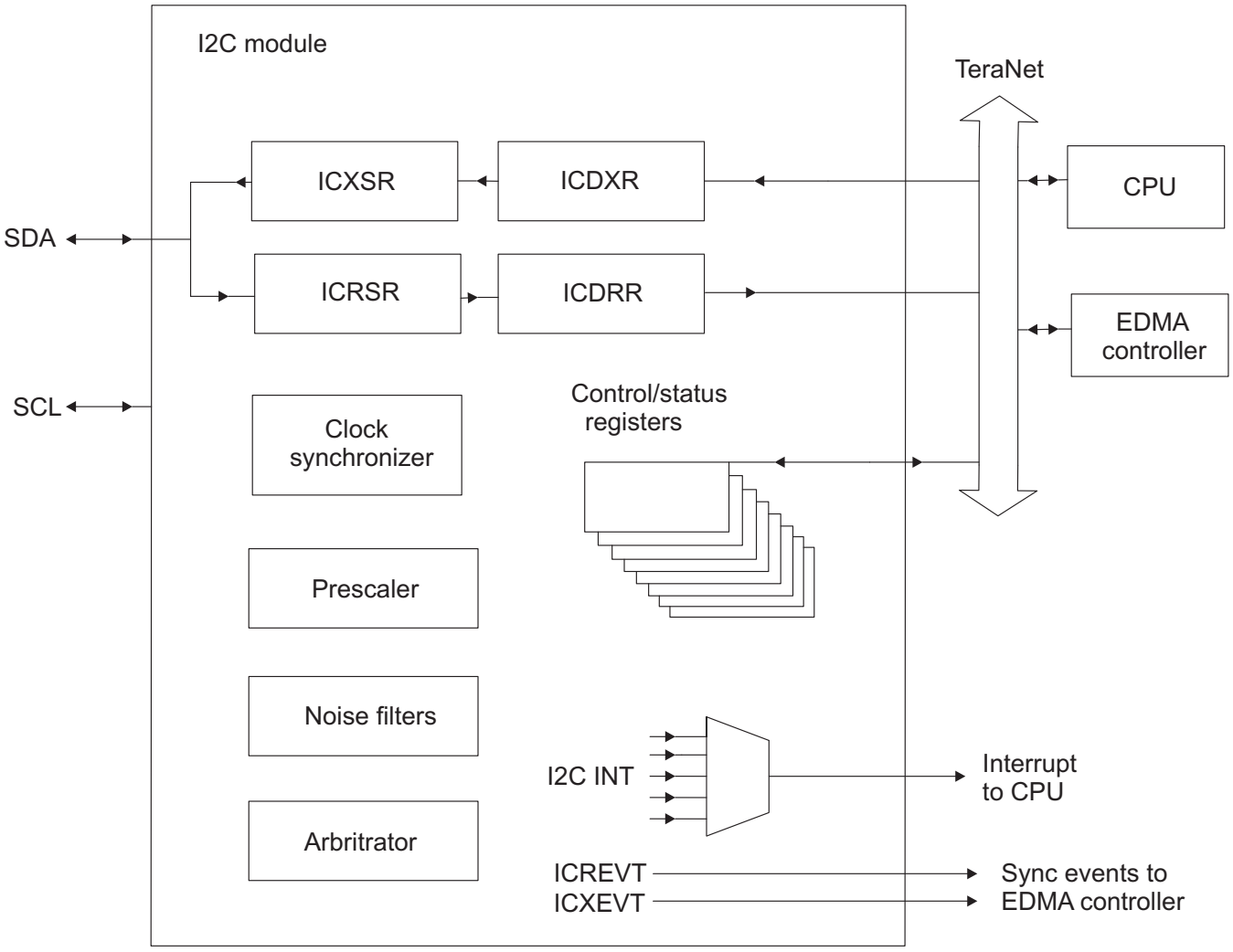
11.8.4 I2C Functional Description

NOTE: In this section, a single instance of I^2C is described for simplification as all three general-purpose I2C modules are functionally identical.

11.8.4.1 I2C Block Diagram

Figure 11-540 shows the I2C module block diagram.

Figure 11-540. I2C Block Diagram



The I2C module consists of the following primary blocks:

- A serial interface: one data pin (SDA) and one clock pin (SCL)
- Data registers to temporarily hold receive data and transmit data traveling between the SDA pin and the CPU or the EDMA controller
- Control and status registers
- A peripheral data bus interface to enable the CPU and the EDMA controller to access the I2C module registers
- A clock synchronizer to synchronize the I^2C input clock (from the clock generator) and the clock on the SCL pin, and to synchronize data transfers with masters of different clock speeds
- A prescaler to divide down the input clock that is driven to the I2C module

- A noise filter on each of the two pins, SDA and SCL
- An arbitrator to handle arbitration between the I2C module (when it is a master) and another master
- Interrupt generation logic, so that an interrupt can be sent to the CPU
- EDMA event generation logic, so that activity in the EDMA controller can be synchronized to data reception and data transmission in the I2C module

Figure 11-540 shows the four registers used for transmission and reception. The CPU or the EDMA controller writes data for transmission to I2C_ICDXR and reads received data from I2C_ICDRR. When the I2C module is configured as a transmitter, data written to I2C_ICDXR is copied to ICXSR and shifted out on the SDA pin one bit a time. When the I2C module is configured as a receiver, received data is shifted into ICRSR and then copied to I2C_ICDRR.

11.8.4.2 I2C Clock Generation

The I2C module is recommended to operate with a module clock in a range of 7 to 12 MHz. This clock is generated via the I^2C prescaler block. The I^2C prescaler register I2C_ICPSC is used to divide-down the input clock to obtain a clock within the specified range for the I2C module.

As shown in Figure 11-541, the PLL Controller produces an I^2C input clock with a programmed frequency. The clock is then divided twice more inside the I2C module to produce to the module clock and the master clock.

The module clock determines the frequency at which the I2C module operates. Figure 11-541 shows how this clock is generated. A programmable prescaler in the I2C module divides down the I^2C input clock to produce the module clock. To specify the divide-down value, initialize the IPSC field of the I^2C prescaler register I2C_ICPSC. The resulting frequency is:

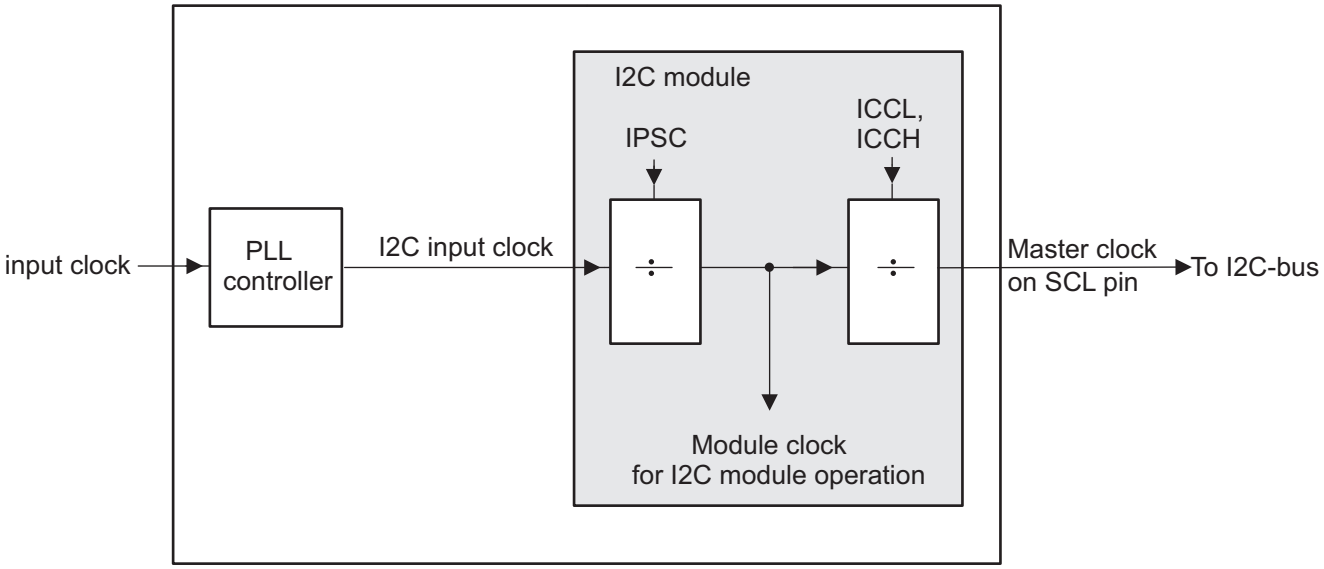
$$\text{module clock frequency} = \frac{I^2C \text{ input clock frequency}}{(\text{IPSC}+1)}$$

The prescaler must be initialized only while the I2C module is in the reset state (IRS = 0 in I2C_ICMDR). The prescaled frequency takes effect only when IRS is changed to 1. Changing the IPSC value while IRS = 1 has no effect.

The master clock appears on the SCL pin when the I2C module is configured to be a master on the I^2C-bus. This clock controls the timing of communication between the I2C module and a slave. As shown in Figure 11-541, a second clock divider in the I2C module divides down the module clock to produce the master clock. The clock divider uses the ICCL value of I2C_ICCLKL to divide-down the low portion of the module clock signal and uses the ICCH value of I2C_ICCLKH to divide-down the high portion of the module clock signal. The resulting frequency is:

$$\text{master clock frequency} = \frac{\text{module clock frequency}}{(\text{ICCL}+6)+(\text{ICCH}+6)}$$

Figure 11-541. Clocking Diagram for the I2C Module



11.8.4.3 I2C Reset

The I2C module has two reset sources:

- Hardware reset: it causes all I2C module registers to be reset to their default values.
- Software reset: it can be applied by writing 0 to the I²C reset (IRS) bit in the I²C mode register I2C_ICMDR. This bit has exactly the same action on the module logic as the hardware reset (all registers are reset to their default values).

In both cases, the I2C module remains disabled until the IRS bit is changed to 1. Upon reset the SDA and SCL pins are in the high-impedance (HiZ) state.

NOTE: The IRS bit must be cleared to 0 during I2C module configuration/reconfiguration. Forcing IRS to 0 can be used to save power and to clear error conditions.

NOTE: If the IRS bit is cleared to 0 during a transfer, this can cause the I²C bus to hang. For more information, see Section 11.8.4.8, *I²C Bus Hang Caused by Reset*.

11.8.4.4 I2C Interrupt Support

The I2C module is capable of interrupting a device CPU and sends a single interrupt signal to the CPU. The CPU can determine which I²C events caused the interrupt by reading the I²C interrupt vector register I2C_ICIVR. I2C_ICIVR contains a binary-coded interrupt vector type to indicate which interrupt has occurred. Reading I2C_ICIVR clears the interrupt flag; if other interrupts are pending, a new interrupt is generated. If there is more than one pending interrupt flag, reading I2C_ICIVR clears the highest-priority interrupt flag.

The I2C module can generate the interrupts described in Table 11-1180. Each interrupt has a flag bit in the I²C interrupt status register I2C_ICSTR and a mask bit in the interrupt mask register I2C_ICIMR. When one of the specified events occurs, its flag bit is set. If the corresponding mask bit is 0, the interrupt request is blocked; if the mask bit is 1, the request is forwarded to the CPU as an I²C interrupt.

Table 11-1180. Descriptions of the I2C Interrupt Events

I²C Interrupt	Initiating Event
Arbitration-lost interrupt (AL)	Generated when the I²C arbitration procedure is lost or illegal START/STOP conditions occur.
No-acknowledge interrupt (NACK)	Generated when the master I²C does not receive any acknowledge from the receiver.
Registers-ready-for-access (ARDY)	Generated by the I²C when the previously programmed address, data and command have been interrupt performed and the status bits have been updated. This interrupt is used to let the controlling processor know that the I²C registers are ready to be accessed.
Receive interrupt/status (ICRINT and ICRRDY)	Generated when the received data in the receive-shift register ICRSR has been copied into the I2C_ICDRR. The ICRRDY bit can also be polled by the CPU to read the received data in the I2C_ICDRR.
Transmit interrupt/status (ICXINT and ICXRDY)	Generated when the transmitted data has been copied from I2C_ICDXR to the transmit-shift register ICXSR and shifted out on the SDA pin. This bit can also polled by the CPU to write the next transmitted data into the I2C_ICDXR.
Stop-Condition-Detection interrupt (SCD)	Generated when a STOP condition has been detected.
Address-as-Slave interrupt (AAS)	Generated when the I²C has recognized its own slave address or an address of all (8) zeros.

11.8.4.5 I2C DMA Events

The I2C module generates two DMA events which can be used by a EDMA controller to handle transmit and receive data. Activity in EDMA channels can be synchronized to these events:

- Receive/Read event ICREVT: When receive data has been copied from the receive shift register ICRSR to the data receive register I2C_ICDRR, the I2C module sends an REVT signal to the EDMA controller. In response, the EDMA controller can read the data from I2C_ICDRR.
- Transmit/Write event ICXEVT: When transmit data has been copied from the data transmit register I2C_ICDXR to the transmit shift register ICXSR, the I2C module sends an XEVT signal to the EDMA controller. In response, the EDMA controller can write the next transmit data value to I2C_ICDXR.

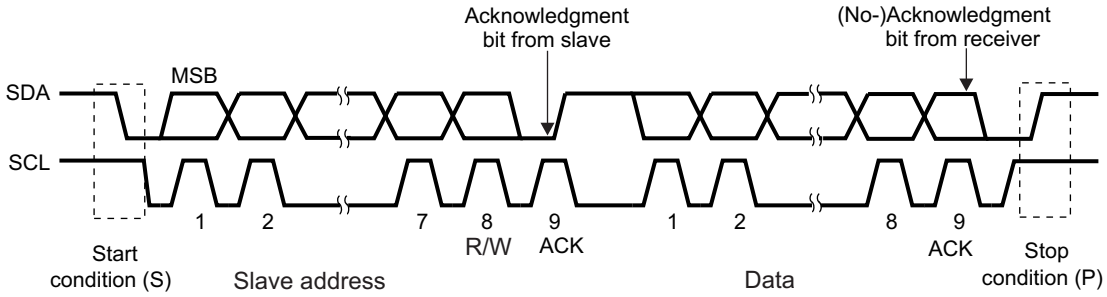
11.8.4.6 I2C Operation

11.8.4.6.1 I2C Serial Data Format

The I2C module supports a data word size of 1 bit to 8 bits depending on the bit count (BC) bits of I2C_ICMDR.

Figure 11-542 shows an example of a data transfer on the I²C-bus for an 8-bit data word (BC = 000 in I2C_ICMDR). Each bit put on the SDA line corresponds to one pulse on the SCL line and the data is always transferred with the most-significant bit (MSB) first. The number of data words that can be transmitted or received is unrestricted; however, the transmitters and receivers must agree on the number of data bits in a word being transferred.

Figure 11-542. I²C Data Transfer



The I2C module supports the following data formats:

- 7-bit addressing mode
- Free data format mode

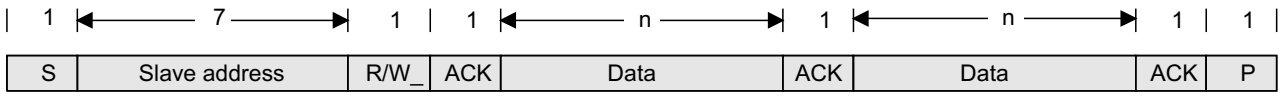
11.8.4.6.1.1 7-Bit Addressing Format

In the 7-bit addressing format Figure 11-543, the first byte after a START condition (S) consists of a 7-bit slave address followed by a R/$\overline{W}$ bit. The R/$\overline{W}$ bit determines the direction of the data:

- R/$\overline{W}$ = 0: The master writes (transmits) data to the addressed slave.
- R/$\overline{W}$ = 1: The master reads (receives) data from the slave.

An extra clock cycle dedicated for acknowledgment (ACK) is inserted after the R/$\overline{W}$ bit. If the ACK bit is inserted by the slave, it is followed by n bits of data from the transmitter (master or slave, depending on the R/$\overline{W}$ bit), n is a number from 1 to 8 determined by the bit count (BC) bits of I2C_ICMDR. After the data bits have been transferred, the receiver inserts an ACK bit. To select the 7-bit addressing format, write 0 to the expanded address enable (XA) bit of I2C_ICMDR.

Figure 11-543. I2C Module 7-Bit Addressing Format (FDF = 0, XA = 0 in I2C_ICMDR)



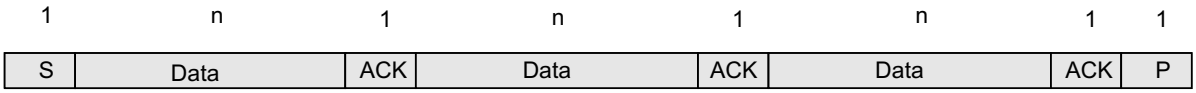
n = The number of data bits (from 1 to 8) specified by the bit count (BC) field of ICMDR.

11.8.4.6.1.2 Free Data Format

In the free data format Figure 11-544, the first bits after a START condition (S) are a data word. An ACK bit is inserted after each data word, which can be from 1 to 8 bits, depending on the bit count (BC) bits of I2C_ICMDR. No address or data-direction bit is sent. Therefore, the transmitter and the receiver must both support the free data format, and the direction of the data must be constant throughout the transfer.

To select the free data format, write 1 to the free data format (FDF) bit of I2C_ICMDR.

Figure 11-544. I2C Module Free Data Format (FDF = 1 in I2C_ICMDR)

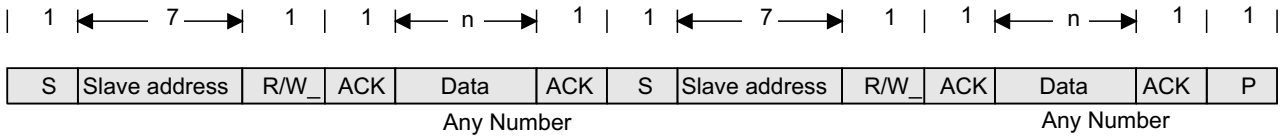


n = The number of data bits (from 1 to 8) specified by the bit count (BC) field of ICMDR.

11.8.4.6.1.3 Using a Repeated START Condition

The repeated START condition can be used with the 7-bit addressing and free data format. The 7-bit addressing format using a repeated START condition (S) is shown in Figure 11-545. At the end of each data word, the master can drive another START condition. Using this capability, a master can transmit/receive any number of data words before driving a STOP condition. The length of a data word can be from 1 to 8 bits and is selected with the bit count (BC) bits of I2C_ICMDR.

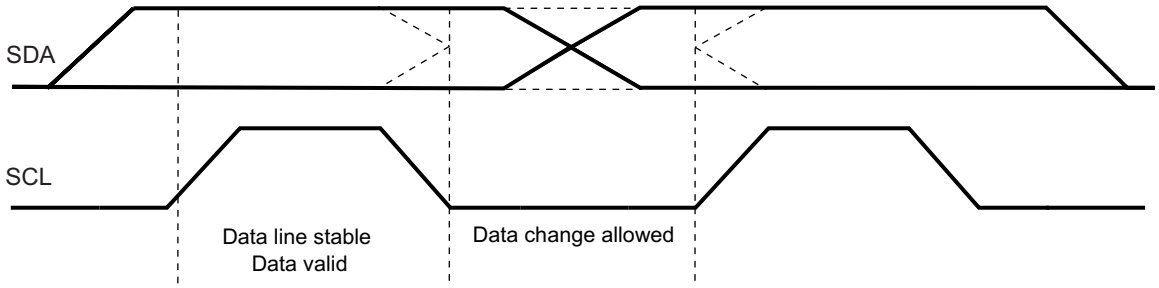
Figure 11-545. I2C Module 7-Bit Addressing Format With Repeated START Condition



11.8.4.6.2 I2C Data Validity

The data on the SDA line must be stable during the high period of the clock (see Figure 11-546). The high and low states of the data line can only change when the clock signal on the SCL line is low.

Figure 11-546. Bit Transfer on the I²C-Bus

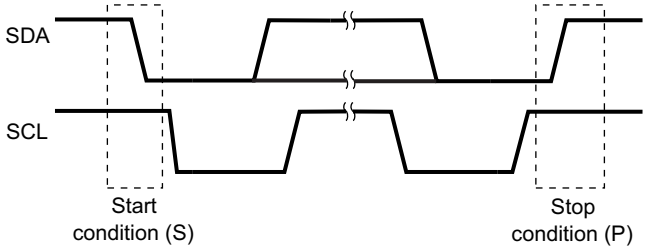


11.8.4.6.3 I2C START and STOP Conditions

The I2C module generates START and STOP conditions when it is configured as a master on the I²C-bus. As shown in Figure 11-547:

- The START condition is defined as a high-to-low transition on the SDA line while SCL is high. A master drives this condition to indicate the start of a data transfer.
- The STOP condition is defined as a low-to-high transition on the SDA line while SCL is high. A master drives this condition to indicate the end of a data transfer.

Figure 11-547. I²C START and STOP Condition Events



After a START condition and before a subsequent STOP condition, the I²C-bus is considered busy, and the bus busy (BB) bit of I2C_ICSTR is 1. Between a STOP condition and the next START condition, the bus is considered free, and BB is 0.

For the I2C module to start a data transfer with a START condition, the master mode (MST) bit and the START condition (STT) bit in I2C_ICMDR must both be set to 1. For the I2C module to end a data transfer with a STOP condition, the STOP condition (STP) bit must be set to 1. When BB is set to 1 and STT is set to 1, a repeated START condition is generated. For a description of the MST, STT, and STP bits, see Section 11.8.6.9.

11.8.4.6.4 I2C Operating Modes

The I2C module has four basic operating modes to support data transfers as a master and as a slave. For the names and descriptions of the modes, see Table 11-1181.

If the I2C module is a master, it begins as a master-transmitter and typically transmits an address for a particular slave. When giving data to the slave, the I2C module must remain a master-transmitter. In order to receive data from a slave, the I2C module must be changed to the master-receiver mode.

If the I2C module is a slave, it begins as a slave-receiver and, typically, sends acknowledgment when it recognizes its slave address from a master. If the master will be sending data to the I2C module, the module must remain a slave-receiver. If the master has requested data from the I2C module, the module must be changed to the slave-transmitter mode.

Table 11-1181. Operating Modes of the I2C Module

Operating Mode	Description
Slave-receiver mode	The I2C module is a slave and receives data from a master. All slave modules begin in this mode. In this mode, serial data bits received on SDA are shifted in with the clock pulses that are generated by the master. As a slave, the I2C module does not generate the clock signal, but it can hold SCL low while the intervention of the CPU is required (RSFULL = 1 in I2C_ICSTR) after data has been received.
Slave-transmitter mode	The I2C module is a slave and transmits data to a master. This mode can only be entered from the slave-receiver mode; the I2C module must first receive a command from the master. When any of the 7-bit/free data addressing format is being used, the I2C module enters its slave-transmitter mode if the slave address is the same as its own address (in I2C_ICOAR) and the master has transmitted R/$\overline{W}$ = 1. As a slave-transmitter, the I2C module then shifts the serial data out on SDA with the clock pulses that are generated by the master. While a slave, the I2C module does not generate the clock signal, but it can hold SCL low while the intervention of the CPU is required (XSMT = 0 in I2C_ICSTR) after data has been transmitted.
Master-receiver mode	The I2C module is a master and receives data from a slave. This mode can only be entered from the master-transmitter mode; the I2C module must first transmit a command to the slave. When any of the 7-bit/free data addressing formats is being used, the I2C module enters its master-receiver mode after transmitting the slave address and R/$\overline{W}$ = 1. Serial data bits on SDA are shifted into the I2C module with the clock pulses generated by the I2C module on SCL. The clock pulses are inhibited and SCL is held low when the intervention of the CPU is required (RSFULL = 1 in I2C_ICSTR) after data has been received.
Master-transmitter mode	The I2C module is a master and transmits control information and data to a slave. All master modules begin in this mode. In this mode, data assembled in any of the 7-bit/free data addressing formats is shifted out on SDA. The bit shifting is synchronized with the clock pulses generated by the I2C module on SCL. The clock pulses are inhibited and SCL is held low when the intervention of the CPU is required (XSMT = 0 in I2C_ICSTR) after data has been transmitted.

11.8.4.6.5 NACK Bit Generation

When the I2C module is a receiver (master or slave), it can acknowledge or ignore bits sent by the transmitter. To ignore any new bits, the I2C module must send a no-acknowledge (NACK) bit during the acknowledge cycle on the bus. Table 11-1182 summarizes the various ways the I2C module sends a NACK bit.

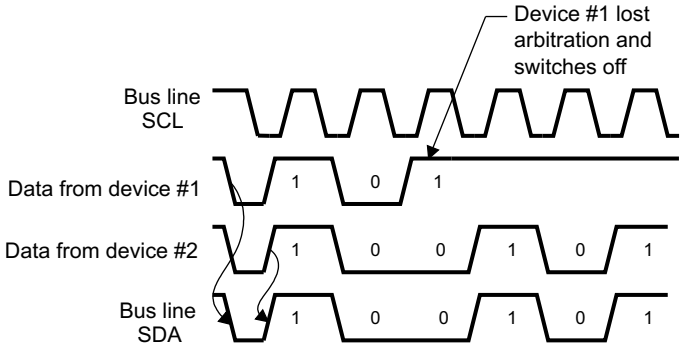
Table 11-1182. Generating a NACK Bit

	Description	
I2C module Condition	**Basic**	**Optional**
Slave-receiver mode	1. Disable data transfers (STT = 0 in I2C_ICSTR). 2. Allow an overrun condition (RSFULL = 1 in I2C_ICSTR). 3. Reset the module (IRS = 0 in I2C_ICMDR).	Before the rising edge of the last data bit intended to be received, set the NACKMOD bit of I2C_ICMDR.
Master-receiver mode AND Repeat mode (RM = 1 in I2C_ICMDR)	1. Generate a STOP condition (STOP = 1 in I2C_ICMDR). 2. Reset the module (IRS = 0 in I2C_ICMDR).	Before the rising edge of the last data bit intended to be received, set the NACKMOD bit of I2C_ICMDR.
Master-receiver mode AND Nonrepeat mode (RM = 0 in I2C_ICMDR)	1. If STP = 1 in I2C_ICMDR, allow the internal data counter to count down to 0 and force a STOP condition. 2. If STP = 0, make STP = 1 to generate a STOP condition. 3. Reset the module (IRS = 0 in I2C_ICMDR).	Before the rising edge of the last data bit intended to be received, set the NACKMOD bit of I2C_ICMDR.

11.8.4.6.6 I2C Bus Arbitration

If two or more master-transmitters simultaneously start a transmission on the same bus, an arbitration procedure is invoked. The arbitration procedure uses the data presented on the serial data bus (SDA) by the competing transmitters. Figure 11-548 illustrates the arbitration procedure between two devices.

Figure 11-548. Arbitration Procedure Between Two Master-Transmitters



The first master-transmitter, which drives SDA high, is overruled by another master-transmitter that drives SDA low. The arbitration procedure gives priority to the device that transmits the serial data stream with the lowest binary value. Should two or more devices send identical first bytes, arbitration continues on the subsequent bytes.

If the I2C module is the losing master, it switches to the slave-receiver mode, sets the arbitration lost (AL) flag, and generates the arbitration-lost interrupt.

If during a serial transfer the arbitration procedure is still in progress when a repeated START condition or a STOP condition is transmitted to SDA, the master-transmitters involved must send the repeated START condition or the STOP condition at the same position in the format frame. Arbitration is not allowed between:

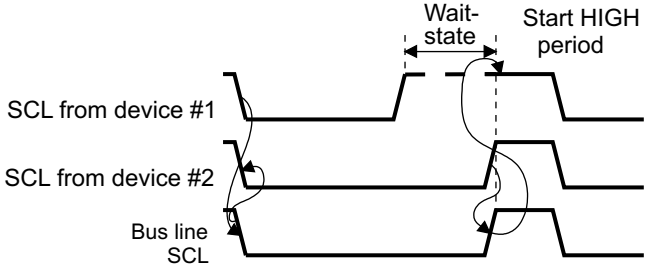
- A repeated START condition and a data bit
- A repeated START condition and a STOP condition
- A STOP condition and a data bit

11.8.4.6.7 I2C Clock Synchronization

Under normal conditions, only one master device generates the clock signal, SCL. During the arbitration procedure, however, there are two or more masters and the clock must be synchronized so that the data output can be compared. Figure 11-549 illustrates the clock synchronization. The wired-AND property of SCL means that a device that first generates a low period on SCL (device #1) overrules the other devices. At this high-to-low transition, the clock generators of the other devices are forced to start their own low period. The SCL is held low by the device with the longest low period. The other devices that finish their low periods must wait for SCL to be released, before starting their high periods. A synchronized signal on SCL is obtained, where the slowest device determines the length of the low period and the fastest device determines the length of the high period.

If a device pulls down the clock line for a longer time, the result is that all clock generators must enter the wait state. In this way, a slave slows down a fast master and the slow device creates enough time to store a received data word or to prepare a data word to be transmitted.

Figure 11-549. Synchronization of Two I²C Clock Generators During Arbitration



11.8.4.7 I2C Emulation Considerations

The response of the I^2C events to emulation suspend events (such as halts and breakpoints) is controlled by the FREE bit in the I^2C mode register I2C_ICMDR. The I2C module either stops exchanging data (FREE = 0) or continues to run (FREE = 1) when an emulation suspend event occurs. How the I^2C module terminates data transactions is affected by whether the I^2C module is acting as a master or a slave. For a description of the FREE bit see Section 11.8.6.9.

11.8.4.8 I2C Bus Hang Caused by Reset

It is generally known that the I^2C bus can hang if an I^2C master is removed from the bus in the middle of a data read. This can occur because the I^2C protocol does not mandate a minimum clock rate. Therefore, if a master is reset in the middle of a read while a slave is driving the data line low, the slave will continue driving the data line low while it waits for the next clock edge. This prevents bus masters from initiating transfers. If this condition is detected, the following three steps will clear the bus hang condition:

- An I^2C master must generate up to 9 clock cycles.
- After each clock cycle, the data pin must be observed to determine whether it has gone high while the clock is high.
- As soon as the data pin is observed high, the master can initiate a start condition.

Section 11.8.7 contains more information on this topic.

11.8.5 I2C Programming Guide

11.8.5.1 Configure the I2C Module Before Enabling the Controller

Before enabling the I²C controller, perform the following steps:

1. Enable the Functional clock (see Table 11-1178).
2. Program the prescaler to obtain an approximately 12-MHz internal sampling clock by programming the corresponding value in the I2C*i*.I2C_ICPSC[7-0] IPSC bit field. This value depends on the frequency of the functional clock (I2C*i*_FCLK).
3. Program the I²C*i*.I2C_ICCLKL[15-0] ICLL and I²C*i*.I2C_ICCLKH[15-0] ICCH bit fields to obtain a bit rate of 100 kbps, 400 kbps.

- Configure the Own Address of the I²C controller by storing it in the I2C*i*.I2C_ICOAR [6-0] OADDR bit field.
- Take the I²C controller out of reset by setting the I2C*i*.I2C_ICMDR [5] IRS bit to 1.

11.8.5.2 Initialize the I2C Controller

Configure the I2C*i*.I2C_ICMDR.

1. For master or slave mode, set the I2C*i*.I2C_ICMDR[10] MST bit (0: slave; 1: master).
2. For transmitter or receiver mode, set the I2C*i*.I2C_ICMDR[9] TRX bit (0: receiver; 1: transmitter).
3. If using an interrupt to transmit and receive data, set the corresponding bit in the I2C*i*.I2C_ICIMR register to 1 (the I2C*i*.I2C_ICIMR [4] ICXRDY bit for the transmit interrupt, the I2C*i*.I2C_ICIMR [3] ICRDRDY bit for the receive interrupt).
4. If using DMA to receive and transmit data, set the corresponding bit in the I2C*i*.I2C_ICDRR register to 1 (the I2C*i*.I2C_ICDRR [7] D bit for the receive DMA channel, the I2C*i*.I2C_ICDRR [7] D bit for the transmit DMA channel).

11.8.5.3 Configure Slave Address and the Data Control Register

In master mode, configure the slave address register by programming the I2C*i*.I2C_ICSAR [9-0] SADDR bit field and the number of data bytes (I2C data payload) associated with the transfer by programming the I2C*i*.I2C_ICCNT [15-0] ICDC bit field.

11.8.5.4 Initiate a Transfer

Poll the I2C*i*.I2C_ICSTR [12] BB bit. If it is cleared to 0 (bus not busy), configure the I2C*i*.I2C_ICMDR [13] STT and I2C*i*.I2C_ICMDR [11] STP bits. To initiate a transfer, the I2C*i*.I2C_ICMDR [13] STT bit must be set to 1, and it is not mandatory to set the I2C*i*.I2C_ICMDR [11] STP bit to 1.

11.8.6 I2C Registers

Table 11-1184 lists the memory-mapped registers for the I²C. All register offset addresses not listed in Table 11-1184 should be considered as reserved locations and the register contents should not be modified.

Table 11-1183. I²C Instances

Instance	Base Address
I2C_0	0253 0000h
I2C_1	0253 0400h
I2C_2	0253 0800h

Table 11-1184. I²C Registers

Offset	Acronym	Register Name	I2C_0 Physical Address	I2C_1 Physical Address	I2C_2 Physical Address	Section
0h	I2C_ICOAR	Own Address Register	0253 0000h	0253 0400h	0253 0800h	Section 11.8.6.1
4h	I2C_ICIMR	Interrupt Mask Register	0253 0004h	0253 0404h	0253 0804h	Section 11.8.6.2
8h	I2C_ICSTR	Interrupt Status Register	0253 0008h	0253 0408h	0253 0808h	Section 11.8.6.3
Ch	I2C_ICCLKL	Clock Low-time Divider Register	0253 000Ch	0253 040Ch	0253 080Ch	Section 11.8.6.4.1
10h	I2C_ICCLKH	Clock High-time Divider Register	0253 0010h	0253 0410h	0253 0810h	Section 11.8.6.4.2
14h	I2C_ICCNT	Data Count Register	0253 0014h	0253 0414h	0253 0814h	Section 11.8.6.5
18h	I2C_ICDRR	Data Receive Register	0253 0018h	0253 0418h	0253 0818h	Section 11.8.6.6
1Ch	I2C_ICSAR	Slave Address Register	0253 001Ch	0253 041Ch	0253 081Ch	Section 11.8.6.7
20h	I2C_ICDXR	Data Transmit Register	0253 0020h	0253 0420h	0253 0820h	Section 11.8.6.8
24h	I2C_ICMDR	Mode Register	0253 0024h	0253 0424h	0253 0824h	Section 11.8.6.9
28h	I2C_ICIVR	Interrupt Vector Register	0253 0028h	0253 0428h	0253 0828h	Section 11.8.6.10
2Ch	I2C_ICEMDR	Extended Mode Register	0253 002Ch	0253 042Ch	0253 082Ch	Section 11.8.6.11
30h	I2C_ICPSC	Prescaler Register	0253 0030h	0253 0430h	0253 0830h	Section 11.8.6.12
34h	I2C_ICPID1	Peripheral Identification Register 1	0253 0034h	0253 0434h	0253 0834h	Section 11.8.6.13.1
38h	I2C_ICPID2	Peripheral Identification Register 2	0253 0038h	0253 0438h	0253 0838h	Section 11.8.6.13.2

11.8.6.1 I2C_ICOAR Register (Offset = 0h) [reset = 0h]

The I^2C own address register I2C_ICOAR is used to specify its own slave address, which distinguishes it from other slaves connected to the I^2C-bus. If the 7-bit addressing mode is selected (XA = 0 in I2C_ICMDR), only bits 6-0 are used; bits 9-7 are ignored. The I2C_ICOAR register is shown in Figure 11-550 and described in Table 11-1186

Table 11-1185. I2C_ICOAR Instances

Instance	Physical Address
I2C_0	0253 0000h
I2C_1	0253 0400h
I2C_2	0253 0800h

Figure 11-550. I2C_ICOAR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						OADDR									
R-0h																						R/W-0h									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1186. I2C_ICOAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	These reserved bit location are always read as zeroes. A value written to this field has no effect.
9-0	OADDR	R/W	0h	Value = 0-3FFh Own slave address. Provides the slave address of the I^2C. In 7-bit addressing mode (XA = 0 in I2C_ICMDR): bits 6-0 provide the 7-bit slave address of the I^2C. Bits 9-7 are ignored. In 10-bit addressing mode (XA = 1 in I2C_ICMDR): bits 9-0 provide the 10-bit slave address of the I^2C.(1)

(1) 10-bit addressing mode is not supported feature on this device.

Table 11-1187. Register Call Summary for I2C_ICOAR

I2C Programming Guide • I2C Configure the Module Before Enabling the Controller: [0]
I2C Registers • I2C_ICOAR Register (Offset = 0h) [reset = 0h]: [0][1] • I2C Registers: [0] • I2C_ICMDR Register (Offset = 24h) [reset = 0h]: [0][1][2]
I2C Functional Description • I2C Operating Modes: [0]

11.8.6.2 I2C_ICIMR Register (Offset = 4h) [reset = 0h]

The I²C interrupt mask register I2C_ICIMR is used by the CPU to individually enable or disable I²C interrupt requests. The I2C_ICIMR register is shown in Figure 11-551 and described Table 11-1189.

Table 11-1188. I2C_ICIMR Instances

Instance	Physical Address
I2C_0	0253 0004h
I2C_1	0253 0404h
I2C_2	0253 0804h

Figure 11-551. I2C_ICIMR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	AAS	SCD	ICXRDY	ICRDRDY	ARDY	NACK	AL
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1189. I2C_ICIMR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	These reserved bit location are always read as zeroes. A value written to this field has no effect.
6	AAS	R/W	0h	Address-as-slave interrupt enable bit. • 0 = Interrupt request is disabled. • 1 = Interrupt request is enabled.
5	SCD	R/W	0h	Stop condition detected interrupt enable bit. • 0 = Interrupt request is disabled. • 1 = Interrupt request is enabled.
4	ICXRDY	R/W	0h	Transmit-data-ready interrupt enable bit. • 0 = Interrupt request is disabled. • 1 = Interrupt request is enabled.
3	ICRDRDY	R/W	0h	Receive-data-ready interrupt enable bit. • 0 = Interrupt request is disabled. • 1 = Interrupt request is enabled.
2	ARDY	R/W	0h	Register-access-ready interrupt enable bit. • 0 = Interrupt request is disabled. • 1 = Interrupt request is enabled.
1	NACK	R/W	0h	No-acknowledgment interrupt enable bit. • 0 = Interrupt request is disabled. • 1 = Interrupt request is enabled.

Table 11-1189. I2C_ICIMR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	AL	R/W	0h	Arbitration-lost interrupt enable bit. • 0 = Interrupt request is disabled. • 1 = Interrupt request is enabled.

Table 11-1190. Register Call Summary for I2C_ICIMR

I2C Programming Guide • Initialize the I2C Controller: [0][1][2]
I2C Registers • I2C_ICIMR Register (Offset = 4h) [reset = 0h]: [0][1] • I2C Registers: [0]
I2C Functional Description • I2C Interrupt Support: [0]

11.8.6.3 I2C_ICSTR Register (Offset = 8h) [reset = 0h]

The I²C interrupt status register I2C_ICSTR is used by the CPU to determine which interrupt has occurred and to read status information. The I2C_ICSTR register is shown in Figure 11-552 and described in Table 11-1192.

Table 11-1191. I2C_ICSTR Instances

Instance	Physical Address
I2C_0	0253 0008h
I2C_1	0253 0408h
I2C_2	0253 0808h

Figure 11-552. I2C_ICSTR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	SDIR	NACKSNT	BB	RSFULL	XSMT_	AAS	AD0
R-0h	R/W1toCl-0h	R/W1toCl-0h	R/W1toCl-0h	R-0h	R/W-1h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED		SCD	ICXRDY	ICRDRDY	ARDY	NACK	AL
R-0h		R/W1toCl-0h	R/W1toCl-0h	R/W1toCl-0h	R/W1toCl-0h	R/W1toCl-0h	R/W1toCl-0h

LEGEND: R = Read Only; R/W = Read/Write; R/W1toCl = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-1192. I2C_ICSTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	These reserved bit location are always read as zeroes. A value written to this field has no effect.
14	SDIR	R/W1toCl	0h	Slave direction bit. In digital-loopback mode (DLB), the SDIR bit is cleared to 0. • 0 = I²C is acting as a master-transmitter/receiver or a slave-receiver. SDIR is cleared by one of the following events: – A STOP or a START condition. – SDIR is manually cleared. To clear this bit, write a 1 to it. • 1 = I²C is acting as a slave-transmitter.
13	NACKSNT	R/W1toCl	0h	No-acknowledgment sent bit. NACKSNT bit is used when the I²C is in the receiver mode. One instance in which NACKSNT is affected is when the NACK mode is used (see the description for NACKMOD in Section 11.8.6.9). • 0 = NACK is not sent. NACKSNT is cleared by one of the following events: – It is manually cleared. To clear this bit, write a 1 to it. – The I²C is reset (either when 0 is written to the IRS bit of I2C_ICMDR or when the device is reset). • 1 = NACK is sent. A no-acknowledge bit was sent during the acknowledge cycle on the I²C-bus.

Table 11-1192. I2C_ICSTR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	BB	R/W1toCl	0h	Bus busy bit. BB bit indicates whether the I²C-bus is busy or is free for another data transfer. In the master mode, BB is controlled by the software. • 0 = Bus is free. BB is cleared by one of the following events: – The I²C receives or transmits a STOP bit (bus free). – BB is manually cleared. To clear this bit, write a 1 to it. – The I²C is reset (either when 0 is written to the IRS bit of I2C_ICMDR or when the device is reset). • 1 = Bus is busy. When the STT bit in I2C_ICMDR is set to 1, a restart condition is generated. BB is set by one of the following events: – The I²C has received or transmitted a START bit on the bus. – SCL is in a low state and the IRS bit in I2C_ICMDR is 0.
11	RSFULL	R	0h	Receive shift register full bit. RSFULL indicates an overrun condition during reception. Overrun occurs when the receive shift register ICRSR is full with new data but the previous data has not been read from the data receive register I2C_ICDRR. The new data will not be copied to I2C_ICDRR until the previous data is read. As new bits arrive from the SDA pin, they overwrite the bits in ICRSR. • 0 = No overrun is detected. RSFULL is cleared by one of the following events: – I2C_ICDRR is read. – The I²C is reset (either when 0 is written to the IRS bit of I2C_ICMDR or when the device is reset). • 1 = Overrun is detected.
10	XSMT	R/W	1h	Underflow occurs when the transmit shift register ICXSR is empty but the data transmit register I2C_ICDXR has not been loaded since the last I2C_ICDXR-to-ICXSR transfer. The next I2C_ICDXR-to-ICXSR transfer will not occur until new data is in I2C_ICDXR. If new data is not transferred in time, the previous data may be re-transmitted on the SDA pin. • 0 = Underflow is detected. • 1 = No underflow is detected. XSMT is set by one of the following events: – Data is written to I2C_ICDXR. – The I²C is reset (either when 0 is written to the IRS bit of I2C_ICMDR or when the device is reset).
9	AAS	R	0h	Addressed-as-slave bit. • 0 = The AAS bit has been cleared by a repeated START condition or by a STOP condition. • 1 = AAS is set by one of the following events: – I²C has recognized its own slave address or an address of all zeros (general call). – The first data word has been received in the free data format (FDF = 1 in I2C_ICMDR).
8	AD0	R	0h	Address 0 bit. • 0 = AD0 has been cleared by a START or STOP condition. • 1 = An address of all zeros (general call) is detected.
7-6	RESERVED	R	0h	These reserved bit location are always read as zeroes. A value written to this field has no effect.

Table 11-1192. I2C_ICSTR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	SCD	R/W1toCl	0h	Stop condition detected bit. SCD indicates when a STOP condition has been detected on the I ^{2}C bus. The STOP condition could be generated by the I^{2}C or by another I^{2}C device connected to the bus. • 0 = No STOP condition has been detected. SCD is cleared by one of the following events: – By reading the INCODE bits in ICICR as 110b. – SCD is manually cleared. To clear this bit, write a 1 to it. • 1 = A STOP condition has been detected.
4	ICXRDY	R/W1toCl	0h	Transmit-data-ready interrupt flag bit. ICXRDY indicates that the data transmit register I2C_ICDXR is ready to accept new data because the previous data has been copied from I2C_ICDXR to the transmit shift register ICXSR. The CPU can poll ICXRDY or use the XRDY interrupt request. • 0 = I2C_ICDXR is not ready. ICXRDY is cleared by one of the following events: – Data is written to I2C_ICDXR. – ICXRDY is manually cleared. To clear this bit, write a 1 to it. • 1 I= CDXR is ready. Data has been copied from I2C_ICDXR to ICXSR. ICXRDY is forced to 1 when the I^{2}C is reset.
3	ICRDRDY	R/W1toCl	0h	Receive-data-ready interrupt flag bit. ICRDRDY indicates that the data receive register I2C_ICDRR is ready to be read because data has been copied from the receive shift register ICRSR to I2C_ICDRR. The CPU can poll ICRDRDY or use the RRDY interrupt request. • 0 = I2C_ICDRR is not ready. ICRDRDY is cleared by one of the following events: – I2C_ICDRR is read. – ICRDRDY is manually cleared. To clear this bit, write a 1 to it. – The I^{2}C is reset (either when 0 is written to the IRS bit of I2C_ICMDR or when the device is reset). • 1 = I2C_ICDRR is ready. Data has been copied from ICRSR to I2C_ICDRR.
2	ARDY	R/W1toCl	0h	Register-access-ready interrupt flag bit (only applicable when the I^{2}C is in the master mode). ARDY indicates that the I^{2}C registers are ready to be accessed because the previously programmed address, data, and command values have been used. The CPU can poll ARDY or use the ARDY interrupt request. • 0 = The registers are not ready to be accessed. ARDY is cleared by one of the following events: – The I^{2}C starts using the current register contents. – ARDY is manually cleared. To clear this bit, write a 1 to it. – The I^{2}C is reset (either when 0 is written to the IRS bit of I2C_ICMDR or when the device is reset). • 1 = The registers are ready to be accessed. • In the nonrepeat mode (RM = 0 in I2C_ICMDR): If STP = 0 in I2C_ICMDR, ARDY is set when the internal data counter counts down to 0. If STP = 1, ARDY is not affected (instead, the I^{2}C generates a STOP condition when the counter reaches 0). • In the repeat mode (RM = 1): ARDY is set at the end of each data word transmitted from I2C_ICDXR.

Table 11-1192. I2C_ICSTR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	NACK	R/W1toCl	0h	No-acknowledgment interrupt flag bit. NACK applies when the I^2C is a transmitter (master or slave). NACK indicates whether the I^2C has detected an acknowledge bit (ACK) or a no-acknowledge bit (NACK) from the receiver. The CPU can poll NACK or use the NACK interrupt request. • 0 = ACK received/NACK is not received. NACK is cleared by one of the following events: – An acknowledge bit (ACK) has been sent by the receiver. – NACK is manually cleared. To clear this bit, write a 1 to it. – The CPU reads the interrupt source register ICISR when the register contains the code for a NACK interrupt. – The I^2C is reset (either when 0 is written to the IRS bit of I2C_ICMDR or when the device is reset). • 1= NACK bit is received. The hardware detects that a no-acknowledge (NACK) bit has been received. Note: While the I^2C performs a general call transfer, NACK is 1, even if one or more slaves send acknowledgment.
0	AL	R/W1toCl	0h	Arbitration-lost interrupt flag bit (only applicable when the I^2C is a master-transmitter). AL primarily indicates when the I^2C has lost an arbitration contest with another master-transmitter. The CPU can poll AL or use the AL interrupt request. • 0 = Arbitration is not lost. AL is cleared by one of the following events: • AL is manually cleared. To clear this bit, write a 1 to it. • The CPU reads the interrupt source register ICISR when the register contains the code for an AL interrupt. • The I^2C is reset (either when 0 is written to the IRS bit of I2C_ICMDR or when the device is reset). • 1 = Arbitration is lost. AL is set by one of the following events: • The I^2C senses that it has lost an arbitration with two or more competing transmitters that started a transmission almost simultaneously. • The I^2C attempts to start a transfer while the BB (bus busy) bit is set to 1. When AL is set to 1, the MST and STP bits of I2C_ICMDR are cleared, and the I^2C becomes a slave-receiver.

Table 11-1193. Register Call Summary for I2C_ICSTR

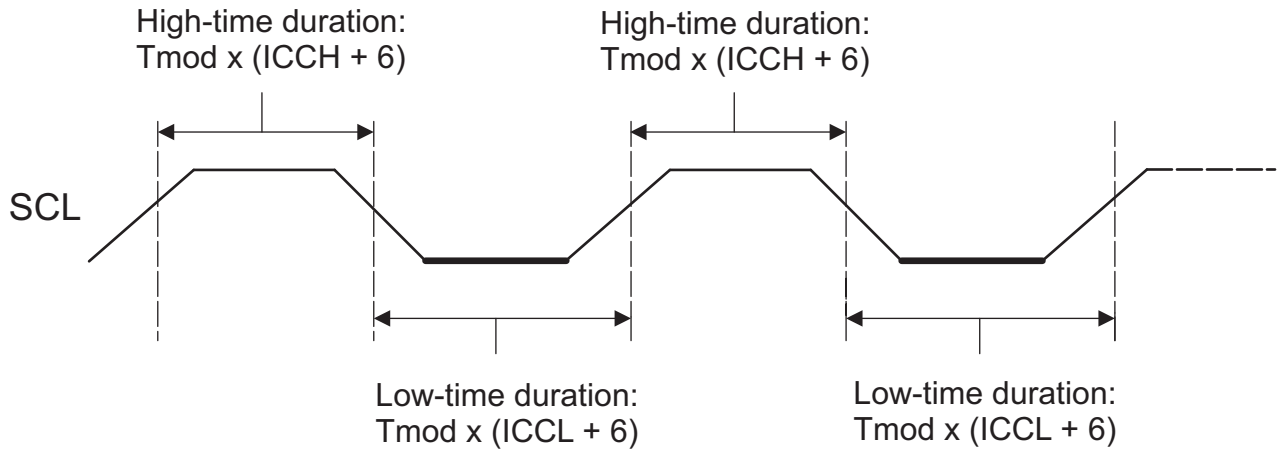
I2C Programming Guide • Initiate a Transfer: [0]
I2C Registers • I2C_ICSTR Register (Offset = 8h) [reset = 0h]: [0][1] • I2C Registers: [0] • I2C_ICMDR Register (Offset = 24h) [reset = 0h]: [0]
I2C Functional Description • NACK Bit Generation: [0][1] • I2C Interrupt Support: [0] • I2C START and STOP Conditions: [0] • I2C Operating Modes: [0][1][2][3]

11.8.6.4 I²C Clock Divider Registers (I2C_ICCLKL and I2C_ICCLKH)

When the I²C is a master, the module clock is divided down for use as the master clock on the SCL pin. As shown in Figure 11-553, the shape of the master clock depends on two divide-down values, ICCL and ICCH.The frequency of the master clock can be calculated as:

$$\text{master clock frequency} = \frac{\text{module clock frequency}}{(\text{ICCL}+6)+(\text{ICCH}+6)}$$

Figure 11-553. Roles of the Clock Divide-Down Values (ICCL and ICCH)



Tmod = module clock period = 1/ module clock frequency

11.8.6.4.1 I2C_ICCLKL Register (Offset = Ch) [reset = 0h]

The I²C clock low-time divider register I2C_ICCLKL is shown in Figure 11-554 and described in Table 11-1195. For each master clock cycle, ICCL determines the amount of time the signal is low. I2C_ICCLKL must be configured while the I²C is still in reset (IRS = 0 in I2C_ICMDR).

Table 11-1194. I2C_ICCLKL Instances

Instance	Physical Address
I2C_0	0253 000Ch
I2C_1	0253 040Ch
I2C_2	0253 080Ch

Figure 11-554. I2C_ICCLKL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ICCL															
R-0h																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1195. I2C_ICCLKL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Value = 0 These reserved bit location are always read as zeroes. A value written to this field has no effect.
15-0	ICCL	R/W	0h	Value = 0-FFFFh Clock low-time divide-down value of 1-65536. The period of the module clock is multiplied by (ICCL + 6) to produce the low-time duration of the master clock on the SCL pin.

Table 11-1196. Register Call Summary for I2C_ICCLKL

I2C Programming Guide • I2C Configure the Module Before Enabling the Controller: [0]
I2C Registers • I2C Registers: [0]
I2C Clock Divider Registers (I2C_ICCLKL and I2C_ICCLKH) • I2C_ICCLKL Register (Offset = Ch) [reset = 0h]: [0][1]
I2C Functional Description • I2C Clock Generation: [0]

11.8.6.4.2 I2C_ICCLKH Register (Offset = 10h) [reset = 0h]

The I²C clock high-time divider register I2C_ICCLKH is shown in Figure 11-555 and described in Table 11-1198. For each master clock cycle, ICCH determines the amount of time the signal is high. I2C_ICCLKH must be configured while the I²C is still in reset (IRS = 0 in I2C_ICMDR).

Table 11-1197. I2C_ICCLKH Instances

Instance	Physical Address
I2C_0	0253 0010h
I2C_1	0253 0410h
I2C_2	0253 0810h

Figure 11-555. I2C_ICCLKH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ICCH															
R-0h																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1198. I2C_ICCLKH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Value = 0 These reserved bit location are always read as zeroes. A value written to this field has no effect.
15-0	ICCH	R/W	0h	Value = 0-FFFFh Clock high-time divide-down value of 1-65536. The period of the module clock is multiplied by (ICCH + 6) to produce the high-time duration of the master clock on the SCL pin.

Table 11-1199. Register Call Summary for I2C_ICCLKH

I2C Programming Guide • I2C Configure the Module Before Enabling the Controller: [0]
I2C Registers • I2C Registers: [0]
I2C Clock Divider Registers (I2C_ICCLKL and I2C_ICCLKH) • I2C_ICCLKH Register (Offset = 10h) [reset = 0h]: [0][1]
I2C Functional Description • I2C Clock Generation: [0]

11.8.6.5 I2C_ICCNT Register (Offset = 14h) [reset = 0h]

The I^2C data count register I2C_ICCNT is used to indicate how many data words to transfer when the I^2C is configured as a master-transmitter (MST = 1 and TRX = 1 in I2C_ICMDR) and the repeat mode is off (RM = 0 in I2C_ICMDR). In the repeat mode (RM = 1), I2C_ICCNT is not used. The I2C_ICCNT register is shown in Figure 11-556 and described in Table 11-1201.

The value written to I2C_ICCNT is copied to an internal data counter. The internal data counter is decremented by 1 for each data word transferred (I2C_ICCNT remains unchanged). If a STOP condition is requested (STP = 1 in I2C_ICMDR), the I^2C terminates the transfer with a STOP condition when the countdown is complete (that is, when the last data word has been transferred).

Table 11-1200. I2C_ICCNT Instances

Instance	Physical Address
I2C_0	0253 0014h
I2C_1	0253 0414h
I2C_2	0253 0814h

Figure 11-556. I2C_ICCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ICDC															
R-0h																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1201. I2C_ICCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Value = 0 These reserved bit location are always read as zeroes. A value written to this field has no effect.
15-0	ICDC	R/W	0h	Value = 0-FFFFh Data count value. When RM = 0 in I2C_ICMDR, ICDC indicates the number of data words to transfer in the nonrepeat mode. When RM = 1 in I2C_ICMDR, the value in I2C_ICCNT is a don't care. If STP = 1 in I2C_ICMDR, a STOP condition is generated when the internal data counter counts down to 0. • 0 = The start value loaded to the internal data counter is 65536. • 1h-FFFFh = The start value loaded to internal data counter is 1-65535.

Table 11-1202. Register Call Summary for I2C_ICCNT

I2C Programming Guide • Configure Slave Address and the Data Control Register: [0]
I2C Registers • I2C_ICCNT Register (Offset = 14h) [reset = 0h]: [0][1][2][3][4][5] • I2C Registers: [0] • I2C_ICMDR Register (Offset = 24h) [reset = 0h]: [0][1][2][3]

11.8.6.6 I2C_ICDRR Register (Offset = 18h) [reset = 0h]

The I²C data receive register I2C_ICDRR is used by the DSP to read the receive data. The I2C_ICDRR can receive a data value of up to 8 bits; data values with fewer than 8 bits are right-aligned in the D bits and the remaining D bits are undefined. The number of data bits is selected by the bit count bits (BC) of I2C_ICMDR. The I²C receive shift register ICRSR shifts in the received data from the SDA pin. Once data is complete, the I²C copies the contents of ICRSR into I2C_ICDRR. The CPU and the EDMA controller cannot access ICRSR. The I2C_ICDRR register is shown in Figure 11-557 and described in Table 11-1204.

Table 11-1203. I2C_ICDRR Instances

Instance	Physical Address
I2C_0	0253 0018h
I2C_1	0253 0418h
I2C_2	0253 0818h

Figure 11-557. I2C_ICDRR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								D							
R-0h																								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1204. I2C_ICDRR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Value = 0 These reserved bit location are always read as zeroes. A value written to this field has no effect.
7-0	D	R	0h	Value = 0-FFh Receive data.

Table 11-1205. Register Call Summary for I2C_ICDRR

I2C Programming Guide • Initialize the I2C Controller: [0][1][2]
I2C Registers • I2C_ICDRR Register (Offset = 18h) [reset = 0h]: [0][1][2][3] • I2C_ICSTR Register (Offset = 8h) [reset = 0h]: [0][1][2][3][4][5][6][7][8] • I2C Registers: [0] • I2C_ICMDR Register (Offset = 24h) [reset = 0h]: [0][1]
I2C Functional Description • I2C DMA Events: [0][1] • I2C Interrupt Support: [0][1] • I2C Block Diagram: [0][1]

11.8.6.7 I2C_ICSAR Register (Offset = 1Ch) [reset = 0h]

The I^2C slave address register I2C_ICSAR contains a 7-bit slave address. When the I^2C is not using the free data format (FDF = 0 in I2C_ICMDR), it uses this address to initiate data transfers with a slave or slaves. When the address is non-zero, the address is for a particular slave. When the address is 0, the address is a general call to all slaves. If the 7-bit addressing mode is selected (XA = 0 in I2C_ICMDR), only bits 6-0 of I2C_ICSAR are used; bits 9-7 are ignored. The I2C_ICSAR register is shown in Figure 11-558 and described in Table 11-1207.

Table 11-1206. I2C_ICSAR Instances

Instance	Physical Address
I2C_0	0253 001Ch
I2C_1	0253 041Ch
I2C_2	0253 081Ch

Figure 11-558. I2C_ICSAR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						SADDR									
R-0h																						R/W-3FFh									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1207. I2C_ICSAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Value = 0 These reserved bit location are always read as zeroes. A value written to this field has no effect.
9-0	SADDR	R/W	3FFh	Value = 0-3FFh Slave address. Provides the slave address of the I^2C. In 7-bit addressing mode (XA = 0 in I2C_ICMDR): bits 6-0 provide the 7-bit slave address that the I^2C transmits when it is in the master-transmitter mode. Bits 9-7 are ignored. In 10-bit addressing mode (XA = 1 in I2C_ICMDR): Bits 9-0 provide the 10-bit slave address that the I^2C transmits when it is in the master-transmitter mode.(1)

(1) 10-bit addressing mode is not supported feature on this device.

Table 11-1208. Register Call Summary for I2C_ICSAR

I2C Programming Guide • Configure Slave Address and the Data Control Register: [0]
I2C Registers • I2C_ICSAR Register (Offset = 1Ch) [reset = 0h]: [0][1][2] • I2C Registers: [0] • I2C_ICMDR Register (Offset = 24h) [reset = 0h]: [0][1][2]

11.8.6.8 I2C_ICDXR Register (Offset = 20h) [reset = 0h]

The CPU writes transmit data to the I²C data transmit register I2C_ICDXR. The I2C_ICDXR can accept a data value of up to 8 bits. When writing a data value with fewer than 8 bits, the CPU must make sure that the value is right-aligned in the D bits. The number of data bits is selected by the bit count bits (BC) of I2C_ICMDR. Once data is written to I2C_ICDXR, the I²C copies the contents of I2C_ICDXR into the I²C transmit shift register ICXSR. The ICXSR shifts out the transmit data from the SDA pin. The CPU and the EDMA controller cannot access ICXSR. The I2C_ICDXR register is shown in Figure 11-559 and described in Table 11-1210.

Table 11-1209. I2C_ICDXR Instances

Instance	Physical Address
I2C_0	0253 0020h
I2C_1	0253 0420h
I2C_2	0253 0820h

Figure 11-559. I2C_ICDXR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								D							
R-0h																								R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1210. I2C_ICDXR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Value = 0 These reserved bit location are always read as zeroes. A value written to this field has no effect.
7-0	D	R/W	0h	Value = 0-FFh Transmit data.

Table 11-1211. Register Call Summary for I2C_ICDXR

I2C Registers • I2C_ICSTR Register (Offset = 8h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10] • I2C_ICDXR Register (Offset = 20h) [reset = 0h]: [0][1][2][3][4] • I2C_ICEMDR Register (Offset = 2Ch) [reset = 0h]: [0] • I2C Registers: [0] • I2C_ICMDR Register (Offset = 24h) [reset = 0h]: [0][1]
I2C Functional Description • I2C DMA Events: [0][1] • I2C Interrupt Support: [0][1] • I2C Block Diagram: [0][1]

11.8.6.9 I2C_ICMDR Register (Offset = 24h) [reset = 0h]

The I²C mode register I2C_ICMDR contains the control bits of the I²C. The I2C_ICMDR register is shown in Figure 11-560 and described in Table 11-1213.

Table 11-1212. I2C_ICMDR Instances

Instance	Physical Address
I2C_0	0253 0024h
I2C_1	0253 0424h
I2C_2	0253 0824h

Figure 11-560. I2C_ICMDR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
NACKMOD	FREE	STT	RES	STP	MST	TRX	XA
R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RM	DLB	IRS	STB	FDF	BC		
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1213. I2C_ICMDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	These reserved bit location are always read as zeroes. A value written to this field has no effect.
15	NACKMOD	R/W	0h	No-acknowledge (NACK) mode bit (only applicable when the I²C is a receiver). • 0 = In slave-receiver mode: The I²C sends an acknowledge (ACK) bit to the transmitter during the each acknowledge cycle on the bus. The I²C only sends a no-acknowledge (NACK) bit if you set the NACKMOD bit. In master-receiver mode: The I²C sends an ACK bit during each acknowledge cycle until the internal data counter counts down to 0. When the counter reaches 0, the I²C sends a NACK bit to the transmitter. To have a NACK bit sent earlier, you must set the NACKMOD bit. • 1 = In either slave-receiver or master-receiver mode: The I²C sends a NACK bit to the transmitter during the next acknowledge cycle on the bus. Once the NACK bit has been sent, NACKMOD is cleared. To send a NACK bit in the next acknowledge cycle, you must set NACKMOD before the rising edge of the last data bit.

Table 11-1213. I2C_ICMDR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	FREE	R/W	0h	This emulation mode bit is used to determine the state of the I²C when a breakpoint is encountered in the high-level language debugger. • 0 = When I²C is master: If SCL is low when the breakpoint occurs, the I²C stops immediately and keeps driving SCL low, whether the I²C is the transmitter or the receiver. If SCL is high, the I²C waits until SCL becomes low and then stops. When I²C is slave: A breakpoint forces the I²C to stop when the current transmission/reception is complete. • 1 = The I²C runs free; that is, it continues to operate when a breakpoint occurs.
13	STT	R/W	0h	START condition bit (only applicable when the I²C is a master). The RM, STT, and STP bits determine when the I²C starts and stops data transmissions (see Table 11-1214). Note that the STT and STP bits can be used to terminate the repeat mode. • 0 = In master mode, STT is automatically cleared after the START condition has been generated. In slave mode, if STT is 0, the I²C does not monitor the bus for commands from a master. As a result, the I²C performs no data transfers. • 1 = In master mode, setting STT to 1 causes the I²C to generate a START condition on the I²C-bus. In slave mode, if STT is 1, the I²C monitors the bus and transmits/receives data in response to commands from a master.
12	RESERVED	R	0h	These reserved bit location are always read as zeroes. A value written to this field has no effect.
11	STP	R/W	0h	STOP condition bit (only applicable when the I²C is a master). The RM, STT, and STP bits determine when the I²C starts and stops data transmissions (see Table 11-1214). Note that the STT and STP bits can be used to terminate the repeat mode. • 0 = STP is automatically cleared after the STOP condition has been generated. • 1 = STP has been set by the DSP to generate a STOP condition when the internal data counter of the I²C counts down to 0.
10	MST	R/W	0h	Master mode bit. MST determines whether the I²C is in the slave mode or the master mode. MST is automatically changed from 1 to 0 when the I²C master generates a STOP condition (see Table 11-1215). • 0 = Slave mode. The I²C is a slave and receives the serial clock from the master. • 1 = Master mode. The I²C is a master and generates the serial clock on the SCL pin.
9	TRX	R/W	0h	Transmitter mode bit. When relevant, TRX selects whether the I²C is in the transmitter mode or the receiver mode. Table 11-1215 summarizes when TRX is used and when it is a don't care. • 0 = Receiver mode. The I²C is a receiver and receives data on the SDA pin. • 1 = Transmitter mode. The I²C is a transmitter and transmits data on the SDA pin.
8	XA	R/W	0h	Expanded address enable bit. • 0 = 7-bit addressing mode (normal address mode). The I²C transmits 7-bit slave addresses (from bits 6-0 of I2C_ICSAR), and its own slave address has 7 bits (bits 6-0 of I2C_ICOAR). • 1 = 10-bit addressing mode (expanded address mode). The I²C transmits 10-bit slave addresses (from bits 9-0 of I2C_ICSAR), and its own slave address has 10 bits (bits 9-0 of I2C_ICOAR).(1)

(1) 10-bit addressing mode is not supported feature on this device.

Table 11-1213. I2C_ICMDR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	RM	R/W	0h	Repeat mode bit (only applicable when the I^2C is a master-transmitter). The RM, STT, and STP bits determine when the I^2C starts and stops data transmissions (see Table 11-1214). If the I^2C is configured in slave mode, the RM bit is don't care. • 0 = Nonrepeat mode. The value in the data count register I2C_ICCNT determines how many data words are received/transmitted by the I^2C. • 1 = Repeat mode. Data words are continuously received/transmitted by the I^2C until the STP bit is manually set to 1, regardless of the value in I2C_ICCNT.
6	DLB	R/W	0h	Digital loopback mode bit (only applicable when the I^2C is a master-transmitter). This bit disables or enables the digital loopback mode of the I^2C. The effects of this bit are shown in Figure 11-561. Note that DLB mode in the free data format mode (DLB = 1 and FDF = 1) is not supported. • 0 = Digital loopback mode is disabled. • 1 = Digital loopback mode is enabled. In this mode, the MST bit must be set to 1 and data transmitted out of I2C_ICDXR is received in I2C_ICDRR after n DSP cycles by an internal path, where: n = ((I^2C input clock frequency/module clock frequency) × 8) The transmit clock is also the receive clock. The address transmitted on the SDA pin is the address in I2C_ICOAR.
5	IRS	R/W	0h	I^2C reset bit. Note that if IRS is reset during a transfer, it can cause the I^2C bus to hang. For more information, see Section 11.8.4.8. • 0 = The I^2C is in reset/disabled. When this bit is cleared to 0, all status bits (in I2C_ICSTR) are set to their default values. SDA and SCL are in a high-impedance state. • 1 = The I^2C is enabled.
4	STB	R/W	0h	START byte mode bit (only applicable when the I^2C is a master). As described in version 2.1 of the Philips I^2C-bus specification, the START byte can be used to help a slave that needs extra time to detect a START condition. When the I^2C is a slave, the I^2C ignores a START byte from a master, regardless of the value of the STB bit. • 0 = The I^2C is not in the START byte mode. • 1 = The I^2C is in the START byte mode. When you set the START condition bit (STT), the I^2C begins the transfer with more than just a START condition. Specifically, it generates: 1. A START condition 2. A START byte (0000 0001b) 3. A dummy acknowledge clock pulse 4. A repeated START condition The I^2C sends the slave address that is in I2C_ICSAR.
3	FDF	R/W	0h	Free data format mode bit. Note that DLB mode in the free data format mode (DLB = 1 and FDF = 1) is not supported (see Table 11-1215). • 0 = Free data format mode is disabled. Transfers use the 7-bit addressing format selected by the XA bit. • 1 = Free data format mode is enabled.

Table 11-1213. I2C_ICMDR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	BC	R/W	0h	Bit count bits. BC defines the number of bits (1 to 8) in the next data word that is to be received or transmitted by the I^2C. The number of bits selected with BC must match the data size of the other device. Note that when BC = 0, a data word has 8 bits. If the bit count is less than 8, receive data is right aligned in the D bits of I2C_ICDRR and the remaining D bits are undefined. Also, transmit data written to I2C_ICDXR must be right aligned. • 0 = 8 bits per data word • 1h = 1 bit per data word • 2h = 2 bits per data word • 3h = 3 bits per data word • 4h = 4 bits per data word • 5h = 5 bits per data word • 6h = 6 bits per data word • 7h = 7 bits per data word

Table 11-1214. Master-Transmitter/Receiver Bus Activity Defined by RM, STT, and STP Bits

I2C_ICMDR Bit				
RM	STT	STP	Bus Activity (1)	Description
0	0	0	None	No activity
0	0	1	P	STOP condition
0	1	0	S-A-D..(n)..D	START condition, slave address, n data words (n = value in I2C_ICCNT)
0	1	1	S-A-D..(n)..D-P	START condition, slave address, n data words, STOP condition (n = value in I2C_ICCNT)
1	0	0	None	No activity
1	0	1	P	STOP condition
1	1	0	S-A-D-D-D..	Repeat mode transfer: START condition, slave address, continuous data transfers until STOP condition or next START condition
1	1	1	None	Reserved bit combination (No activity)

(1) A = Address; D = Data word; P = STOP condition; S = START condition

Table 11-1215. How the MST and FDF Bits Affect the Role of TRX Bit

I2C_ICMDR Bit			
MST	FDF	I^2C State	Function of TRX Bit
0	0	In slave mode but not free data format mode	TRX is a don't care. Depending on the command from the master, the I^2C responds as a receiver or a transmitter.
0	1	In slave mode and free data format mode	The free data format mode requires that the transmitter and receiver be fixed. TRX identifies the role of the I^2C: • TRX = 0: The I^2C is a receiver. • TRX = 1: The I^2C is a transmitter.
1	0	In master mode but not free data format mode	TRX identifies the role of the I^2C: • TRX = 0: The I^2C is a receiver. • TRX = 1: The I^2C is a transmitter.
1	1	In master mode and free data format mode	The free data format mode requires that the transmitter and receiver be fixed. TRX identifies the role of the I^2C: • TRX = 0: The I^2C is a receiver. • TRX = 1: The I^2C is a transmitter.

Figure 11-561. Block Diagram Showing the Effects of the Digital Loopback Mode (DLB) Bit

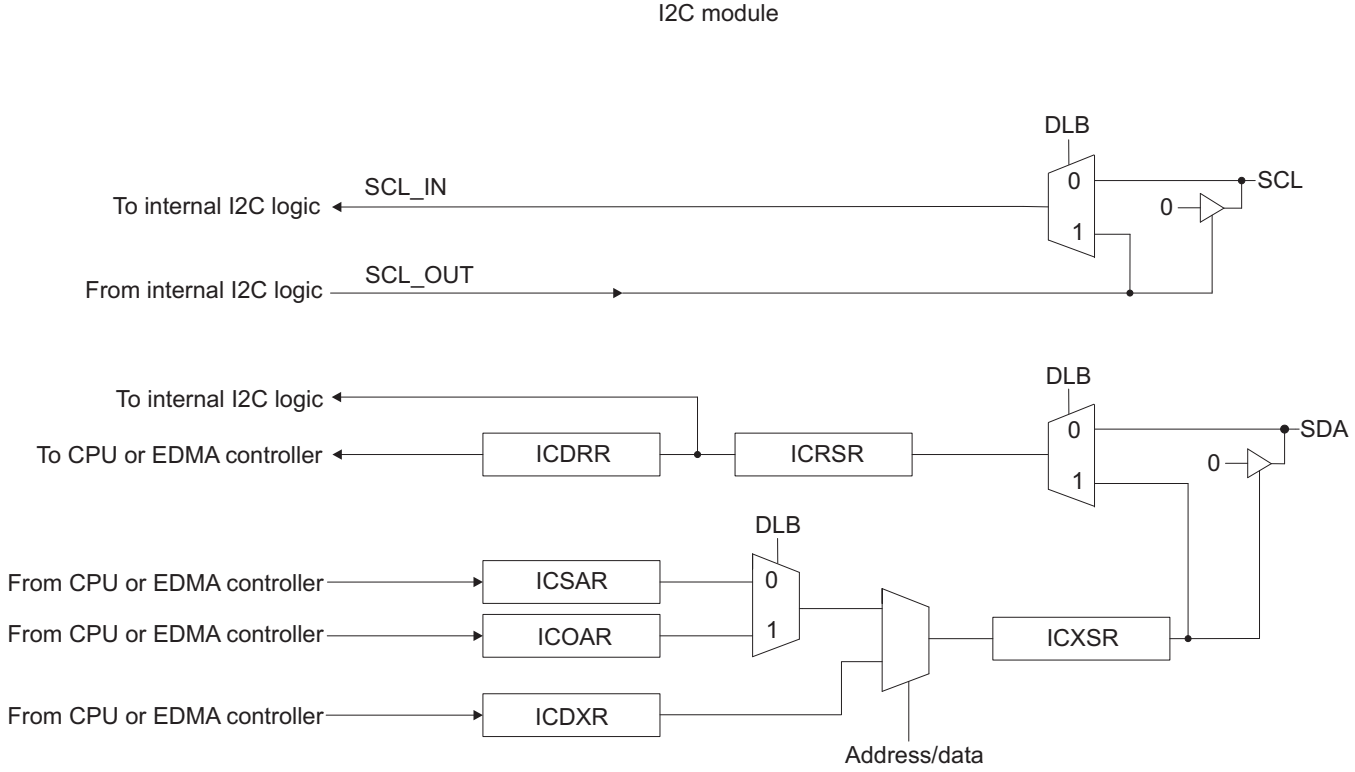


Table 11-1216. Register Call Summary for I2C_ICMDR

I2C Programming Guide • Initialize the I2C Controller: [0][1][2] • I2C Configure the Module Before Enabling the Controller: [0] • Initiate a Transfer: [0][1][2][3]
I2C Registers • I2C_ICDRR Register (Offset = 18h) [reset = 0h]: [0] • I2C_ICDXR Register (Offset = 20h) [reset = 0h]: [0] • I2C_ICSAR Register (Offset = 1Ch) [reset = 0h]: [0][1][2][3] • I2C_ICCNT Register (Offset = 14h) [reset = 0h]: [0][1][2][3][4][5] • I2C_ICSTR Register (Offset = 8h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12][13] • I2C_ICOAR Register (Offset = 0h) [reset = 0h]: [0][1][2] • I2C_ICPSC Register (Offset = 30h) [reset = 0h]: [0][1] • I2C Registers: [0] • I2C_ICMDR Register (Offset = 24h) [reset = 0h]: [0][1][2][3]
I2C Clock Divider Registers (I2C_ICCLKL and I2C_ICCLKH) • I2C_ICCLKL Register (Offset = Ch) [reset = 0h]: [0] • I2C_ICCLKH Register (Offset = 10h) [reset = 0h]: [0]
I2C Functional Description • I2C Clock Generation: [0] • NACK Bit Generation: [0][1][2][3][4][5][6][7][8][9] • I2C Emulation Considerations: [0] • Free Data Format: [0][1] • Using a Repeated START Condition: [0] • I2C Reset: [0] • 7-Bit Addressing Format: [0][1] • I2C START and STOP Conditions: [0]

11.8.6.10 I2C_ICIVR Register (Offset = 28h) [reset = 0h]

The I²C interrupt vector register I2C_ICIVR is used by the CPU to determine which event generated the I²C interrupt. Reading I2C_ICIVR clears the interrupt flag; if other interrupts are pending, a new interrupt is generated. If there are more than one interrupt flag, reading I2C_ICIVR clears the highest priority interrupt flag. Note that you must read (clear) I2C_ICIVR before doing another start; otherwise, I2C_ICIVR could contain an incorrect (old interrupt flags) value. The I2C_ICIVR register is shown in Figure 11-562 and described in Table 11-1218.

Table 11-1217. I2C_ICIVR Instances

Instance	Physical Address
I2C_0	0253 0028h
I2C_1	0253 0428h
I2C_2	0253 0828h

Figure 11-562. I2C_ICIVR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													INTCODE		
R-0h													R-0h		

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1218. I2C_ICIVR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Value = 0 These reserved bit location are always read as zeroes. A value written to this field has no effect.
2-0	INTCODE	R	0h	Value = 0-7h Interrupt code bits. The binary code in INTCODE indicates which event generated an I²C interrupt. • 0h = None • 1h = Arbitration-lost interrupt (AL) • 2h = No-acknowledgment interrupt (NACK) • 3h = Register-access-ready interrupt (ARDY) • 4h = Receive-data-ready interrupt (ICRDRDY) • 5h = Transmit-data-ready interrupt (ICXRDY) • 6h = Stop condition detected interrupt (SCD) • 7h = Address-as-slave interrupt (AAS)

Table 11-1219. Register Call Summary for I2C_ICIVR

I2C Registers • I2C_ICIVR Register (Offset = 28h) [reset = 0h]: [0][1][2][3][4][5] • I2C Registers: [0]
I2C Functional Description • I2C Interrupt Support: [0][1][2][3]

11.8.6.11 I2C_ICEMDR Register (Offset = 2Ch) [reset = 0h]

The I²C extended mode register I2C_ICEMDR is used to indicate which condition generates a transmit data-ready interrupt. The I2C_ICEMDR register is shown in Figure 11-563 and described in Table 11-1221.

Table 11-1220. I2C_ICEMDR Instances

Instance	Physical Address
I2C_0	0253 002Ch
I2C_1	0253 042Ch
I2C_2	0253 082Ch

Figure 11-563. I2C_ICEMDR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						IGNACK	BCM
R-0h						R/W-0h	R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1221. I2C_ICEMDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	These reserved bit location are always read as 0s. A value written to this field has no effect.
1	IGNACK	R/W	0h	Ignore NACK mode. • 0 = The master transmitter operates normally, discontinue the data transfer, and set the ARDY and NACK status bits when a NACK signal is received from the slave. • 1 = The master transmitter ignores a NACK received from the slave.
0	BCM	R/W	1h	Backward compatibility mode bit. Determines which condition generates a transmit data ready interrupt. The BCM bit has an effect only when the I²C is operating as a slave-transmitter. • 0 = The transmit data ready interrupt is generated when the master requests more data by sending an acknowledge signal after the transmission of the last data. • 1 = The transmit data ready interrupt is generated when the data in I2C_ICDXR is copied to ICXSR.

Table 11-1222. Register Call Summary for I2C_ICEMDR

I2C Registers
- I2C_ICEMDR Register (Offset = 2Ch) [reset = 0h]: [0][1]
- I2C Registers: [0]

11.8.6.12 I2C_ICPSC Register (Offset = 30h) [reset = 0h]

The I²C prescaler register I2C_ICPSC is used for dividing down the I²C input clock to obtain the desired module clock for the operation of the I²C. The I2C_ICPSC register is shown in Figure 11-564 and described in Table 11-1224.

The IPSC bits must be initialized while the I²C is in reset (IRS = 0 in I2C_ICMDR). The prescaled frequency takes effect only when the IRS bit is changed to 1. Changing the IPSC value while IRS = 1 has no effect.

Table 11-1223. I2C_ICPSC Instances

Instance	Physical Address
I2C_0	0253 0030h
I2C_1	0253 0430h
I2C_2	0253 0830h

Figure 11-564. I2C_ICPSC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								IPSC							
R-0h																								R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1224. I2C_ICPSC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Value = 0 These reserved bit location are always read as zeroes. A value written to this field has no effect.
7-0	IPSC	R/W	0h	Value = 0-FFh I²C prescaler divide-down value. IPSC determines how much the I²C input clock is divided to create the I²C module clock: • I²C clock frequency = I²C input clock frequency/(IPSC + 1) Note: IPSC must be initialized while the I²C is in reset (IRS = 0 in I2C_ICMDR).

Table 11-1225. Register Call Summary for I2C_ICPSC

I2C Programming Guide • I2C Configure the Module Before Enabling the Controller: [0]
I2C Registers • I2C_ICPSC Register (Offset = 30h) [reset = 0h]: [0][1] • I2C Registers: [0]
I2C Functional Description • I2C Clock Generation: [0][1]

11.8.6.13 I²C Peripheral Identification Registers (I2C_ICPID1 and I2C_ICPID2)

The I²C peripheral identification registers (ICPIDn) contain identification data (class, revision, and type) for the peripheral.

11.8.6.13.1 I2C_ICPID1 Register (Offset = 34h) [reset = 00004415h]

I2C_ICPID1 is shown in Figure 11-565 and described in Table 11-1227.

Table 11-1226. I2C_ICPID1 Instances

Instance	Physical Address
I2C_0	0253 0034h
I2C_1	0253 0434h
I2C_2	0253 0834h

Figure 11-565. I2C_ICPID1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-00004415h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1227. I2C_ICPID1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	00004415h	TI internal data. Identifies revision of peripheral.

Table 11-1228. Register Call Summary for I2C_ICPID1

I2C Peripheral Identification Registers (I2C_ICPID1 and I2C_ICPID2) • I2C_ICPID1 Register (Offset = 34h) [reset = 00004415h]: [0]
I2C Registers • I2C Registers: [0]

11.8.6.13.2 I2C_ICPID2 Register (Offset = 38h) [reset = 00002206h]

I2C_ICPID2 is shown in Figure 11-566 and described in Table 11-1230.

Table 11-1229. I2C_ICPID2 Instances

Instance	Physical Address
I2C_0	0253 0038h
I2C_1	0253 0438h
I2C_2	0253 0838h

Figure 11-566. I2C_ICPID2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-00002206h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1230. I2C_ICPID2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	00002206h	TI internal data. Identifies revision of peripheral.

Table 11-1231. Register Call Summary for I2C_ICPID2

I2C Peripheral Identification Registers (I2C_ICPID1 and I2C_ICPID2) • I2C_ICPID2 Register (Offset = 38h) [reset = 0h]: [0]
I2C Registers • I2C Registers: [0]

11.8.7 I2C Appendix: I2C Lockup Issue

SoCs can use a memory device connected with an I^2C interface as a source for boot code. A reset of the SoC while it is reading from the memory device can cause the I^2C interface to enter a state which will prevent the SoC from accessing the attached memory device. Once the I^2C interface enters this state it will not recover, even if additional SoC resets are issued. This condition will prevent the SoC from booting until the power is removed from the system or from the I^2C device. This failure mode is possible as part of the normal behavior of the I^2C interface as described in the I^2C specification. To prevent a system from entering this unrecoverable state, a hardware workaround is needed to force the I^2C interface into a usable condition. This appendix describes the problem and a hardware workaround that can be used to fix the issue.

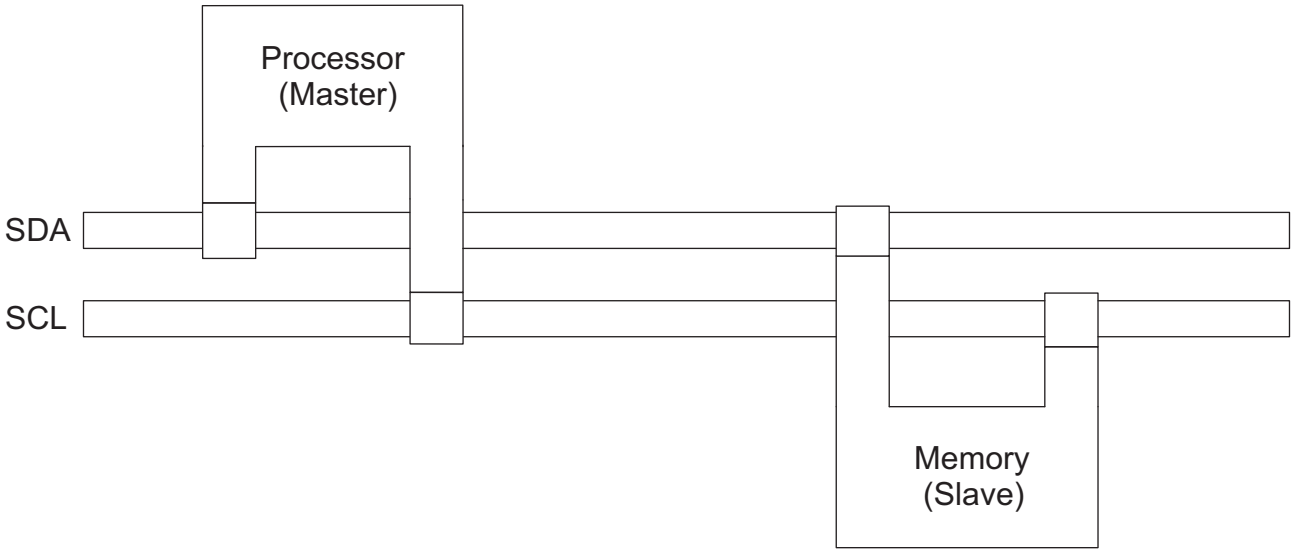
The I^2C interface consists of two bidirectional lines, SDA and SCL, which are pulled to a positive supply voltage with external resistors. The output stages for these signals are open-drain or open-collector, allowing multiple devices to share the same I^2C interfaces. These devices can be either a master-type device or a slave-type device.

Master devices initiate data transfers and generate the clock signals to permit transfers. Once a master device has initiated a transfer and gained control of the bus through the arbitration process it continues to control the bus until the end of an access. Under certain conditions if a reset to the controlling master device occurs during a transfer, the bus can remain in a locked state which does not allow the transfer to be completed and does not allow any other master device to gain control. This condition, inherent to the I^2C specification, must be considered when designing a system dependent on I^2C accesses.

This appendix presents a brief overview of the I^2C bus and describes the condition that can halt accesses. The scenario of a single I^2C master device connected to a slave device is described as well as a strategy to recover from the locked condition. This is followed by a discussion of the additional complications found in a multi-master environment.

In the simplest environment, a single master device will be connected to one or more slave devices. The example, shown in Figure 11-567 is a processor attached to a memory device using the I^2C interface.

Figure 11-567. Example I^2C Interface

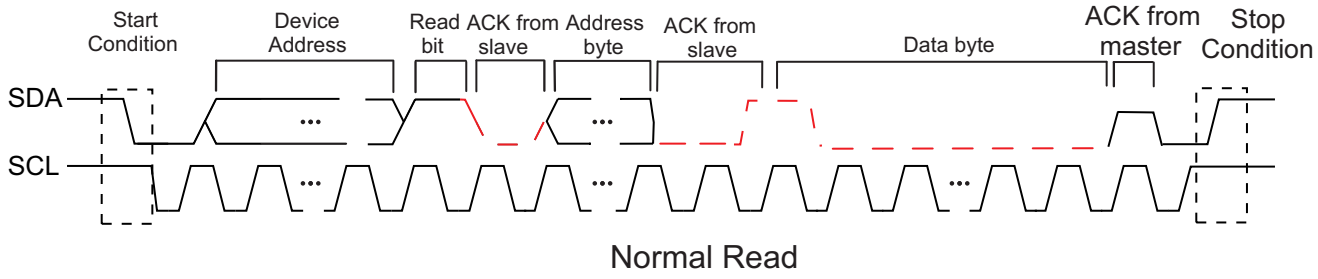


To initiate a transfer the master device will generate a start condition. The start condition is defined as a high-to-low transition on the SDA line while SCL is high. Once the master has initiated a transfer with a start condition, the bus is considered busy until the master generates a stop condition. The stop condition is defined as a low-to-high transition of the SDA line while the SCL is high. Once the transfer is initiated by the start condition, the SDA line must remain stable while SCL is high. SDA will only transition when SCL is low until the master releases the bus with a stop condition.

The start condition is followed by a series of byte transfers consisting of eight bits followed by a single acknowledge bit. The first byte contains the slave device address and a read/write bit. Once the slave device has received the correct address and the read/write bit, it will drive the SDA signal low as an acknowledgement that it has received the information correctly. The bytes that follow the device address vary in function from device to device. In the case of a sequential write to a slave memory device, the device address will be followed by a word address byte and a series of data bytes. Again, each byte transmitted from the master is acknowledged by the slave by driving the ninth bit low. When the master is done transmitting data it will send the stop condition releasing the bus.

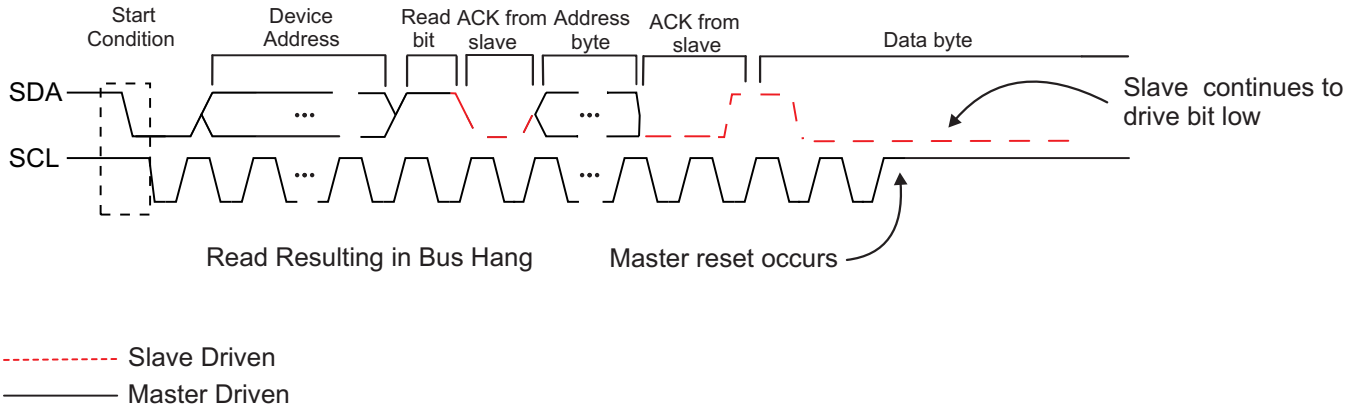
During a write cycle the slave will only drive the SDA low during the acknowledge (ACK). During a read cycle the slave will drive the data byte one bit at a time in response to the read command. There is a period of time during which the slave device will drive eight consecutive bits based on the clock provided by the master. When the master drives the clock low, it will drive the next bit onto the SDA line. When the master stops driving SCL, the pull-up resistor will return it to a high state signaling the slave device to hold the SDA signal level. Once all eight bits are clocked onto the SDA, the master can end the access by generating a stop condition freeing the bus. Figure 11-568 shows a normal read cycle. The period when the slave is driving SDA is indicated by the red lines.

Figure 11-568. Normal Read Cycle



If a reset to the master device occurs during the data transfer of a read cycle, there is the possibility of the bus entering in a state that will prevent any master from regaining control. Consider the situation shown in Figure 11-569.

Figure 11-569. Bus-Hang Read Cycle

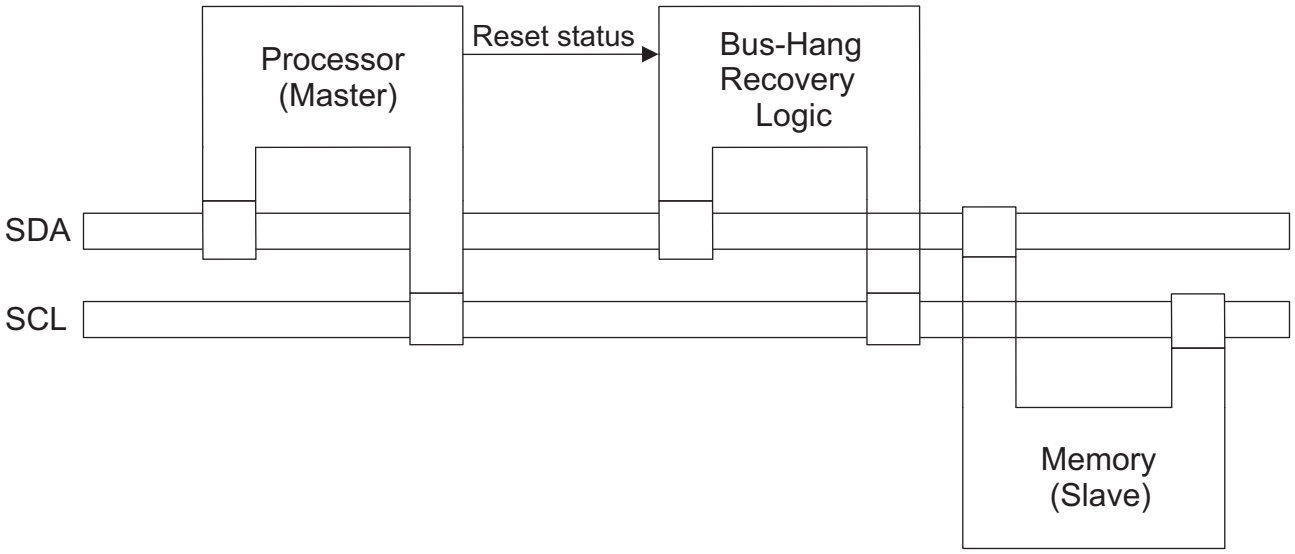


The master initiates a read in the normal fashion but a reset occurs while the SCL is high and the slave is driving the SDA low. The slave device will continue to drive SDA low waiting for the master to provide the next clock however the master has been reset and is no longer attempting to complete the read cycle. In addition, if the master attempts to access the bus after its reset has been released it will fail to arbitrate for control.

Arbitration for the I²C bus is performed on the SDA line. The master will attempt to generate a start condition beginning with both the SDA and the SCL lines high. Remember that the SDA and SCL lines are open-collector so the slave device will continue to hold the SDA line in a low state. When the master detects that the SDA line is not high, it will assume that some other device is in control of the bus and discontinue its attempt to start an access. This state will continue indefinitely blocking all attempts to access devices on the I²C interface.

This state is especially debilitating if the I²C slave is a memory device used as a boot memory. If the bus hang occurs during a reset, the processor acting as an I²C master will not be able to read its boot code and will fail to initialize. The processor cannot recover from this condition, even if additional resets are applied. To resolve this condition, additional hardware attached to the I²C bus is required (see Figure 11-570).

Figure 11-570. I²C Master Bus Hang



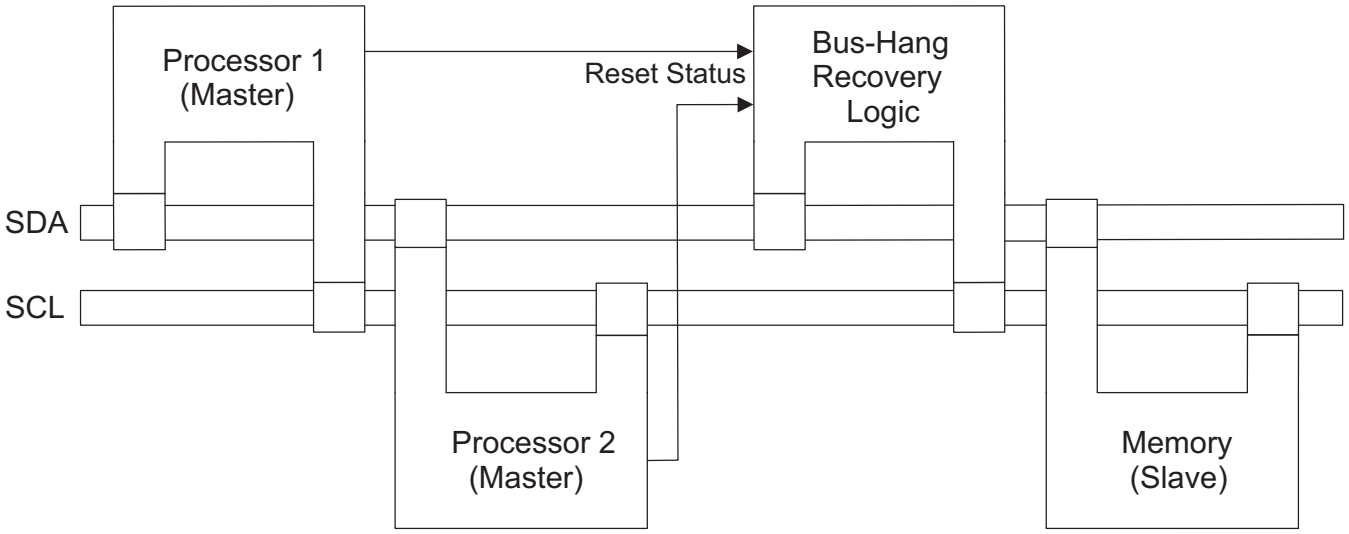
The bus-hang recovery logic needs to have knowledge of the reset status of the processor either through a RESETSTAT or a GPIO signal as well as the ability to monitor and drive the SDA and SCL signals. This logic may be a state machine in a programmable logic device or a software-driven solution in a microcontroller or processor using GPIO pins, but it must meet the open-collector driver requirements for the SDA and SCL signals to ensure the proper operation of the I²C.

Once the logic senses that the processor has entered reset, it should sample the SDA signal to determine if it is stuck in a low state. In a single master environment, the SDA should be pulled high as soon as the reset is applied to the master. If the logic senses that the SDA is low during a reset state, it can take action to free the bus. The logic should toggle SCL nine times ending in a high state.

These clock pulses must meet all the specification timing requirements for the high and low states. This will clock any remaining data bits from the slave memory device. Once these bits have been transmitted on the SDA, the slave device should release SDA and allow it to return to a high state. The recovery logic should sample SDA to ensure it has been released. The I²C bus should now be ready to respond to any requests for data by the processor.

The I²C interface with multiple master devices requires more complex logic to clear a bus-hang condition. In addition to determining that a master has been reset, the logic must also determine that a second master is not making a valid access to the bus (see Figure 11-571).

Figure 11-571. Multiple Master Bus Hang



In this environment, a reset to one of the processors during a read can cause the same bus-hang condition described above but detection of that condition is more complicated. When a single master is present on the bus, it is clear that there should not be any activity while that master is in reset. In this case, it is sufficient to sample the SDA signal while the RESET to the master is active, as described above. If multiple master devices are connected to the I^2C, a second master can access a working I^2C interface while the first master is in reset. This would create the condition where SDA is low while the first processor is in reset. In this environment, the logic must monitor the reset status from both master devices to determine if either is in reset.

Once a reset has been detected, the logic should monitor the SDA signal and no further action is needed. If the SDA signal is low and the SCL is toggling, another master is accessing the slave device and the interface is not hung. Again, no further action is needed. If the SDA signal is low and the SCL signal is high for too long the bus-hang condition is present and the logic must take action to recover.

The period of time that defines a failure can vary, depending on the slave devices on the bus and the speed of the I^2C interface. Slave devices can only add wait time to accesses by holding the SCL low so the average time that the SCL is high and the SDA is low should depend on the frequency that the masters are using to drive the interface. Once the slowest frequency is determined, the longest value for SCL high during an access should be multiplied by four to determine the period of time that defines a bus-hang condition. The logic should sample the SDA and SCL many times during the expected clock period to detect a stall condition.

If a bus-hang condition is present, the logic should send nine clock pulses onto SCL complete the in-process read from the slave memory. The slave should release SDA and allow it to be pulled high. In multi-master environments, a stop sequence should be generated after the nine clock pulses to release the bus for arbitration. This should clear the bus-hang condition and allow the I^2C master devices to access the memory.

11.9 Multi-Channel Audio Serial Port (McASP)

This section describes the Multi-channel Audio Serial Port (McASP) module.

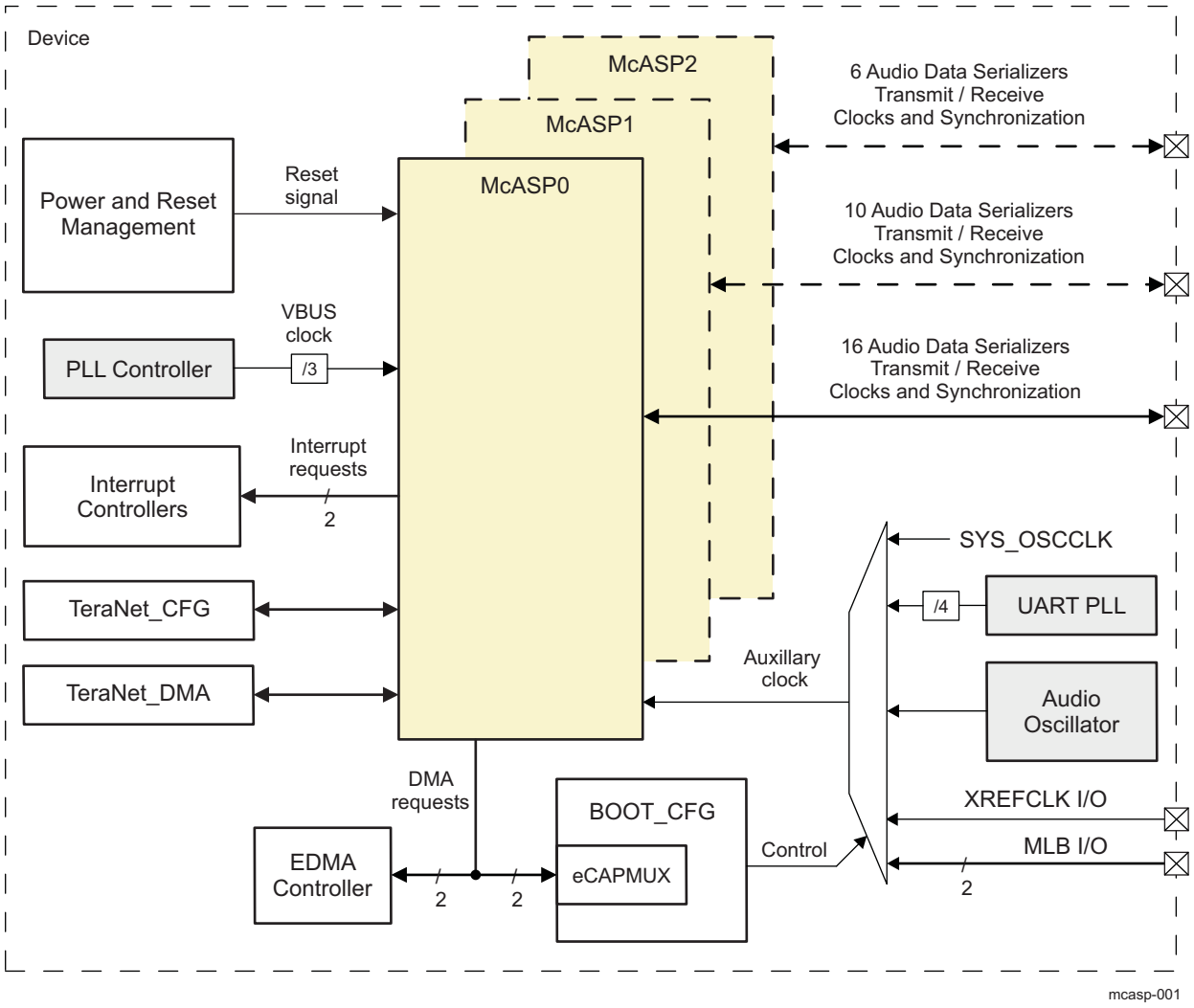
11.9.1 McASP Overview

The Multi-channel Audio Serial Port (McASP) module functions as a general-purpose audio serial port optimized for the needs of multichannel audio applications. The McASP supports transmission and reception of time-division multiplexed (TDM) and Inter-IC Sound (I^2S) protocols. In addition, it supports intercomponent digital audio interface transmission (DIT).

The McASP consists of transmit and receive sections that may operate synchronized, or completely independently with separate master clocks, bit clocks, and frame syncs, and using different transmit modes with different bit-stream formats. The McASP module also includes up to 16 serializers that can be individually enabled to either transmit or receive.

Figure 11-572 shows the McASP modules in the device.

Figure 11-572. McASP Modules Overview



The device integrates three McASP modules (McASP0, McASP1, and McASP2) with:

- McASP0 supporting 16 serializers with independent TX/RX clock zones
- McASP1 supporting 10 serializers with independent TX/RX clock zones
- McASP2 supporting 6 serializers with independent TX/RX clock zones

Each McASP module includes the following main features:

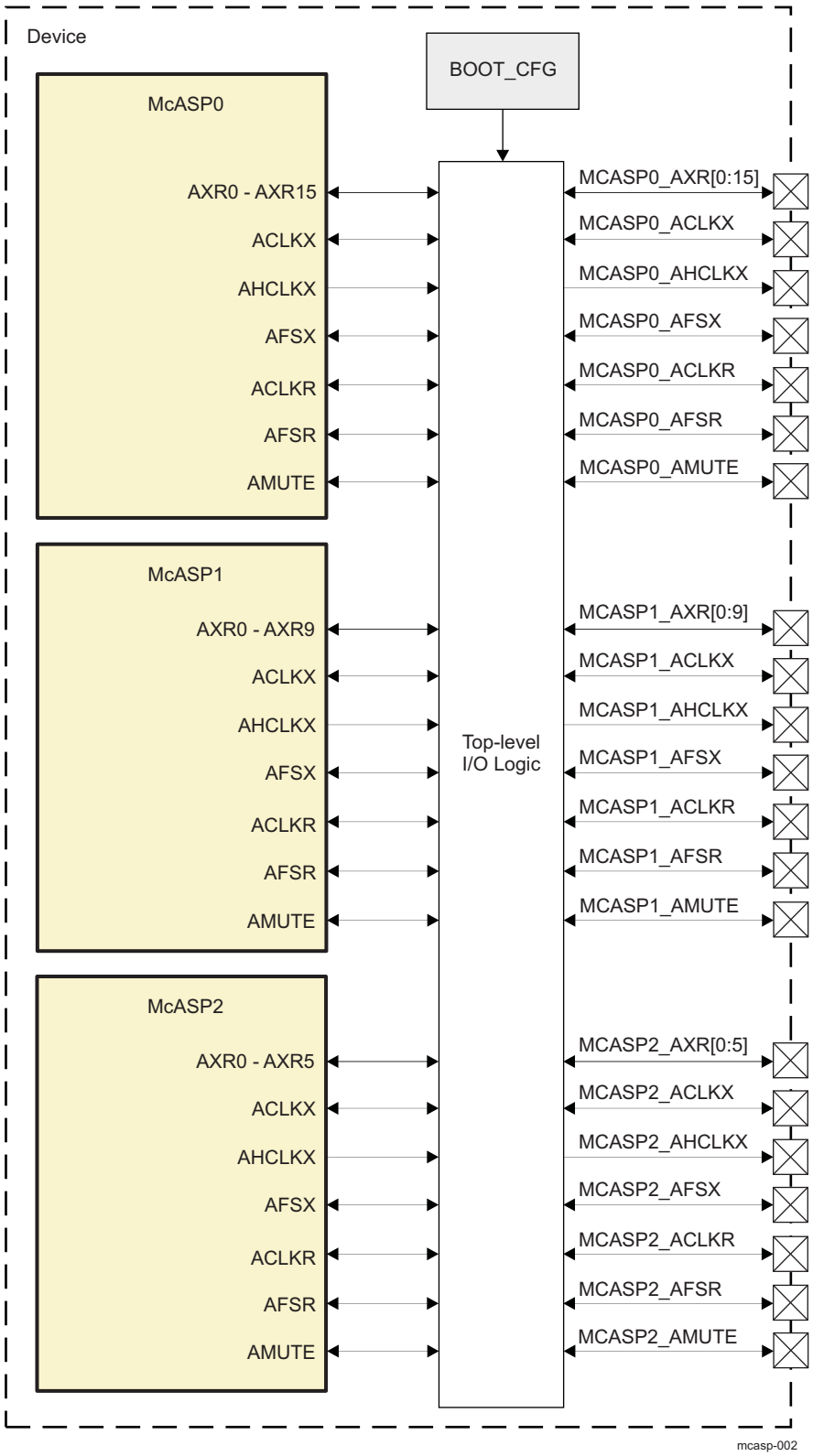
- Up to 16 individually assignable serializers, each with its own data pins (AXR)
- A single 32-bit buffer per serializer for transmit and receive operations
- 2x interconnect slave interface ports:
  - A configuration (CFG) port
  - A slave DMA data port synchronized with functional clock
- Two independent clock generator modules for transmit and receive
  - Clocking flexibility allows the McASP to receive and transmit at different rates. For example, the McASP can receive data at 48 kHz but output up-sampled data at 96 kHz or 192 kHz.
- Configurable functional clocks:
  - May be generated internally (master mode)
  - May be supplied by an external device (slave mode)
  - May be divided down internally
- Independent transmit and receive modules, each providing:
  - Programmable clock and frame sync generator
  - TDM streams from 2 to 32, and 384 time slots
  - Support for time slot sizes of 8, 12, 16, 20, 24, 28, and 32 bits
  - Data formatter for bit manipulation
- Glueless connection to audio analog-to-digital converters (ADC), digital-to-analog converters (DAC), codec, digital audio interface receiver (DIR), and S/PDIF transmit physical layer components.
- Support for wide variety of I^2S and similar bit-stream formats
- Integrated digital audio interface transmitter (DIT):
  - S/PDIF, IEC60958-1, AES-3 formats.
  - Enhanced channel status/user data RAM
- 384-slot TDM with external digital audio interface receiver (DIR) device
  - For DIR reception, an external DIR receiver integrated circuit should be used with I^2S output format and connected to the McASP receive section
- Support for 2x DMA requests (1 per direction) per each McASP module:
  - 1 level-sensitive transmit direct memory access (DMA) request common for all of the McASP serializers
  - 1 level-sensitive receive direct memory access (DMA) request common for all of the McASP serializers
- One transmit interrupt request common for all serializers per McASP module
- One receive interrupt request common for all serializers per McASP module
- Extensive error checking and recovery:
  - Transmit underruns and receiver overruns due to the system not meeting real-time requirements
  - Early or late frame sync in TDM mode
  - DMA error due to incorrect programming
- McASP Audio FIFO (AFIFO):
  - Provides additional data buffering
  - Provides added tolerance to variations in host/DMA controller response times
  - May be used as a DMA event pacer
  - Independent Read FIFO and Write FIFO
  - 256 bytes of RAM for each FIFO (read and write), where
    - 256 bytes = four 32-bit words per serializer in the case of 16 data pins
    - 256 bytes = 64 32-bit words in the case of one data pin
  - Option to bypass Write FIFO and/or Read FIFO independently

NOTE: Because each serializer is assigned to a single McASP data pin (AXR), the user may configure each serializer for either transmit or receive mode, but not both directions at the same time. Each serializer has its own direction control.

11.9.2 McASP Environment

This section describes the McASP application fields from an environment point of view (external connections). This section also lists the possible interfaces and describes the protocol and data format used in each case. Figure 11-573 shows the McASP modules in their environment in the device.

Figure 11-573. McASP Modules Environment



11.9.2.1 McASP Signals

Table 11-1232 describes the McASP module pins, their corresponding signal names at device level and specifies their links to functions.

Table 11-1232. McASP I/O Signals

Module Pin Name	Device Level Signal Name	I/O[1]	Description	Module Pin Reset Value
		McASP0 module		
AXR0	MCASP0_AXR[0]	I/O	Audio transmit/receive data – channel 0	HiZ
AXR1	MCASP0_AXR[1]	I/O	Audio transmit/receive data – channel 1	HiZ
AXR2	MCASP0_AXR[2]	I/O	Audio transmit/receive data – channel 2	HiZ
AXR3	MCASP0_AXR[3]	I/O	Audio transmit/receive data – channel 3	HiZ
AXR4	MCASP0_AXR[4]	I/O	Audio transmit/receive data – channel 4	HiZ
AXR5	MCASP0_AXR[5]	I/O	Audio transmit/receive data – channel 5	HiZ
AXR6	MCASP0_AXR[6]	I/O	Audio transmit/receive data – channel 6	HiZ
AXR7	MCASP0_AXR[7]	I/O	Audio transmit/receive data – channel 7	HiZ
AXR8	MCASP0_AXR[8]	I/O	Audio transmit/receive data – channel 8	HiZ
AXR9	MCASP0_AXR[9]	I/O	Audio transmit/receive data – channel 9	HiZ
AXR10	MCASP0_AXR[10]	I/O	Audio transmit/receive data – channel 10	HiZ
AXR11	MCASP0_AXR[11]	I/O	Audio transmit/receive data – channel 11	HiZ
AXR12	MCASP0_AXR[12]	I/O	Audio transmit/receive data – channel 12	HiZ
AXR13	MCASP0_AXR[13]	I/O	Audio transmit/receive data – channel 13	HiZ
AXR14	MCASP0_AXR[14]	I/O	Audio transmit/receive data – channel 14	HiZ
AXR15	MCASP0_AXR[15]	I/O	Audio transmit/receive data – channel 15	HiZ
ACLKX	MCASP0_ACLKX	I/O	Transmit bit clock	HiZ
AHCLKX	MCASP0_AHCLKX	O	Transmit high-frequency master clock	HiZ
AFSX	MCASP0_AFSX	I/O	Transmit frame synchronization	HiZ
ACLKR	MCASP0_ACLKR	I/O	Receive bit clock	HiZ
AFSR	MCASP0_AFSR	I/O	Receive frame synchronization	HiZ
AMUTE	MCASP0_AMUTE	I/O	Audio mute	HiZ
		McASP1 module		
AXR0	MCASP1_AXR[0]	I/O	Audio transmit/receive data – channel 0	HiZ
AXR1	MCASP1_AXR[1]	I/O	Audio transmit/receive data – channel 1	HiZ
AXR2	MCASP1_AXR[2]	I/O	Audio transmit/receive data – channel 2	HiZ
AXR3	MCASP1_AXR[3]	I/O	Audio transmit/receive data – channel 3	HiZ
AXR4	MCASP1_AXR[4]	I/O	Audio transmit/receive data – channel 4	HiZ
AXR5	MCASP1_AXR[5]	I/O	Audio transmit/receive data – channel 5	HiZ
AXR6	MCASP1_AXR[6]	I/O	Audio transmit/receive data – channel 6	HiZ
AXR7	MCASP1_AXR[7]	I/O	Audio transmit/receive data – channel 7	HiZ
AXR8	MCASP1_AXR[8]	I/O	Audio transmit/receive data – channel 8	HiZ
AXR9	MCASP1_AXR[9]	I/O	Audio transmit/receive data – channel 9	HiZ
ACLKX	MCASP1_ACLKX	I/O	Transmit bit clock	HiZ
AHCLKX	MCASP1_AHCLKX	O	Transmit high-frequency master clock	HiZ
AFSX	MCASP1_AFSX	I/O	Transmit frame synchronization	HiZ
ACLKR	MCASP1_ACLKR	I/O	Receive bit clock	HiZ
AFSR	MCASP1_AFSR	I/O	Receive frame synchronization	HiZ
AMUTE	MCASP1_AMUTE	I/O	Audio mute	HiZ
		McASP2 module		
AXR0	MCASP2_AXR[0]	I/O	Audio transmit/receive data – channel 0	HiZ

(1) I = Input; O = Output; I/O = Bidirectional

Table 11-1232. McASP I/O Signals (continued)

Module Pin Name	Device Level Signal Name	I/O[1]	Description	Module Pin Reset Value
AXR1	MCASP2_AXR[1]	I/O	Audio transmit/receive data – channel 1	HiZ
AXR2	MCASP2_AXR[2]	I/O	Audio transmit/receive data – channel 2	HiZ
AXR3	MCASP2_AXR[3]	I/O	Audio transmit/receive data – channel 3	HiZ
AXR4	MCASP2_AXR[4]	I/O	Audio transmit/receive data – channel 4	HiZ
AXR5	MCASP2_AXR[5]	I/O	Audio transmit/receive data – channel 5	HiZ
ACLKX	MCASP2_ACLKX	I/O	Transmit bit clock	HiZ
AHCLKX	MCASP2_AHCLKX	O	Transmit high-frequency master clock	HiZ
AFSX	MCASP2_AFSX	I/O	Transmit frame synchronization	HiZ
ACLKR	MCASP2_ACLKR	I/O	Receive bit clock	HiZ
AFSR	MCASP2_AFSR	I/O	Receive frame synchronization	HiZ
AMUTE	MCASP2_AMUTE	I/O	Audio mute	HiZ

NOTE: A serializer AXRn data pin (where n = 0 up to 15) is shared between the transmit and receive logic of this serializer. The direction of data is determined in the MCASP_PDIR and the function (Tx or Rx) is selected in the corresponding serializer control registers MCASP_SRCTL0 through MCASP_SRCTL15.

11.9.2.2 McASP Protocols and Data Formats

11.9.2.2.1 Protocols Supported

The McASP modules supports a wide variety of protocols:

- Transmit section supports:
  - Wide variety of I^2S and similar bit-stream formats
  - TDM streams from 2 to 32 time slots
  - S/PDIF, IEC60958-1, AES-3 formats
- Receive section supports:
  - Wide variety of I^2S and similar bit-stream formats
  - TDM streams from 2 to 32 time slots
  - TDM stream of 384 time slots specifically designed for easy interface to external digital interface receiver (DIR) device transmitting DIR frames to McASP using the I^2S protocol (one time slot for each DIR subframe)

The transmit and receive sections of each module may each be individually programmed to support the following options on the basic serial protocol:

- Programmable clock and frame sync polarity (rising or falling edge): ACLKR/X, AHCLKR/X, and AFSR/X.
- Slot length (number of bits per time slot): 8, 12, 16, 20, 24, 28, 32 bits supported.
- Word length (bits per word): 8, 12, 16, 20, 24, 28, 32 bits; always less than or equal to the time slot length.
- First-bit data delay: 0, 1, 2 bit clocks.
- Left/right alignment of word inside slot.
- Bit order: MSB first or LSB first.
- Bit mask/pad/rotate function.
  - Automatically aligns data internally in either Q31 or integer formats.
  - Automatically masks nonsignificant bits (sets to 0, 1, or extends value of another bit).

NOTE: In I²S mode, the transmit and receive sections can support simultaneous transfers on up to all serial data pins operating as 192 kHz stereo channels.

In DIT mode for McASP, additional features of the transmitters are:

- Transmit-only mode 384 time slots (subframe) per frame.
- Biphase encoded LVCMOS output
- Channel status RAM (384 bits).
- User data RAM (384 bits).
- Separate valid bit (V) for subframe A, B.
- Stereo Support Only (Mono means send data 2x via software)

In DIT mode, the transmitter can support a 192 kHz frame rate (stereo) on up to all serial data pins simultaneously (note that the internal bit clock for DIT runs two times faster than the equivalent bit clock for I²S mode, due to the need to generate Biphase Mark Encoded Data).

NOTE: The McASP does NOT natively support DIR-mode reception (that is receiving in the S/PDIF format). To allow this, the McASP can use a DIR-input to I²S-output converter implemented by an external device (that is external DIR component). To facilitate reception in this case, the TDM mode of McASP receivers logic is extended to support a non-standard 384-slot TDM stream.

NOTE: An external transceiver must be connected to the McASP port in the device to translate the electrical signals delivered by the McASP (1.2 V or 1.8 V LVCMOS levels) to the electrical levels of the S/PDIF standard.

11.9.2.2.2 Definition of Terms

The serial bitstream transmitted or received by a McASP serializer is a long sequence of 1s and 0s on an audio transmit/receive pins AXRn. However, the sequence has a hierarchical organization that can be described in terms of frames of data, slots, words, and bits.

A basic synchronous serial interface consists of three important components: clock, frame sync, and data. Figure 11-574 shows two of the three basic components: the clock signal (ACLKX / ACLKR) and the data signals AXRn. Figure 11-574 does not specify whether the clock is for transmit (ACLKX) or receive (ACLKR) because the definitions of terms apply to both receive and transmit interfaces. In operation, each transmitter and receiver uses the signals ACLKX and ACLKR as serial clock, respectively. Optionally, a receiver can use ACLKX as the serial clock when a transmitter and receiver (not from the same serializer) of the McASP are configured to operate synchronously.

- Bit:

  A bit is the smallest entity in the serial data stream. The beginning and end of each bit is marked by an edge of the serial clock. The duration of a bit is a serial clock period. A '1' is represented by a logic high on AXRn pins for the entire duration of the bit. A 0 is represented by a logic low on an AXRn pin for the entire duration of the bit.

- Word:

  A word is a group of bits that make up the data being transferred between the McASP and the external device. Figure 11-574 shows an 8-bit word.

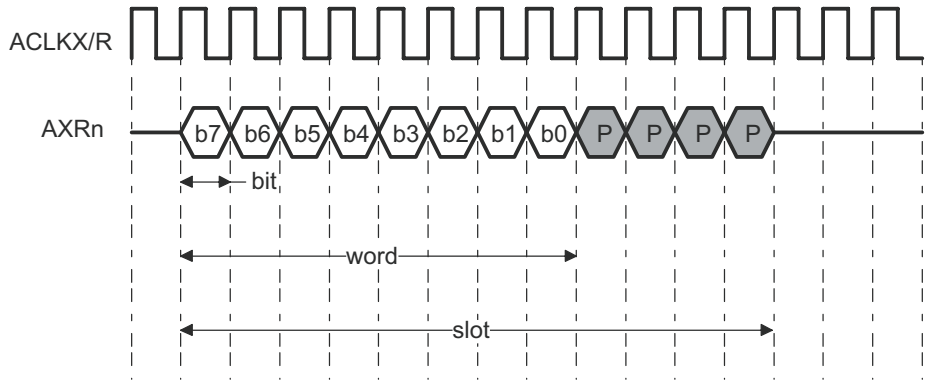
- Slot:

  A slot consists of the bits that make up the word and can consist of additional bits used to pad the word to a convenient number of bits for the interface between the McASP and the external device. In Figure 11-574, the audio data consists of only 8 bits of useful data (8-bit word), but it is padded with four 0s (12-bit slot) to satisfy the desired protocol in interfacing to an external device. Within a slot, the bits can be shifted out of the McASP on an AXRn pin with either MSB or LSB first.

  When the word size is smaller than the slot size, the word can be aligned to the left of the slot (beginning) or to the right of the slot (end). The additional bits in the slot not belonging to the word can

be padded with 0, 1, or with one of the bits (typically, the MSB or LSB) from the data word, that is left-aligned words within a slot are terminated with padding bits and right-aligned words within a slot are preceded by padding bits to fit in the slot size. Figure 11-575 shows these options.

Figure 11-574. McASP Definition of Bit, Word, and Slot



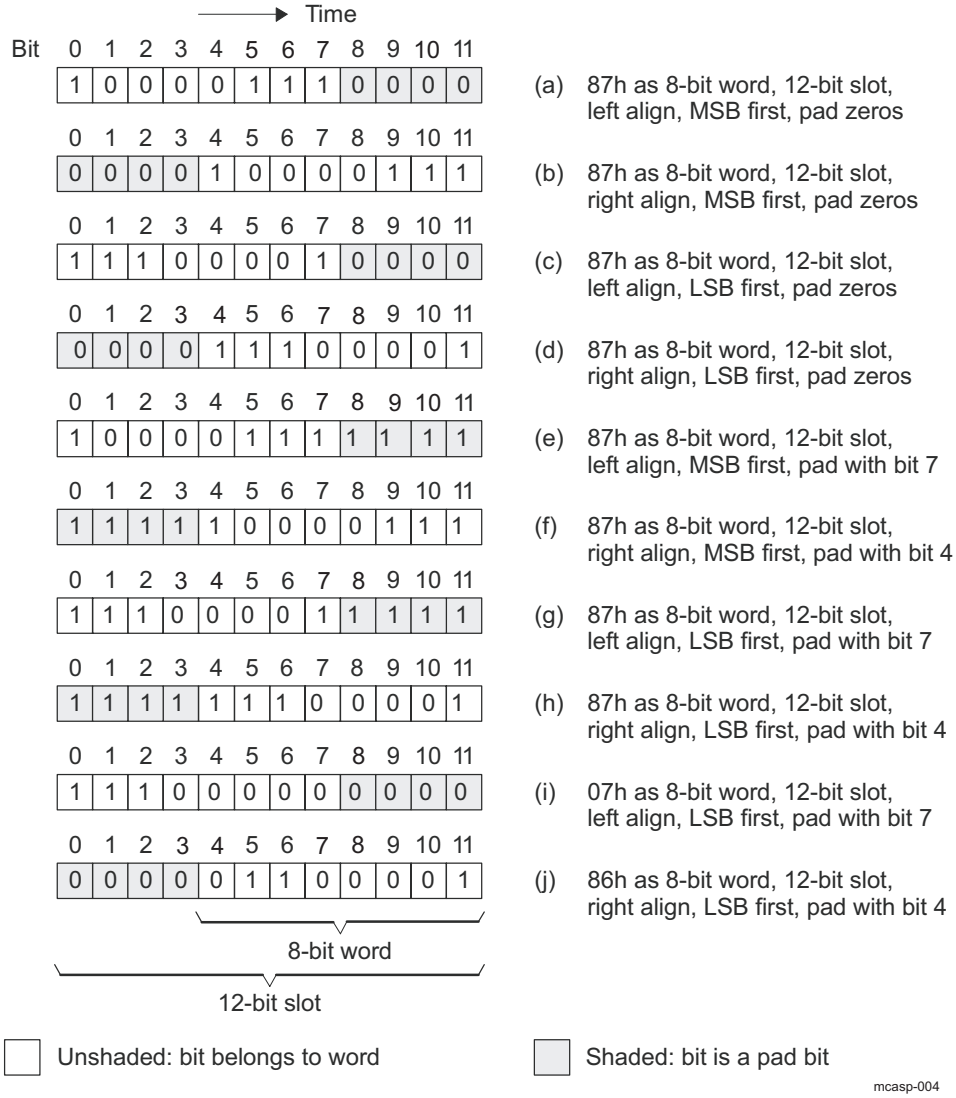
(1) b7:b0 - bits. Bits b7 to b0 form a word.

(2) P - pad bits. Bits b7 to b0, together with the 4 pad bits, form a slot.

(3) In this example, the data is transmitted MSB first, left-aligned.

mcasp-003

Figure 11-575. McASP Bit Order and Word Alignment Within a Slot Examples



- Frame

  The third basic element of a synchronous serial interface is the frame synchronization signal, also referred to as frame sync in this chapter. A frame contains one or multiple slots, as determined by the desired protocol. Figure 11-576 shows an example frame of data and the frame definitions. In operation, the transmitter uses AFSX, and the receiver – AFSR signal. Figure 11-576 does NOT specify whether the frame sync (FS) is for transmit (AFSX) or receive (AFSR) because the definitions of terms apply to both receive and transmit interfaces. In operation, each transmitter/receiver uses AFSX/AFSR as a frame synchronization signal, respectively. Optionally, the receiver can use AFSX as the frame sync when the transmitter and receiver of the McASP are configured to operate synchronously. This example shows two slots in a frame (I^2S format) and a frame-sync (FS) duration of a slot length.

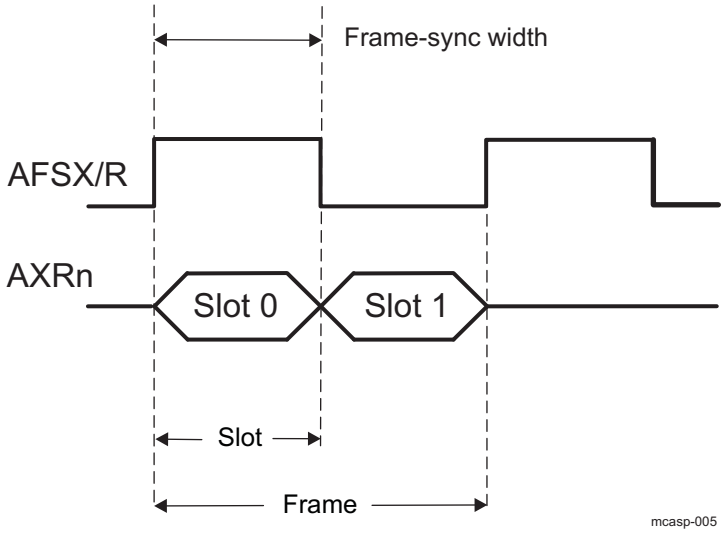
  This section shows only the generic definition of the frame sync. For more information about the frame-sync formats required for the transfer modes and protocols (TDM-mode and DIT-mode supported formats), see Section 11.9.2.2.3, *TDM Format* and Section 11.9.2.2.5, *S/PDIF-Coding Format*.

NOTE: All of the McASP serializers share the same, device pad accessible, clock and frame signals, as follows:

– AHCLKX, ACLKX and AFSX for the transmitting section

– ACLKR and AFSR for the receiving section

Figure 11-576. McASP Definition of Frame and Frame-Sync Width



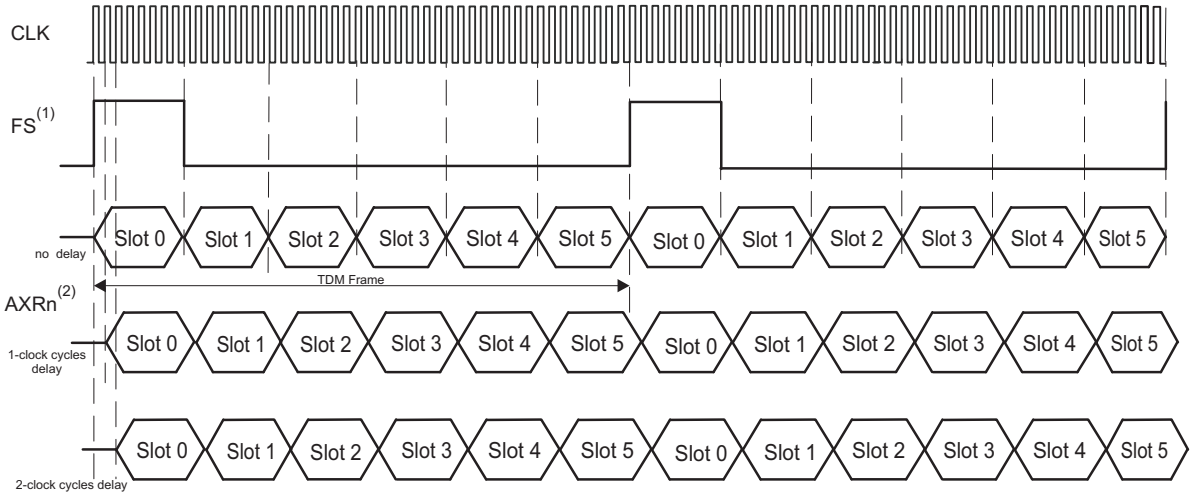
The following terms are used throughout this chapter:

- TDM: Time-division multiplexed. See Section 11.9.2.2.3, *TDM Format*, for details on the TDM protocol.
- I^2S: Inter-Integrated Sound protocol, commonly used on audio interfaces. The McASP supports the I^2S protocol as part of the TDM mode (when configured as a 2-slot frame).
- DIT: Digital audio interface transmit. The McASP supports transmitting in S/PDIF format on each AXRn data pin.
- DIR: Digital audio interface receive. The McASP does NOT natively support receiving in S/PDIF format on AXRn data pins and requires an external DIR-to-TDM or DIR-to-I^2S converter chip for a DIR-frame reception.
- Slot or time slot: For DIT/DIR format, a McASP time slot corresponds to a DIT/DIR subframe.

11.9.2.2.3 TDM Format

The TDM format is used to transfer data between the SoC and one or more analog-to-digital converter (ADC), digital-to-analog converter (DAC), or S/PDIF receiver (DIR) devices. An example for a 6-slot (channel) TDM transmission on one McASP data pin, AXRn, is illustrated on Figure 11-577.

Figure 11-577. McASP TDM Format - 6 Channel Example



(1) - Frame sync duration of 1 slot - length is shown. A single bit - duration of FS is also supported

(2) - Slot 0 of AXRn stream shown offsetted with 0-, 1- and 2-cycle delay from the frame sync respectively

mcasp-006

The TDM format uses three signals in a basic synchronous serial interface: data (AXRn), clock (CLK) and frame sync (FS). The data signal present on AXRn pin is fully synchronous to the serial clock (ACLKX or ACLKR). The data bits are grouped into words and slots (see also Section 11.9.2.2.2, *Definition of Terms*), the latter being also refered to as the "time-slots" or "channels" in TDM terminology. A frame consists of multiple time-slots. Each TDM frame is marked by the frame sync signal (AFSX or AFSR). The TDM transfer is continuous and periodic, with no delays between slots.

Within a certain frame, the last bit of slot N is followed immediately on the next serial clock with the first bit of the next slot N+1. On the boundary between two adjacent TDM-frames, the last bit of the last slot from the frame M, is followed immediately on the next clock cycle with the first bit of the first slot from the next frame M+1. For McASP, there is an option to offset the first bit of the first slot with a 0-,1- or 2-cycle delay from the frame sync signal.

The frame sync - AFSX/AFSR only marks the beginning of slot 0 and start of a new frame. Since it does NOT determine the boundaries of a slot, **there is a requirement for a connected transmitter and receiver to agree on the number of transfered bits per slot.**

In a typical audio system involving McASP module, a single TDM data frame is transfered during each sample period Ts of a data converter. The user has following choices to implement multiple channels:

- Use more data slots (on a price of higher speed serial clock) per frame transmitted/received on just one of the available McASP data pins AXRn.
- Use less number of slots per TDM frame (requires a slower serial clock), making them available on several of the McASP pins AXRn.

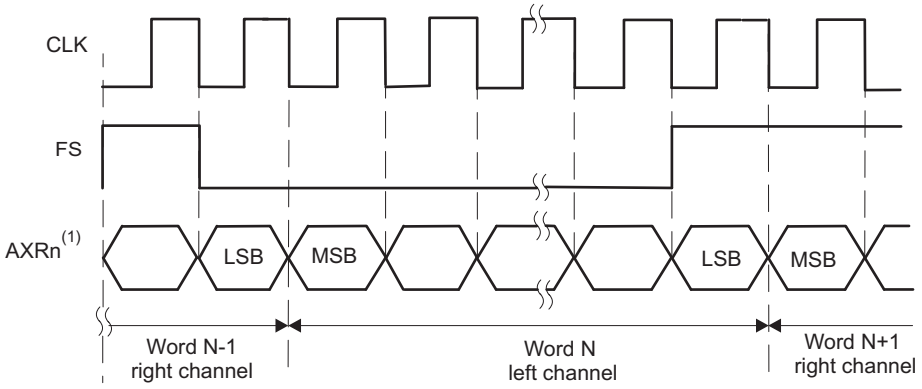
11.9.2.2.4 I²S Format

The Inter-IC Sound (I^2S) format is used extensively in audio interfaces. The TDM transfer mode of the McASP supports the I^2S format when configured for two slots per frame, with a 1-bit data delay, and falling edge of frame sync indicating start of frame.

The TDM transfer mode of the McASP supports the I^2S format when frame is configured to have 2 slots. I^2S format is specifically designed to transfer a stereo channel pair (left and right) over a single data pin AXRn. "Slots" are also commonly referred to as "audio channels". The frame width duration in the I^2S format equals size of a slot. The frame signal is also referred to as "word select" in the I^2S format.

The I^2S protocol is illustrated on Figure 11-578.

Figure 11-578. McASP I²S Format Overview



(1) - The example shows I²S data MSB-first transmission with 1-clock cycle delay between FS and data MSB

mcasp-007

11.9.2.2.5 S/PDIF Coding Format

The McASP transmitter supports the S/PDIF format with 3.3V biphase-mark encoded output. The S/PDIF format is supported by the DIT-transfer mode of the McASP. This section briefly discusses the S/PDIF coding format.

NOTE: The DIR-reception of S/PDIF format frames is NOT natively supported from the device McASP. For this purpose, an external DIR-to-TDM transfer mode adapter can be used between the remote device S/PDIF transmitter output and the McASP TDM-only compatible receiver input.

11.9.2.2.5.1 Biphase-Mark Code

In S/PDIF format, the digital signal is coded using the biphase-mark code (BMC). For each serializer transmitter n, the clock, frame, and data are embedded in only one signal - the data signal AXRn. In the BMC system, each data bit is encoded into two logical states (00, 01, 10, or 11) at the pin. These two logical states form a cell. The duration of the cell, which equals the duration of the data bit, is called a time interval. A logical 1 is represented by two transitions of the signal within a time interval, which corresponds to a cell with logical states 01 or 10. A logical 0 is represented by one transition within a time interval, which corresponds to a cell with logical states 00 or 11. In addition, the logical level at the start of a cell is inverted from the level at the end of the previous cell. Figure 11-579 and Table 11-1233 show how data is encoded to the BMC format.

As shown in Figure 11-579, the clock frequency is twice the unencoded data bit rate. In addition, the clock is always programmed to 128 × f_s, where f_s is the sample rate (see Section 11.9.2.2.5.3, *Frame Format*, for details on how this clock rate is derived based on the S/PDIF format). The device receiving in S/PDIF format can recover the clock and frame information from the BMC signal.

Figure 11-579. McASP Biphase-Mark Code

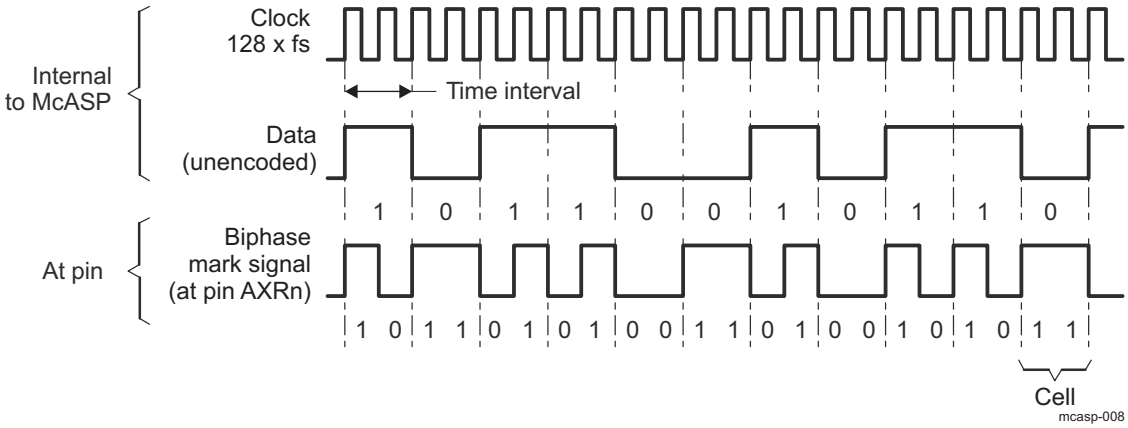


Table 11-1233. McASP Biphase-Mark Encoder

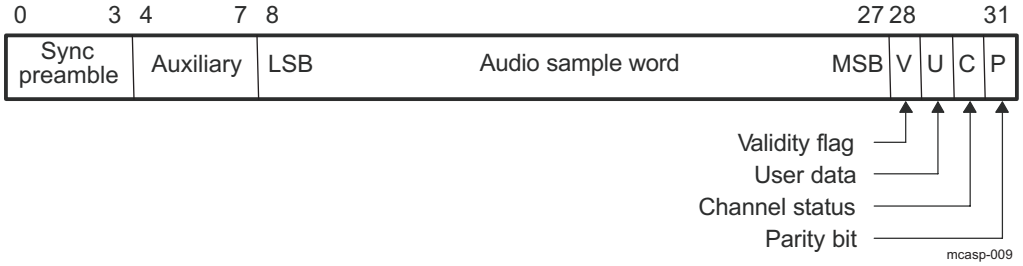
Data (Unencoded)	Previous State at Pin AXRn	BMC-Encoded Cell Output at Pin AXRn
0	0	11
0	1	00
1	0	10
1	1	01

11.9.2.2.5.2 S/PDIF Subframe Format

Every audio sample transmitted in a subframe consists of 32 S/PDIF time intervals (or cells), numbered 0 to 31. Figure 11-580 shows a subframe.

- Time intervals 0–3 carry one of the three permitted preambles to signify the type of audio sample in the current subframe. The preamble is not encoded in BMC format, and therefore the preamble code can contain more than two consecutive 0 or 1 logical states in a row, see Table 11-1234.
- Time intervals 4–27 carry the audio sample word in linear 2s-complement representation. The MSB is carried by time interval 27. When a 24-bit coding range is used, the LSB is in time interval 4. When a 20-bit coding range is used, time intervals 8-27 carry the audio sample word with the LSB in time interval 8. Time intervals 4–7 may be used for other applications and are designated auxiliary sample bits.
- If the source provides fewer bits than the interface allows (20 or 24), the unused LSBs are set to logical 0. For a nonlinear PCM audio application or a data application, the main data field can carry any other information.
- Time interval 28 carries the validity bit (V) associated with the main data field in the subframe.
- Time interval 29 carries the user data channel (U) associated with the main data field in the subframe.
- Time interval 30 carries the channel status information (C) associated with the main data field in the subframe. The channel status indicates if the data in the subframe is digital audio or some other type of data.
- Time interval 31 carries a parity bit (P) such that time intervals 4–31 carry an even number of 1s and an even number of 0s (even parity). As listed in Table 11-1234, the preambles (time intervals 0–3) are also defined with even parity.

Figure 11-580. McASP S/PDIF Subframe Format



As listed in Table 11-1234, the McASP DIT generates only one polarity of preambles, and it assumes the previous logical state is 0. This is because the McASP assures an even-polarity encoding scheme when transmitting in DIT mode. If an underrun condition occurs, the DIT resynchronizes to the correct logic level on the AXRn pin before continuing with the next transmission.

Table 11-1234. McASP Preamble Codes

Preamble Code[1]	Previous Logical State	Logical States on pin AXRn[2]	Description
B (or Z)	0	1110 1000	Start of a block and subframe 1
M (or X)	0	1110 0010	Subframe 1
W (or Y)	0	1110 0100	Subframe 2

(1) Historically, preamble codes are referred to as B, M, and W. For use in professional applications, preambles are referred to as Z, X, and Y, respectively.

(2) The preamble is not BMC-encoded. Each logical state is synchronized to the serial clock. These eight logical states make up time slots (cells) 0 to 3 in the S/PDIF stream.

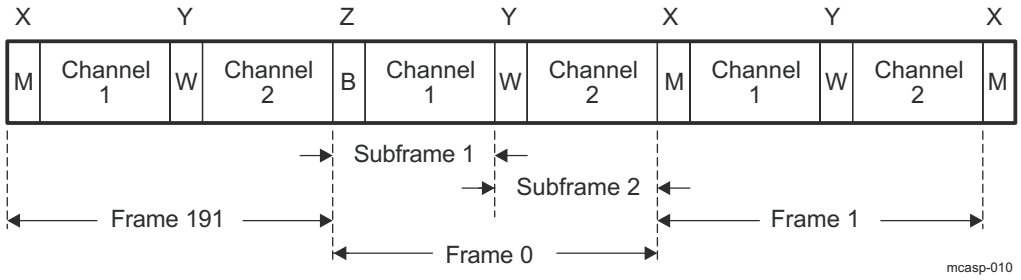
11.9.2.2.5.3 Frame Format

An S/PDIF frame is composed of two subframes (see Figure 11-581). For linear coded audio applications, the rate of frame transmission normally corresponds exactly to the source sampling frequency f_s. The S/PDIF format clock rate is therefore 128 × f_s (128 = 32 cells per subframe × 2 clocks per cell × 2 subframes per sample). For example, for an S/PDIF stream at a 192-kHz sampling frequency, the serial clock is 128 × 192 kHz = 24.58 MHZ.

In 2-channel operation mode, the samples taken from both channels are transmitted by time multiplexing in consecutive subframes. Both subframes contain valid data (cell 28 validity bits for A- and B- channels, both set to '0'). The first subframe (left or A channel in stereophonic operation and primary channel in monophonic operation) normally starts with preamble M. However, the preamble of the first subframe changes to preamble B once every 192 frames to identify the start of the block structure used to organize the channel status information. The second subframe (right or B channel in stereophonic operation and secondary channel in monophonic operation) always starts with preamble W.

In single-channel operation mode in a professional application, the frame format is the same as in the 2-channel mode. Data is carried in the first subframe and may be duplicated in the second subframe. If the second subframe is not carrying duplicate data, cell 28 (validity bit) is set to logical 1.

Figure 11-581. McASP S/PDIF Frame Format



11.9.3 McASP Integration

This section describes McASP modules integration in the device, including information about clocks, resets, and hardware requests.

A McASP module includes the following features:

- One DMA request for transmit event
- One DMA request for receive event
- One interrupt request (IRQ) for transmit
- One interrupt request (IRQ) for receive

Figure 11-582 shows McASP integration.

Figure 11-582. McASP Integration

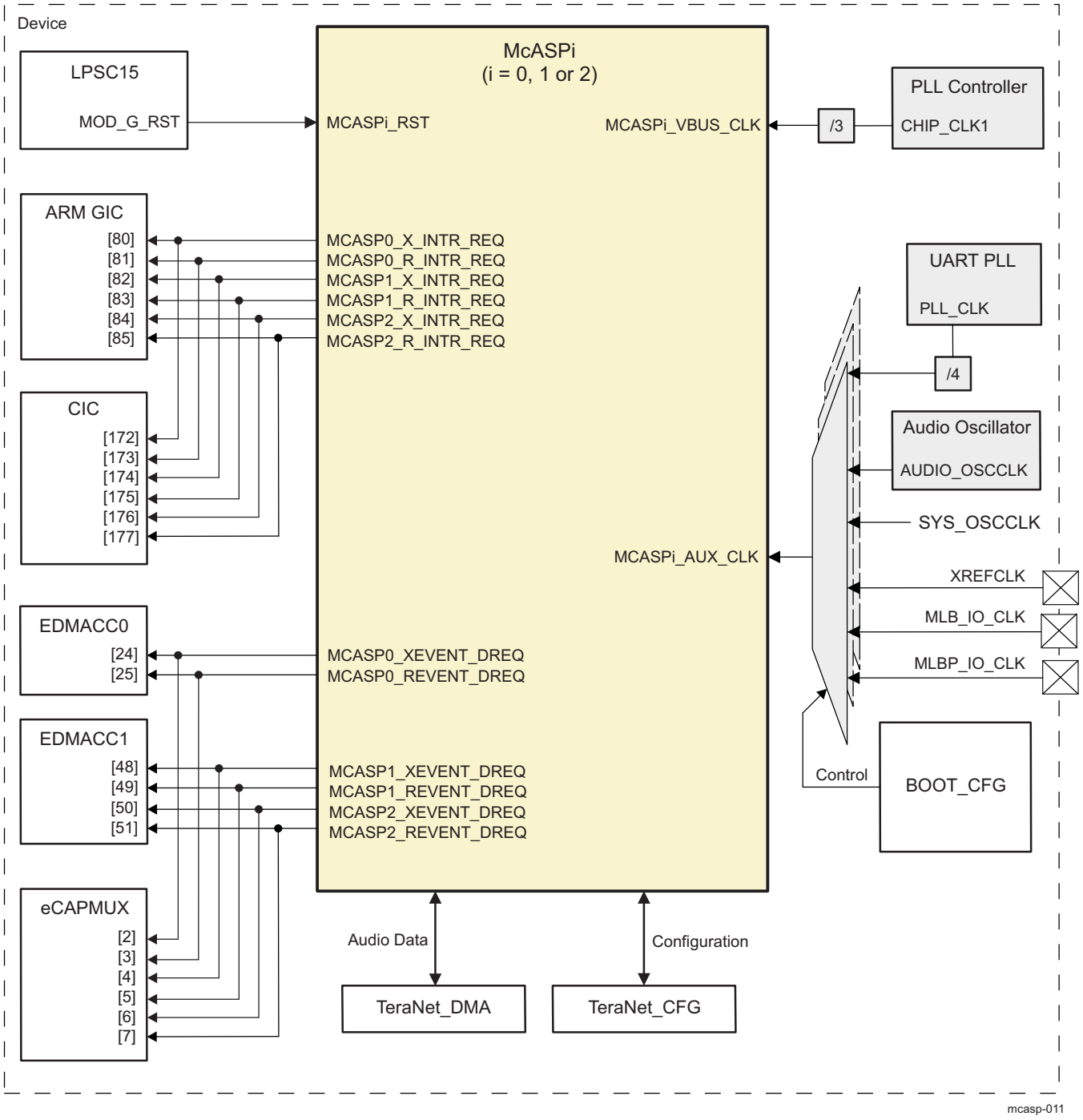


Table 11-1235 through Table 11-1237 summarize the integration of the McASP modules in the device.

Table 11-1235. McASP Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
McASP0 McASP1 McASP2	PD5	LPSC15	TeraNet_DMA (for data traffic) TeraNet_CFG (for configuration and data traffic)

Table 11-1236. McASP Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Destination Signal Description**
McASPi i = 0, 1 or 2	MCASPi_VBUS_CLK	CHIP_CLK1 / 3	PLL Controller	VBUS and functional clock. This clock must be 2x the rate of IO clock. For this SoC, this clock must be at least 150MHz.
	MCASPi_AUX_CLK	PLL_CLK / 5	UART PLL	Auxillary clock AUXCLK. Output of multiplexer (one per McASP module), see Figure 11-582, *McASP Integration*. Multiplexers control is provided via MCASPi_AUXCLK_SEL bit-fields of BOOTCFG_SERIALPORT_CLKCTL register. Audio clocks are derived from this clock.
		AUDIO_OSCCLK	Audio Oscillator	
		XREFCLK	XREFCLK pin	
		MLB_IO_CLK	MLB_CLK pin	
		MLBP_IO_CLK	MLBP_CLK_P/N pins	
		SYS_OSCCLK	SYS_CLK mux	
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Destination Signal Description**
McASPi i = 0, 1 or 2	MCASPi_RST	MOD_G_RST	LPSC15	Reset signal

Table 11-1237. McASP Hardware Requests

Interrupt Requests				
Module Instance	**Event Name**	**Mapped To Input Event [Number]**		**Description**
		ARM GIC	**CIC**	
McASP0	MCASP0_X_INTR_REQ	[80]	[172]	McASP0 Transmit event interrupt
	MCASP0_R_INTR_REQ	[81]	[173]	McASP0 Receive event interrupt
McASP1	MCASP1_X_INTR_REQ	[82]	[174]	McASP1 Transmit event interrupt
	MCASP1_R_INTR_REQ	[83]	[175]	McASP1 Receive event interrupt
McASP2	MCASP2_X_INTR_REQ	[84]	[176]	McASP2 Transmit event interrupt
	MCASP2_R_INTR_REQ	[85]	[177]	McASP2 Receive event interrupt

DMA Requests					
Module Instance	**Event Name**	**Mapped To Input Event [Number]**			**Description**
		eCAPMUX	**EDMACC0**	**EDMACC1**	
McASP0	MCASP0_XEVENT_DREQ	[2]	[24]	-	McASP0 Transmit event (pulse)
	MCASP0_REVENT_DREQ	[3]	[25]	-	McASP0 Receive event (pulse)
McASP1	MCASP1_XEVENT_DREQ	[4]	-	[48]	McASP1 Transmit event (pulse)
	MCASP1_REVENT_DREQ	[5]	-	[49]	McASP1 Receive event (pulse)
McASP2	MCASP2_XEVENT_DREQ	[6]	-	[50]	McASP2 Transmit event (pulse)
	MCASP2_REVENT_DREQ	[7]	-	[51]	McASP2 Receive event (pulse)

NOTE: For more information on device power, reset, and clock management, see the corresponding sections within Chapter 5, *Device Configuration*.

For more information on device BOOT_CFG, see Section 5.1, *Control Module (BOOT_CFG)*.

For more information on device interconnects, see Section 3.1, *Interconnect*.

For more information on device interrupt controllers, see Chapter 9, *Interrupts*.

For more information on device DMA controllers, see Chapter 10, *Enhanced Direct Memory Access (EDMA) Controller*.

For more information on device eCAPMUX, see Section 5.1.3.1.7, *Event Mux Control Registers*.

NOTE: For a description of the interrupt sources, see Section 11.9.4.12, *McASP Events and Interrupt Requests*.

For a description of the DMA sources, see Section 11.9.4.13, *McASP DMA Requests*.

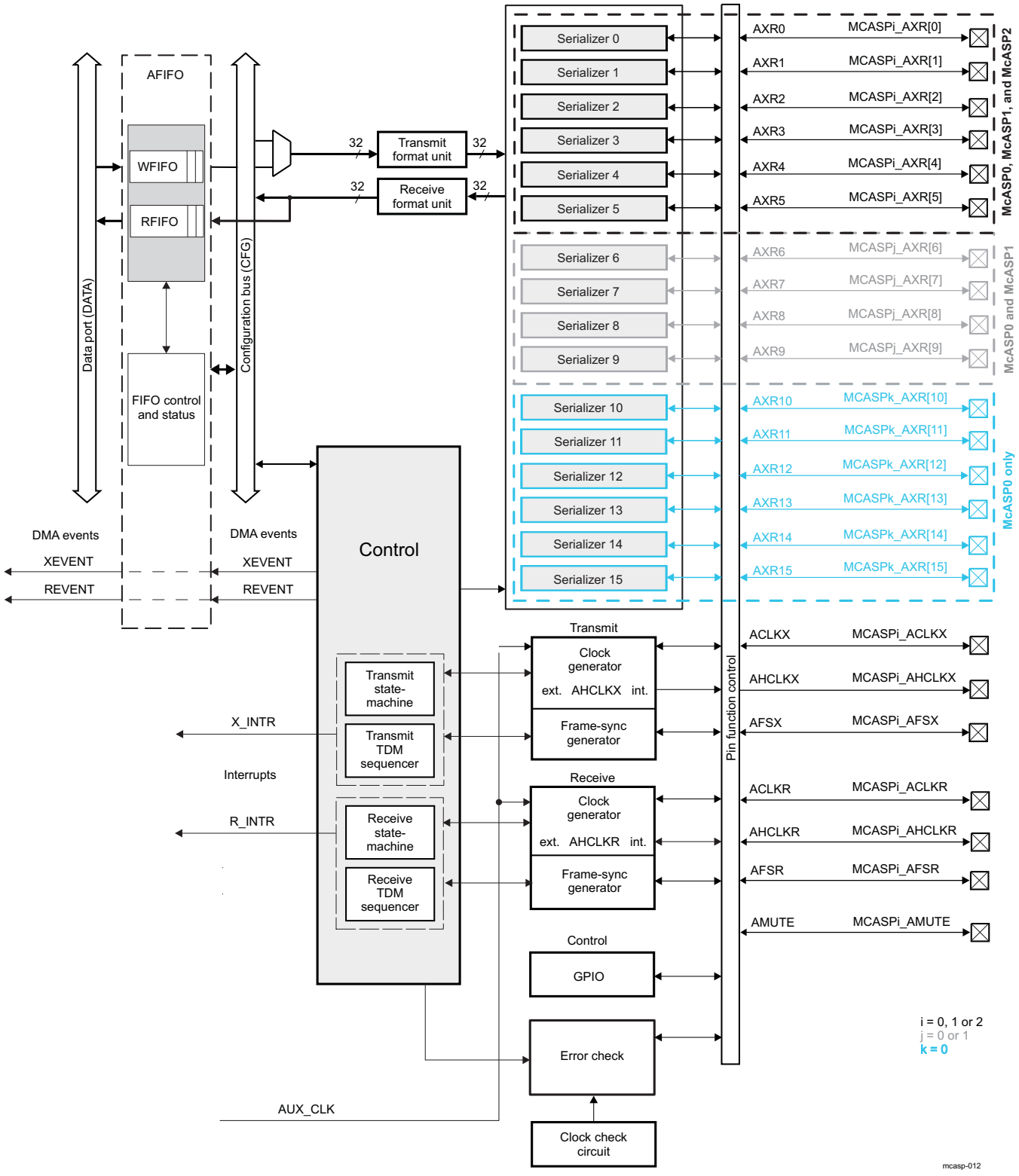
11.9.4 McASP Functional Description

All three McASP modules are functionally indentical, therefore in the text below a single instance of McASP is described. Differences for each McASP module will be highlighted in the text.

11.9.4.1 McASP Block Diagram

Figure 11-583 shows the major blocks of the McASP module. McASP0 has 16 serializers, McASP1 has 10 serializers, and McASP2 has 6 serializers. The serializers share a clock and frame-sync generator, format unit, and error-checking logic independently for the receive and transmit part.

Figure 11-583. McASP Module Block Diagram



NOTE: The internal and external clocks mentioned in this section are with respect to clock and frame-sync generator modules.

11.9.4.2 McASP Clocks and Frame-Sync Configurations

There are three scenarios to provide clock source signals for the Tx part and four scenarios for the Rx part of the McASP serializers. The first three scenarios are identical between the Tx and Rx parts of the McASP. They feature an asynchronous operation between receiver and transmitter channels using independent Tx and Rx bit rate clock sources (either internal or external).

In the first scenario, the transmit - XCLK and receive - RCLK serial clocks (clock at the bit rate) are generated internally by passing through a couple of clock dividers off the internal functional clock source (AUXCLK). In this case, the bit rate clock is generated internally and is driven out on the pin ACLKX for the Tx part and pin ACLKR for the Rx part, respectively. An internally generated high-frequency clock can be optionally driven out onto the AHCLKX pin for the Tx part to serve as a reference clock for other components in the system.

In the second scenario, an external for the device clock, is passed on the ACLKX (for the Tx part) and ACLKR (for the Rx part) pins which are configured as inputs. In this case the Rx- /Tx- high-speed clock logic is bypassed for the XCLK/RCLK generation.

In the third (mixed) scenario, an externally driven (master) high-frequency clock is applied on the AHCLKX (for the Tx part) pin, which is configured as input. In this case the AHCLKX clock frequency can be divided down via programming the ACLKX associated divider to produce the necessary bit rate clock. The high-speed clock divider can NOT be used.

In the fourth clock generation scenario the bit rate clock for McASP receivers - RCLK is derived from the bit rate clock of the McASP transmitters - XCLK for a synchronous operation between transmitters and receivers. Hence, the whole receiver clock generator logic is bypassed.

A typical role of the McASP frame sync signal is to carry the left/right clock (LRCLK) signal when transmitting and receiving stereo data.

For an asynchronous operation, the AFSX (Tx part) and AFSR (Rx part) frame synchronization signals can be sourced internally or delivered externally independently for the Tx and Rx channels. During synchronous operation the receive frame sync - AFSR signal is derived from the transmit frame sync - AFSX signal. A synchronous and asynchronous mode applies to bit rate clock and frame sync signals at the same time.

11.9.4.2.1 Transmit Clock

The transmit high-speed and transmit clock configuration is controlled by the following registers:

- MCASP_ACLKXCTL
- MCASP_AHCLKXCTL

In case the transmit bit clock, ACLKX, is generated internally, the MCASP_ACLKXCTL[5] CLKXM bit must be set to 1. Thus, the clock is divided down by a programmable bit clock divider (the MCASP_ACLKXCTL[4-0] CLKXDIV bit field) from the source signal.

If the transmit high-frequency master clock, AHCLKX, is also sourced internally (that is first scenario described in Section 11.9.4.2, *McASP Clocks and Frame-Sync Configurations*), the MCASP_AHCLKXCTL[15] HCLKXM bit must be set to 1. Thus, the clock is divided down by a programmable high-clock divider (the MCASP_AHCLKXCTL[11-0] HCLKXDIV bit field) from the McASP internal clock source AUXCLK.

Internally, the McASP always shifts transmit data at the rising edge of the internal transmit clock - XCLK, (see Figure 11-584). The CLKXP mux determines if ACLKX needs to be inverted to become XCLK. If MCASP_ACLKXCTL[7] CLKXP = 0, the CLKXP mux directly passes ACLKX signal to XCLK. As a result, the McASP shifts transmit data at the rising edge of ACLKX. If MCASP_ACLKXCTL[7] CLKXP = 1, the CLKX mux passes the inverted version of ACLKX to XCLK. As a result, the McASP shifts transmit data at the falling edge of ACLKX.

It can be seen in Figure 11-584, that XCLK is propagated to the Rx clock logic, to allow an internally synchronous operation between McASP transmitters and receivers. This is used for example in the McASP loopback mode.

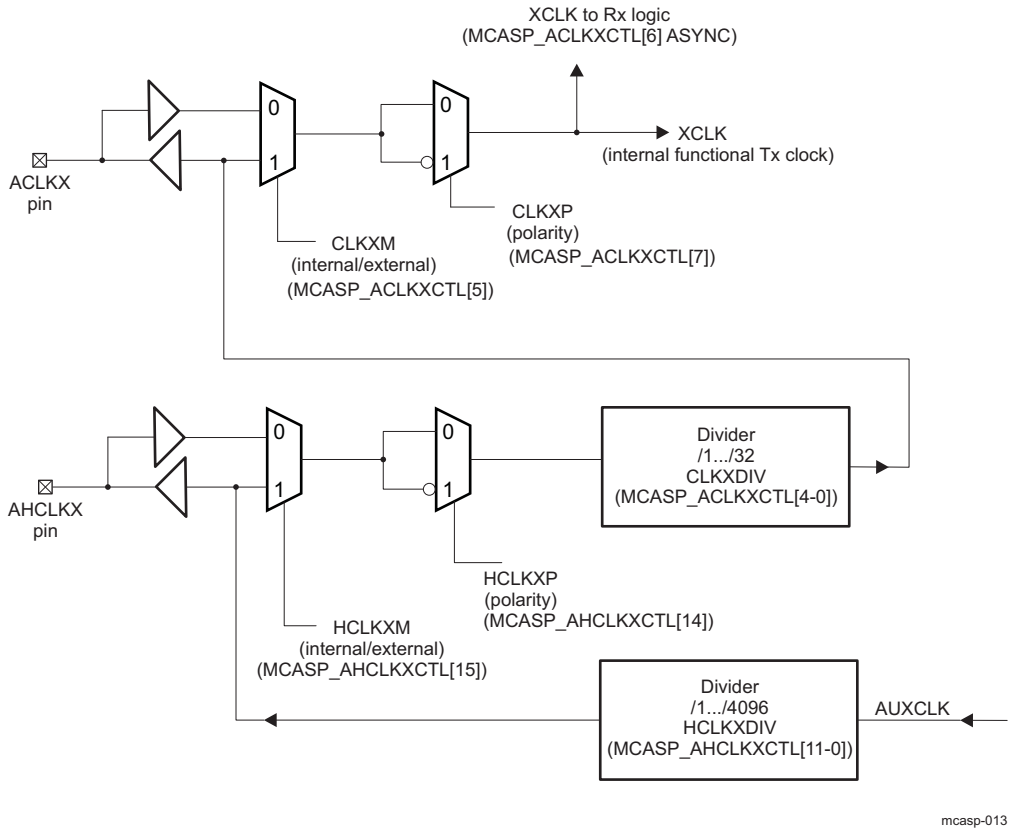
NOTE: The polarity of ACLKX can be controlled in MCASP_ACLKXCTL[7] CLKXP, regardless of ACLKX signal being internally or externally sourced.

In addition, there is an option to invert polarity of the AHCLKX master high speed clock via writing the MCASP_AHCLKXCTL[14] HCLKXP bit.

NOTE: In a similar way, the polarity of AHCLKX clock can be controlled in MCASP_AHCLKXCTL[14] HCLKXP, regardless of the AHCLKX signal being internally or externally sourced.

Figure 11-584 is the block diagram of the transmit clock generator.

Figure 11-584. McASP Transmit Clock Generator Block Diagram



NOTE: For more details on McASP integration, see Section 11.9.2, *McASP Environment*, and Section 11.9.3, *McASP Integration*.

11.9.4.2.2 Receive Clock

The McASP receive clock generator is built on a very similar to the transmit clock generator (but independent) circuit.

The receive clock configuration is controlled by the following registers:

- MCASP_ACLKRCTL
- MCASP_AHCLKRCTL

In case the receive bit clock, ACLKR, is generated internally (but asynchronously to XCLK), the MCASP_ACLKRCTL[5] CLKRM bit must be set to 1. Thus, the clock is divided down by a programmable bit clock divider (the MCASP_ACLKRCTL[4-0] CLKRDIV bit field) from the source signal.

If the receive high-frequency master clock, AHCLKR, is also sourced internally (that is, first scenario described in Section 11.9.4.2, *McASP Clocks and Frame-Sync Configurations*), then the MCASP_AHCLKRCTL[15] HCLKRM bit must be set to 1. Thus, the clock is divided down by a programmable high-clock divider (the MCASP_AHCLKRCTL[11-0] HCLKRDIV bit field) from the McASP internal clock source AUXCLK.

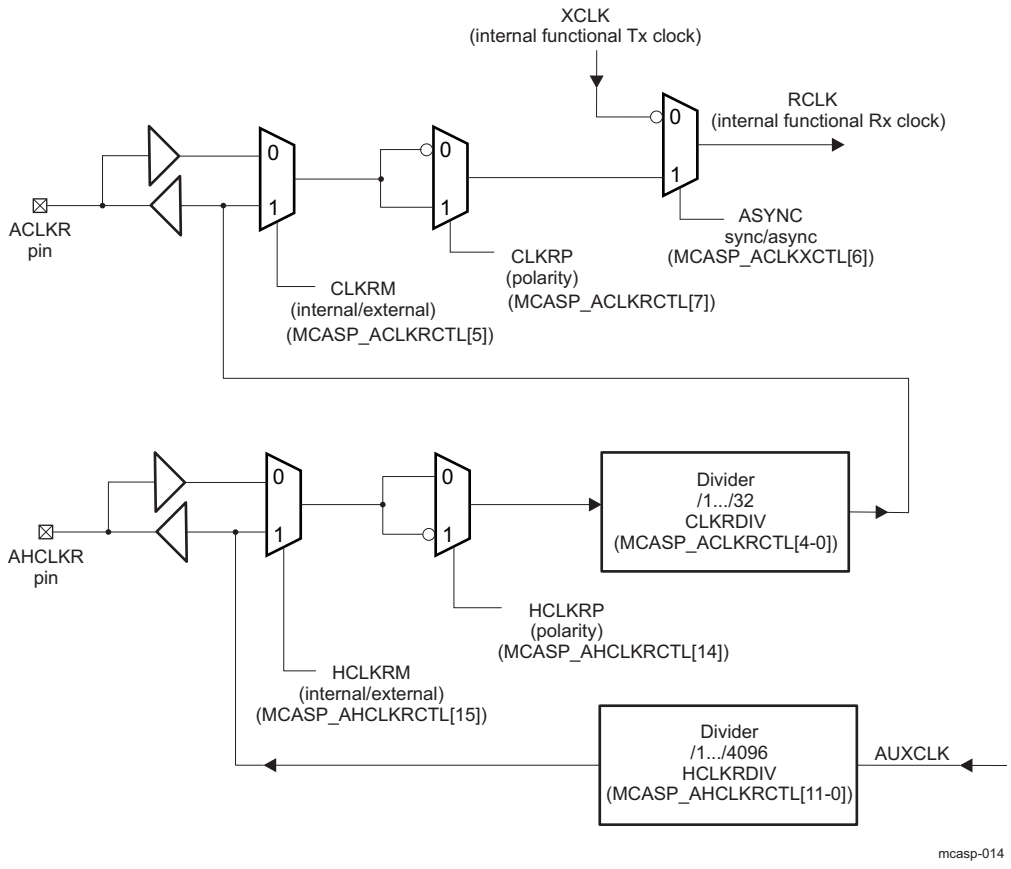
NOTE: The polarity of ACLKR can be controlled in MCASP_ACLKRCTL[7] CLKRP, regardless of ACLKR signal being internally or externally sourced.

In a similar way, the polarity of AHCLKR clock can be controlled in MCASP_AHCLKRCTL[14] HCLKRP, regardless of the AHCLKR signal being internally or externally sourced.

There is an option for the McASP receiver to be configured to operate synchronously to the ACLKX and AFSX signals. The XCLK output of the Tx Clock generator (see Figure 11-584 and Figure 11-585) becomes source of the receive clock (RCLK output), when the MCASP_ACLKXCTL[6] ASYNC bit in the transmit clock control register is set to '0b0'. For more information, refer to Section 11.9.4.2.4.

Figure 11-585 is the block diagram of the receive clock generator.

Figure 11-585. McASP Receive Clock Generator Block Diagram

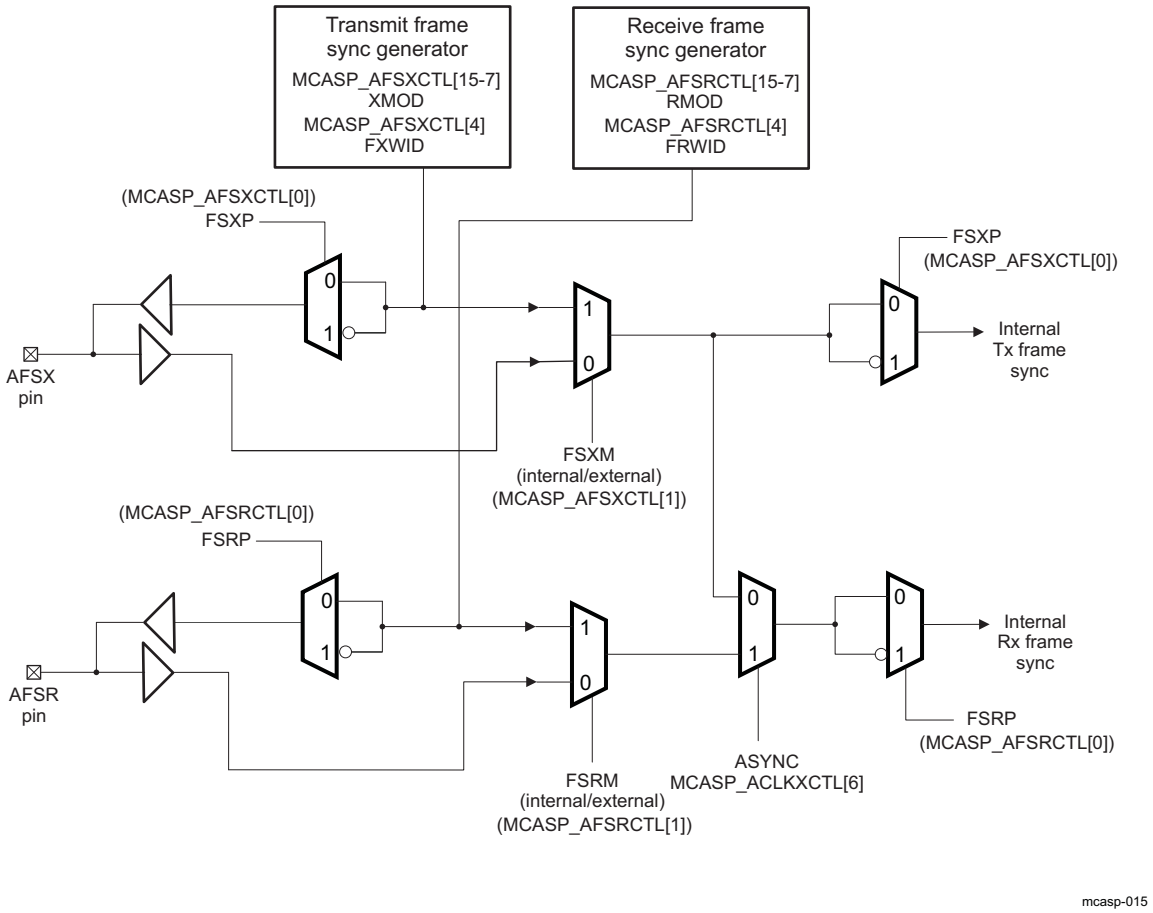


NOTE: For more details on McASP integration, see Section 11.9.2, *McASP Environment*, and Section 11.9.3, *McASP Integration*.

11.9.4.2.3 Frame-Sync Generator

There are two different modes for frame sync: burst and TDM. The McASP frame sync generator logic is illustrated in Figure 11-586. I/O buffers are not part of the McASP module, and they are not shown in the figure.

Figure 11-586. McASP Frame Sync Generator Block Diagram



NOTE: For more details on McASP integration, see Section 11.9.2, *McASP Environment*, and Section 11.9.3, *McASP Integration*.

For the transmit logic, the following frame-sync generator configurations can be selected:

- Internally/externally generated frame-sync via configuring bit MCASP_AFSXCTL[1] FSXM
- Frame-sync polarity: Rising edge or falling edge via configuring bit MCASP_AFSXCTL[0] FSXP
- Frame-sync width: "single bit" or "single word" via configuring bit MCASP_AFSXCTL[4] FXWID
- Frame sync mode - the appropriate frame sync generation pattern for the selected transfer mode is defined in the bitfield MCASP_AFSXCTL[15-7] XMOD, as follows:
  - For DIT mode (384 slots) - MCASP_AFSXCTL[15-7] XMOD = 0x180
  - For I^2S mode (2 TDM slots) - MCASP_AFSXCTL[15-7] XMOD = 0x2
  - For TDM mode (from 3 to 32 TDM slots) - MCASP_AFSXCTL[15-7] XMOD set in range 0x3 - 0x20
- Bit delay: 0, 1, or 2 cycles before the first data bit. This delay is defined in MCASP_XFMT[17-16] XDATDLY

For the receive logic, the following frame-sync generator configurations can be selected:

- Internally/externally generated frame-sync via configuring bit MCASP_AFSRCTL[1] FSRM

- Frame-sync polarity: Rising edge or falling edge via configuring bit MCASP_AFSRCTL[0] FSRP
- Frame-sync width: "single bit" or "single word" via configuring bit MCASP_AFSRCTL[4] FRWID
- Frame sync mode - the appropriate frame sync generation pattern for the selected transfer mode is defined in the bitfield MCASP_AFSRCTL[15-7] RMOD, as follows:
  - For I^2S mode (2 TDM slots) - MCASP_AFSRCTL[15-7] RMOD = 0x2
  - For TDM mode (from 3 to 32 TDM slots) - MCASP_AFSRCTL[15-7] RMOD set in range 0x3–0x20
  - For the special 384-slot TDM mode - MCASP_AFSRCTL[15-7] RMOD=0x180
- Bit delay: 0, 1, or 2 cycles before the first data bit. This delay is defined in MCASP_RFMT[17-16] RDATDLY
- Selecting the source (AFSX or AFSR) of receiver internal frame synchronization. This is done in the same bit - MCASP_ACLKXCTL[6] ASYNC, used to define the receiver internal clock source. For more details, refer to Section 11.9.4.2.4, *McASP Synchronous and Asynchronous Transmit and Receive Operations*.

Regardless of the AFSX/AFSR being internally generated or externally sourced, the polarity of AFSX/AFSR is determined by FSXP/FSRP, respectively, to be either rising or falling edge. If FSXP/FSRP = 0, the frame sync polarity is rising edge. If FSXP/FSRP = 1, the frame sync polarity is falling edge.

NOTE: Certain restrictions apply to the receive and transmit logic settings, when MCASP_ACLKXCTL[6] ASYNC is set to 0b0. They are described in Section 11.9.4.2.4, *McASP Synchronous and Asynchronous Transmit and Receive Operations*.

11.9.4.2.4 McASP Synchronous and Asynchronous Transmit and Receive Operations

Synchronous Transmit and Receive Operations

When MCASP_ACLKXCTL[6] ASYNC is written to 0b0, the transmit and receive sections operate synchronously to the transmit section clock and transmit frame sync signals.

Though Rx section may have a different data format, it has to be configured to have the same slot size than the transmit section one. As shown in Figure 11-585, with the ASYNC bit set to 0b0, the RCLK becomes an inverted version of the transmit clock generator XCLK output.

When MCASP_ACLKXCTL[6] ASYNC = 0b0, both Rx and Tx sections use the same clock and frame sync signals. For this reason, they must be aligned on the following settings:

- MCASP_DITCTL[0] DITEN = 0 (that is, transmission in TDM mode is enabled)
- The total number of bits per frame must be the same (that is, RSSZ * RMOD product value must equal XSSZ * XMOD product value)
- The settings in MCASP_ACLKRCTL are NOT considered
- FSXM must match FSRM
- FXWID must match FRWID

For all other settings, the transmit and receive sections may be programmed independently.

Asynchronous Transmit and Receive Operations -

When MCASP_ACLKXCTL[6] ASYNC = 0b1, Tx and Rx operate independently from each other with separate clock and frame sync signals.

NOTE: Synchronous transmit and receive operations are allowed only in the McASP TDM (I^2S) mode (that is when MCASP_DITCTL[0] DITEN=0b0).

11.9.4.3 McASP Serializers

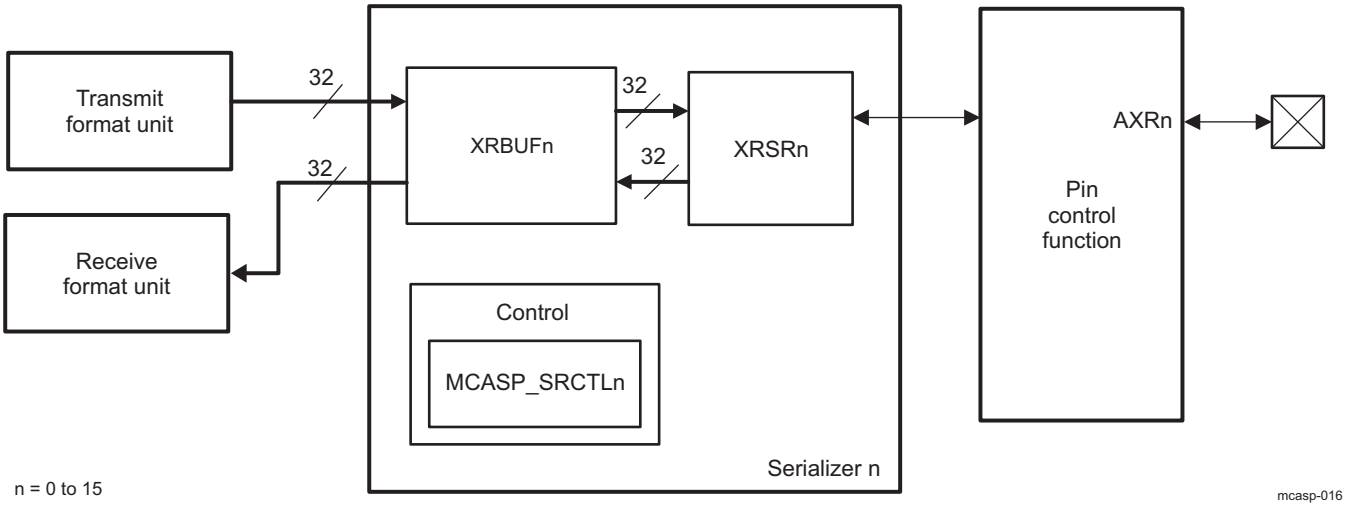
The McASP serializers shift serial data in (Rx) and out (Tx) of the McASP. A given serializer n (n = 0 to 15) consists of a shift register (XRSRn) with a single-entry data buffer XRBUFn used either for transmitting (write accessible in register MCASP_XBUF0 through MCASP_XBUF15) or for receiving (read accessible in register MCASP_RBUF0 through MCASP_RBUF15) data. In addition, each serializer has a dedicated control register (MCASP_SRCTL0 through MCASP_SRCTL15) and a serial bidirectional data pin - AXRn. The register MCASP_SRCTL0 through MCASP_SRCTL15 allows n-th serializer to be configured as a transmitter, receiver, or as inactive. There are transmit and receive data formatting units to support data alignment options of the McASP which are shared between all Tx and Rx serializers, respectively.

A given serializer XRSRn shifter configured as a receiver in MCASP_SRCTL0 through MCASP_SRCTL15, shifts in data through McASP corresponding device level bidirectional data pad AXRn. A given serializer XRSRn shifter configured as a transmitter in MCASP_SRCTL0 through MCASP_SRCTL15, shifts out data on McASP corresponding device level bidirectional data pad AXRn.

The serializer is clocked from the transmit section clock (ACLKX signal), if configured to transmit, or clocked from the receive section clock (ACLKR signal), if configured to receive. A serializer configured to transmit and receive operates in lockstep, which means that for McASP there are at most a couple of zones, one for transmit and one for receive.

Figure 11-587 is the serializer block diagram.

Figure 11-587. McASP Individual Serializer and Connections



Transmission on the n-th serializer (where n = 0 to 15 for McASP0, and n = 0 to 9 for McASP1, and n = 0 to 5 for McASP2) is performed as follows: the CPU services the McASP by writing data into the register MCASP_XBUF0 through MCASP_XBUF15, which is an alias of the serializer data buffer - XRBUFn for transmit function. The data automatically passes through the transmit format unit before reaching the XRBUFn register in the serializer. The data is then copied from the XRBUFn to XRSRn and shifted out from AXRn synchronously to the serial clock.

Reception from the n-th serializer (where n = 0 to 15 for McASP0, and n = 0 to 9 for McASP1, and n = 0 to 5 for McASP2) is performed as follows: the data is shifted into the McASP XRSRn serializer register through the AXRn pin, bit by bit. Once the entire slot becomes available within the XRSRn shift register, the data is copied into the serializer data buffer - XRBUFn, and can be accessed in the MCASP_RBUF0 through MCASP_RBUF15 register, which is an alias of the serializer data buffer - XRBUFn for receive function. When CPU reads data from this register, the McASP passes the data through the receive format unit, hence it returns formatted data to the CPU.

Serializer controls:

A serializer n is configured as inactive via setting bitfield MCASP_SRCTL0 through MCASP_SRCTL15[1-0] SRMOD to 0x0.

For a transmitting serializer, the MCASP_SRCTL0 through MCASP_SRCTL15[3-2] DISMOD bitfield, defines the AXRn pin output state, during inactive slots (HIGH, LOW or HiZ).

Transmit function for the n-th serializer is selected via setting bitfield MCASP_SRCTL0 through MCASP_SRCTL15[1-0] SRMOD to 0x1.

Receive function for the n-th serializer is selected via setting bitfield MCASP_SRCTL0 through MCASP_SRCTL15[1-0] SRMOD to 0x2.

In the DIT-transmission mode (that is S/PDIF format data transmission): in addition to the data, the serializer shifts out other DIT-specific information accordingly (preamble, user data, and so forth). For more information, see Section 11.9.2.2.5, *S/PDIF Coding Format*.

11.9.4.4 McASP Format Units

The McASP has one transmit data formatting unit and one receive data formatting unit, shared between the device McASP serializers. These units automatically remap the data bits within the transmitted or received words between a natural format for the CPU (such as a Q31 representation) and the required format for the external serial device (for example I^2S format). During the remapping process, the format unit can also mask off certain bits.

Since all transmitters share the same data formatting unit, the McASP only supports one transmit format at a time. For example, the McASP does NOT transmit in "I^2S format" on serializer 0, while transmitting "Left Justified" on serializer 1. Likewise, the receiver section of the McASP only supports one data format at a time, and this format applies to all receiving serializers.

NOTE: The McASP can transmit in one format while receiving in a completely different format.

The bit mask and pad stage of each of Tx and Rx format units includes a full 32-bit mask register, allowing selected individual bits to either pass through the stage unchanged, or be masked off. The bit mask and pad then pad the value of the masked off bits by inserting either a 0, a 1, or one of the original 32 bits as the pad value. The last option allows for sign-extension when the sign bit is selected to pad the remaining bits. The rotate right stage performs bitwise rotation by a multiple of 4 bits (between 0 and 28 bits), programmable by the MCASP_RFMT/MCASP_XFMT register. Note that this is a rotation process, not a shifting process, so bit 0 gets shifted back into bit 31 during the rotation. The bit order – reversal stage either passes all 32 bits directly through, or swaps them. This allows for either MSB or LSB first data formats. If bit order reversal is not enabled, then the McASP will naturally transmit and receive in an LSB first order.

The RDATDLY/XDATDLY bits in the MCASP_RFMT/MCASP_XFMT registers, respectively, also determine the data format. For example, the difference between I^2S format and left-justified is determined by the delay between the frame sync edge and the first data bit of a given time slot. For I^2S format, RDATDLY/XDATDLY should be set to a 1-bit delay, whereas for left-justified format, it should be set to a 0-bit delay. The combination of all the options in MCASP_RFMT/MCASP_XFMT registers means that the McASP supports a wide variety of data formats, both on the serial data lines, and in the internal CPU (ARMSS or DSP) data representation.

11.9.4.4.1 Transmit Format Unit

The McASP transmit formatting unit consists of three stages:

- Bit mask (masks off bits)
- Rotate right (aligns data within word)
- Bit reversal (selects between MSB-first or LSB-first)

Figure 11-588 shows the transmit formatting unit.

The McASP transmitter supports serial formats of:

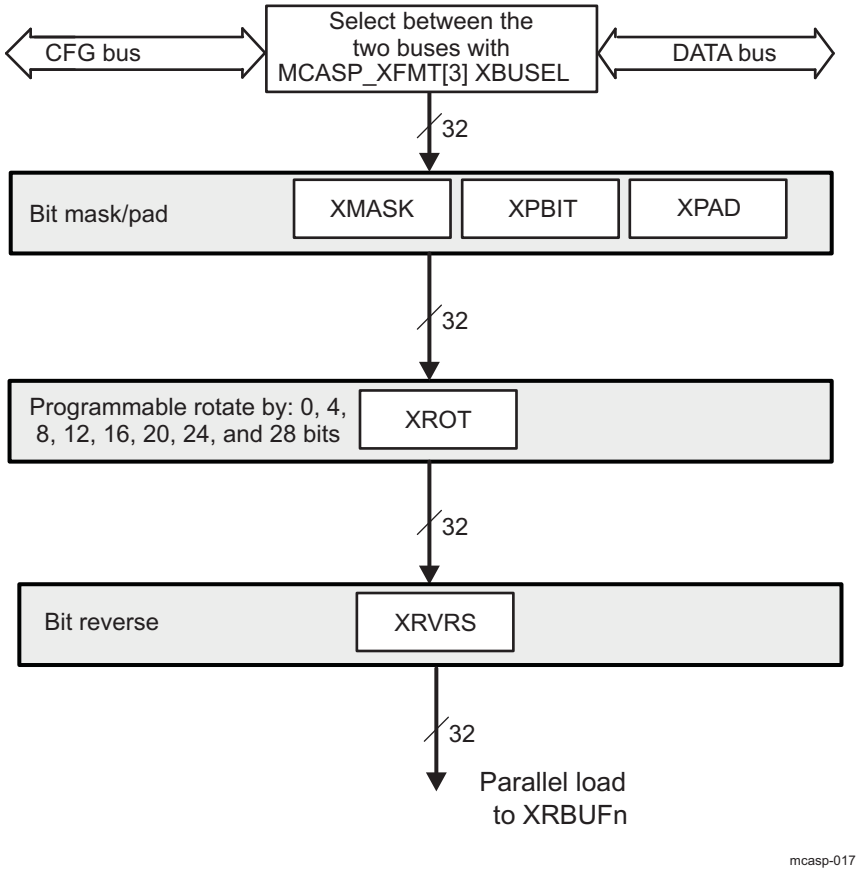
- Slot (or Time slot) size = 8, 12, 16, 20, 24, 28, 32 bits
- Word size <= Slot size
- Alignment: when more bits/slot than bits/words, then:

  - Left aligned = word shifted first, remaining bits are pad
  - Right aligned = pad bits are shifted first, word occupies the last bits in slot
- Order of bits shifted out:
  - MSB: most-significant bit of word is shifted out first, last bit is LSB
  - LSB: least-significant bit of word is shifted out last, last bit is MSB

Hardware support for these serial formats comes from the programmable options in the bitstream format register – MCASP_XFMT:

- XRVRS: bit reverse (1) or no bit reverse (0)
- XROT: rotate right by 0, 4, 8, 12, 16, 20, 24, or 28 bits
- XSSZ: transmit slot size of 8, 12, 16, 20, 24, 28, or 32 bits

Figure 11-588. McASP Transmit Format Unit



As shown in Figure 11-588, the data to the transmit format unit can come from the configuration port (CFG) or the data port (DATA). The selection is made through the MCASP_XFMT[3] XBUSEL bit. According to port type selected, data transfer has different behaviour. For more details, refer to Section 11.9.4.10.1.3, *Transfers Through the DATA Port*, and Section 11.9.4.10.1.4, *Transfers Through the Configuration (CFG) Bus*.

In the transmit format unit (TFU), the input data bits are first masked-off with the MCASP_XMASK[31-0] MCASP_XMASK contents. The masked data is then right-rotated to MCASP_XFMT[2-0] XROT positions, to produce the output word for a TDM- or DIT- transmission.

The bit mask stage includes a full 32-bit mask register, allowing selected individual bits to pass through the stage unchanged or be masked off.

11.9.4.4.1.1 TDM Mode Transmission Data Alignment Settings

The TDM-mode transmission settings are relevant for I²S-protocol and protocols using more than 2 TDM-slots.

XSSZ should always be programmed to match the slot size of the serial stream. **Note that, TDM word size is not directly programmed into the McASP, but rather is used to determine the rotation needed in the XROT field.**

Table 11-1238 shows the XRVRS and XROT fields for each serial format and for both integer and Q31 fractional internal representations.

The Table 11-1238 assumes that all slot size (SLOT in Table 11-1238) and word size (WORD in Table 11-1238) options are multiples of 4, since the transmit rotate right unit only supports rotation by multiples of 4. However, the bit mask/pad unit does allow for any number of significant digits. For example, a Q31 number may have 19 significant digits (word) and be transmitted in a 24-bit slot; this would be formatted as a word size of 20 bits and a slot size of 24 bits. However, it is possible to set the bit mask unit to only pass the 19 most-significant digits (program the mask value to FFFF E000h). The digits that are not significant can be set to a selected pad value, which can be any one of the significant digits, a fixed value of 0, or a fixed value of 1.

The transmit bit mask/pad unit operates on data as an initial step of the transmit format unit, and the data is aligned in the same representation as it is written to the transmitter by the ARMSS or DSP (typically Q31 or integer).

Table 11-1238. McASP TFU TDM Mode Settings

Bit Stream Order	Bit Stream Alignment	Internal Numeric Representation	MCASP_XFMT bits	
			XROT[1]	XRVRS
MSB first[2]	Left aligned	Q31 fraction	0	1
MSB first	Right aligned	Q31 fraction	SLOT - WORD	1
LSB first	Left aligned	Q31 fraction	32 - WORD	0
LSB first	Right aligned	Q31 fraction	32 - SLOT	0
MSB first[2]	Left aligned	Integer	WORD	1
MSB first	Right aligned	Integer	SLOT	1
LSB first	Left aligned	Integer	0	0
LSB first	Right aligned	Integer	(32 - (SLOT - WORD)) % 32	0

(1) WORD = Word size rounded up to the nearest multiple of 4; SLOT = slot size; % = modulo operator
(2) To transmit in I²S format, select MSB first, left aligned, and also select XDATDLY = 01 (1 bit delay)

11.9.4.4.1.2 DIT Mode Transmission Data Alignment Settings

In case of a DIT-mode (S/PDIF protocol) transmission, while left-aligned Q31 data should be right-rotated to a multiple by 4 positions, no right-rotation is required for a right-aligned Q31 data. Because this is a rotation process, not a shifting process, bit 0 gets shifted back into bit 31 during the process.

The MCASP_XFMT[17-16] XDATDLY bit field must be set to a 0-bit delay (0x0 value).

For left-aligned Q31 data, the following transmit format unit settings process the data into right-aligned data, ready for transmission:

- MCASP_XFMT[2-0] XROT =
  - 0x2 (rotate right by 8 bits) – for a 24-bit output audio data
  - 0x3 (rotate right by 12 bits) – for a 20-bit output audio data
  - 0x4 (rotate right by 16 bits) – for a 16-bit output audio data
- MCASP_XFMT[15] XRVRS = 0x0 – Bit reversal is not enabled; the McASP naturally transmits and receives in a LSB-first order.
- MCASP_XMASK[32] MCASP_XMASK = 0xFFFFFF00 – 0xFFFF0000
- MCASP_XFMT[14-13] XPAD = 0x0 (Pad extra bits with 0s.)

For right-aligned data, the following transmit format unit settings process the data into right-aligned audio data ready for transmission:

- MCASP_XFMT[2-0] XROT = 0x0 (rotate right by 0 bits regardless of the audio word length)
- MCASP_XFMT[15] XRVRS = 0x0 – Bit reversal is not enabled; the McASP naturally transmits and receives in a LSB-first order.
- MCASP_XMASK[32] MCASP_XMASK = 0x00FFFFFF – 0x0000FFFF
- MCASP_XFMT[14-13] XPAD = 0x0 (Pad extra bits with 0s.)

The example settings provided in Table 11-1239 should be applied to McASP in cases of DIT-transmitting a Q31 data as a 24-bit, 20-bit and 16-bit left- or right- aligned audio word, respectively. Note that the listed settings let the McASP TFU preserve the most significant bits and cut only the LSBs of the original Q31 CPU data:

Table 11-1239. McASP TFU DIT-Mode Example Settings

Output Audio Word Alignment	Audio Word Length	Right-rotation (multiple of 4-bit positions)	MCASP_XMASK	XROT
LEFT	16	16	0xFFFF0000	0x4
LEFT	20	12	0xFFFFF000	0x3
LEFT	24	8	0xFFFFFF00	0x2
RIGHT	16	0	0x0000FFFF	0x0
RIGHT	20	0	0x000FFFFF	0x0
RIGHT	24	0	0x00FFFFFF	0x0

Assuming that a Q31 data word 0xFA5AFxxx (where x-marked nibbles of the data are applied as padding bits of the word) is generated by CPU on the McASP CFG (peripheral) port. To transmit a left-aligned 20-bit version of same word, preserving the MSBs, according to the Table 11-1239, the user must set MCASP_XMASK = 0xFFFFF000, and to select a right-rotation to 12 positions (XROT = 0x3).

- After applying 0-s (XPAD = 0) as masking-off bits at the first TFU stage, word is transformed to the word 0xFA5AF000.
- After a rotation by 12 positions to the right is performed in TFU, the 20-bit output word obtained is : 0x000FA5AF. Thus the word gets ready for transmission being mapped with its LS-bit as bit 8 and its MS-bit as bit 27 within a S/PDIF bitstream. This word is shifted in a LSB-to-MSB order (XRVRS = 0x0) out of the XRSR register during a DIT-transmission.

Assuming that a right-aligned Q31 data word – 0x yyyyE4B4 is generated by CPU on the McASP CFG (peripheral) port (with the presumption that y-marked nibbles of the input data are applied as padding bits). To transmit a right-aligned 16-bit version of same word, preserving the MSBs, according to the table McASP TFU Example Settings, user is supposed to set MCASP_XMASK = 0x0000FFFF, and to select right-rotation to 0 positions (XROT = 0x0).

- After masking-off with 0s at first TFU stage, word is transformed to 0x0000E4B4.
- Since no rotation is applied, the 16-bit output word obtained is actually the one obtained in the masking stage – 0x0000E4B4.

The above examples use internal representation in integer and Q31 notation, but other fractional notations are also possible.

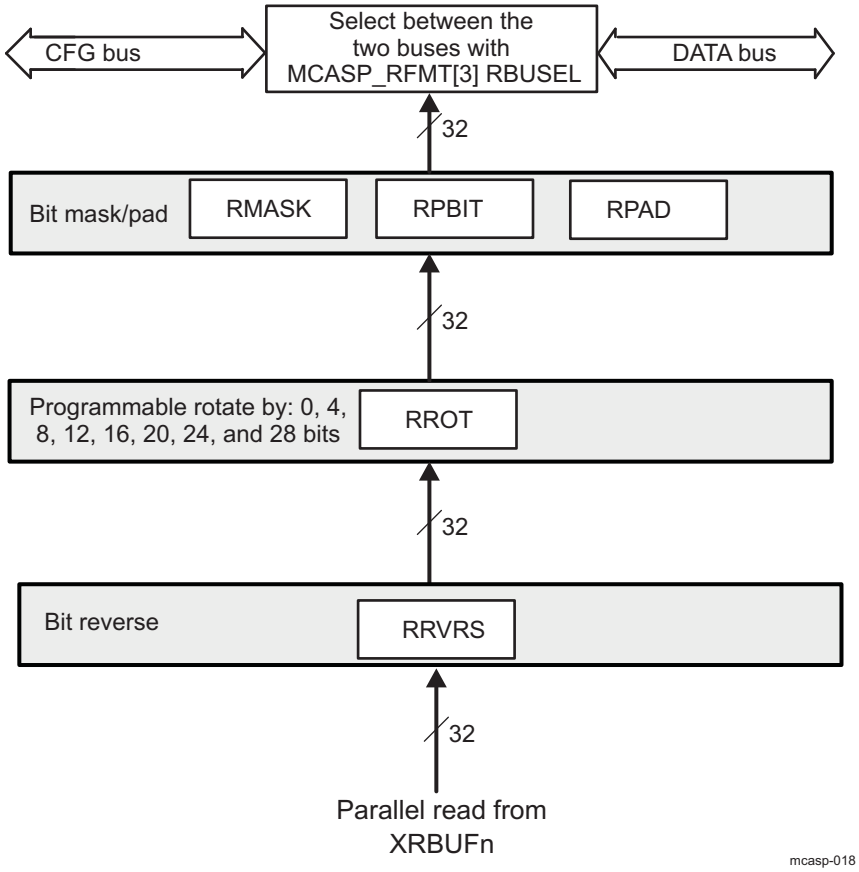
11.9.4.4.2 Receive Format Unit

The McASP receive formatting unit consists of three stages:

- Bit mask (masks off bits)
- Rotate right (aligns data within word)
- Bit reversal (selects between MSB first or LSB first)

Figure 11-589 shows the receive format unit (RFU).

Figure 11-589. McASP Receive Format Unit



The McASP receiver supports serial formats of:

- Slot or time slot size = 8, 12, 16, 20, 24, 28, 32 bits
- Word size <= Slot size
- Alignment when more bits are available per slot than bits per word within the slot, then:
  - Left aligned = word shifted first, remaining bits are pad
  - Right aligned = pad bits are shifted first, word occupies the last bits in slot
- Order of bits shifted out:
  - MSB: most-significant bit of word is shifted out first, last bit is LSB
  - LSB: least-significant bit of word is shifted out last, last bit is MSB

Hardware support for these serial formats comes from the programmable options in the receive bitstream format register – MCASP_RFMT:

- RRVRS: bit reverse (1) or no bit reverse (0)
- RROT: rotate right by 0, 4, 8, 12, 16, 20, 24, or 28 bits
- RSSZ: receive slot size of 8, 12, 16, 20, 24, 28, or 32 bits

As shown on Figure 11-589, the data processed in the RFU can be output to CPUs (device ARMSS/DSP) through the configuration port (CFG) or the data port (DATA). The selection is made through the MCASP_RFMT[3] RBUSEL bit. According to port type selected, data transfer has different behaviour. For more details, refer to Section 11.9.4.10.1.3, *Transfers Through the DATA Port*, and Section 11.9.4.10.1.4, *Transfers Through the Configuration (CFG) Bus*.

11.9.4.4.2.1 TDM Mode Reception Data Alignment Settings

RSSZ should always be programmed to match the slot size of the serial stream. **Note that the word size is not directly programmed into the McASP, but rather is used to determine the rotation needed in the RROT field.**

Table 11-1240 shows the RRVRS and RROT fields for each serial format and for both integer and Q31 fractional internal representations.

Table 11-1240. McASP RFU Settings

Bit Stream Order	Bit Stream Alignment	Internal Numeric Representation	MCASP_RFMT bits	
			RROT[1]	RRVRS
MSB first[2]	Left aligned	Q31 fraction	SLOT	1
MSB first	Right aligned	Q31 fraction	WORD	1
LSB first	Left aligned	Q31 fraction	(32 - (SLOT - WORD)) % 32	0
LSB first	Right aligned	Q31 fraction	0	0
MSB first[2]	Left aligned	Integer	SLOT - WORD	1
MSB first	Right aligned	Integer	0	1
LSB first	Left aligned	Integer	32 - SLOT	0
LSB first	Right aligned	Integer	32 - WORD	0

(1) WORD = Word size rounded up to the nearest multiple of 4; SLOT = slot size; % = modulo operator
(2) To receive in I^2S format, select MSB first, left aligned, and also select RDATDLY = 01 (1 bit delay)

Table 11-1240 assumes that all slot size and word size options are multiples of 4; since the receive rotate right unit only supports rotation by multiples of 4. However, the bit mask/pad unit does allow for any number of significant digits. For example, a Q31 number may have 19 significant digits (word) and be received in a 24-bit slot; this would be formatted as a word size of 20 bits and a slot size of 24 bits. However, it is possible to set the bit mask unit to only pass the 19 most-significant digits (program the mask value to FFFF E000h). The digits that are not significant can be set to a selected pad value, which can be any one of the significant digits, a fixed value of 0, or a fixed value of 1. The receive bit mask/pad unit operates on data as the final step of the receive format unit (see Figure 11-589), and the data is aligned in the same representation as it is read from the receiver by the DSP (typically Q31 or integer).

11.9.4.5 McASP State-Machines

The receive and transmit sections have independent state machines.

Each state-machine controls the interactions between the various units in the McASP Rx and Tx sections, respectively. In addition, each state-machine keeps track of error conditions and serial port status. No serial transfers can occur until the RX/TX state-machine is released from reset.

The transmit state-machine is controlled by the transmit bitstream format register (MCASP_XFMT) and it reports the McASP status and error conditions in the transmitter status register (MCASP_XSTAT).

Similarly, the receive state-machine is controlled by the receive bitstream format register (MCASP_RFMT) and it reports the McASP status and error conditions in the receiver status register (MCASP_RSTAT).

11.9.4.6 McASP TDM Sequencers

There are separate TDM sequencers for the transmit section and the receive section. Each TDM sequencer keeps track of the slot count. In addition, the TDM sequencer checks the bits of MCASP_RTDM/MCASP_XTDM and determines if the McASP should receive/transmit in that time slot.

There are two possibilities for a slot: The McASP either performs Rx/Tx operations during the time slot (transmit/receive bit is active), or the McASP skips Rx/Tx operations during the time slot (transmit/receive bit is inactive). In the latter case, no transfers between the XRBUF and XRSR registers in the serializer would occur during that time slot.

In addition, during time of inactive slots, the serializers programmed as transmitters place their data output pins – AXRn in a predetermined state – logic low, high, or high impedance (tri-stated) as programmed in each serializer control register MCASP_SRCTL0 through MCASP_SRCTL15[3-2] DISMOD. Refer also to Section 11.9.4.9.2.1, *TDM Time Slots Generation and Processing*, for details on how DMA event or interrupt generations are handled during inactive time slots in TDM mode.

In case of a DIT-transmission (S/PDIF transfers): the time division multiplexing (TDM) sequencer is used to count the 384 subframes (slots) in the DIT block. If currently transmitting slot 1, slot 2 (next value of the TDM slot counter) should be used during the encode phase to select the appropriate C, V, and U bit, because the data encoded and written to a MCASP_XBUF0 through MCASP_XBUF15 (where n = 0 to 15) register during the current time slot (slot 1) is actually shifted out on the next time slot.

The transmit TDM sequencer is controlled by the MCASP_XTDM register and reports the current transmit slot to the MCASP_XSLOT[9-0] XSLOTCNT bit field.

11.9.4.7 McASP Software Reset

The McASP can be put into reset through the global transmit and receive control register (MCASP_GBLCTL). A valid serial clock must be supplied to the McASP to assert the software reset bits in the MCASP_GBLCTL register.

11.9.4.8 McASP Power Management

Table 11-1241 describes power-management features available to the McASP.

Table 11-1241. Local Power-Management Features

Feature	Registers	Description
Slave idle modes	MCASP_PWRIDLESYSCONFIG[1-0] IDLE_MODE	Force-idle, no-idle, and smart-idle modes are available.

CAUTION

No wakeup schema is supported for the McASP. To ensure a correct behavior after enabling McASP at device level, the user software is strongly recommended to choose **No Idle** mode, setting MCASP_PWRIDLESYSCONFIG[1-0] IDLE_MODE to 0x1. Before disabling McASP at device level, user software is strongly recommended to choose a **Smart-Idle** mode, setting MCASP_PWRIDLESYSCONFIG[1-0] IDLE_MODE to 0x2.

11.9.4.9 McASP Transfer Modes

11.9.4.9.1 Burst Transfer Mode

The McASP supports a burst transfer mode, which is useful for nonaudio data such as passing control information between two processors. Burst transfer mode uses a synchronous serial format similar to the TDM mode. The frame sync generation is not periodic or time-driven as in TDM mode, but data driven, and the frame sync is generated for each data word transferred.

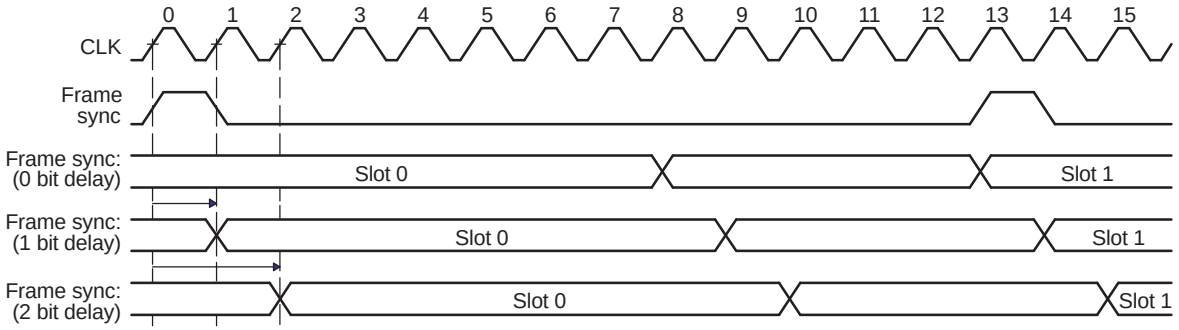
When operating in burst frame sync mode (see Figure 11-590), as specified for transmit (MCASP_AFSXCTL[15-7] XMOD = 0) and receive (MCASP_AFSRCTL[15-7] RMOD = 0), one slot is shifted for each active edge of the frame sync signal that is recognized. Additional clocks after the slot and before the next frame sync edge are ignored.

In burst frame sync mode, the frame sync delay may be specified as 0, 1, or 2 serial clock cycles. This is the delay between the frame sync active edge and the start of the slot. The frame sync signal lasts for a single bit clock duration (MCASP_AFSRCTL[4] FRWID = 0, MCASP_AFSXCTL[4] FXWID = 0).

For transmit, when generating the transmit frame sync internally, the frame sync begins when the previous transmission has completed and when all the XBUFn (for every serializer set to operate as a transmitter) has been updated with new data.

For receive, when generating the receive frame sync internally, frame sync begins when the previous transmission has completed and when all the RBUFn (for every serializer set to operate as a receiver) has been read.

Figure 11-590. McASP Burst Frame Sync Mode



The control registers must be configured as follows for the burst transfer mode. The burst mode specific bit fields are in bold face:

- MCASP_PFUNC: The clock, frame, data pins must be configured for McASP function.
- MCASP_PDIR: The clock, frame, data pins must be configured to the direction desired.
- MCASP_PDOUT, MCASP_PDIN, MCASP_PDSET, MCASP_PDCLR: Not applicable. Leave at default.
- MCASP_AMUTE: Not applicable. Leave at default.
- MCASP_DLBCTL: If loopback mode is desired, configure this register according to Section 11.9.4.14, *McASP Loopback Modes*, otherwise leave this register at default.
- MCASP_DITCTL: DITEN must be left at default 0 to select non-DIT mode. Leave the register at default.
- MCASP_RMASK/MCASP_XMASK: Mask desired bits according to Section 11.9.4.4, *Format Units*.
- MCASP_RFMT/MCASP_XFMT: Program all fields according to data format desired. See Section 11.9.4.4, *Format Units*.
- MCASP_RFMT/MCASP_XFMT: Clear RMOD/XMOD bits to 0 to indicate burst mode. Clear FRWID/FXWID bits to 0 for single bit frame sync duration. Configure other fields as desired.
- MCASP_ACLKRCTL/MCASP_ACLKXCTL: Program all fields according to bit clock desired. See Section 11.9.4.2, *McASP Clock and Frame-Sync Configurations*.
- MCASP_AHCLKRCTL/MCASP_AHCLKXCTL: Program all fields according to high-frequency clock desired, see Section 11.9.4.2, *McASP Clock and Frame-Sync Configurations*.
- MCASP_RTDM/MCASP_XTDM: Program RTDMS0/XTDMS0 to 1 to indicate one active slot only. Leave other fields at default.
- MCASP_RINTCTL/MCASP_XINTCTL: Program all fields according to interrupts desired.
- MCASP_RCLKCHK/MCASP_XCLKCHK: Not applicable. Leave at default.
- MCASP_SRCTL0 through MCASP_SRCTL15: Program SRMOD to inactive/transmitter/receiver as desired. DISMOD is not applicable and should be left at default.
- MCASP_DITCSRA0 through MCASP_DITCSRA5, MCASP_DITCSRB0 through MCASP_DITCSRB5, MCASP_DITUDRA0 through MCASP_DITUDRA5, MCASP_DITUDRB0 through MCASP_DITUDRB5: Not applicable. Leave at default.

11.9.4.9.2 Time-Division Multiplexed (TDM) Transfer Mode

The McASP time-division multiplexed (TDM) transfer mode supports the TDM format discussed in Section 11.9.2.2.3, *TDM Format*.

Transmitting data in the TDM transfer mode requires a minimum set of pins:

- ACLKX – transmit bit clock
- AFSX – transmit frame sync (or commonly called left/right clock)
- One or more serial data pins, AXRn, whose serializers are configured to transmit

For more details on McASP transmitting serializers clock and frame sync options, refer to the section Section 11.9.4.2.1, *Transmit Clock*, and Section 11.9.4.2.3, *Frame-Sync Generator* .

Similarly, to receive data in the TDM transfer mode requires a minimum set of pins:

- ACLKR – receive bit clock
- AFSR – receive frame sync (or commonly called left/right clock)
- One or more serial data pins, AXRn, whose serializers are configured to receive

For more details on McASP receiving serializers clock and frame sync options, refer to the Section 11.9.4.2.2, *Receive Clock*, and Section 11.9.4.2.3, *Frame-Sync Generator* .

The control registers must be configured as follows for the TDM mode. The TDM mode specific bit fields are highlighted in bold:

- MCASP_PFUNC: The clock, frame, data pins must be configured for McASP function.
- MCASP_PDIR: The clock, frame, data pins must be configured to the direction desired.
- MCASP_PDOUT, MCASP_PDIN, MCASP_PDSET, MCASP_PDCLR: Not applicable. Leave at default.
- MCASP_GBLCTL: Follow the initialization sequence described in the , *Operational Modes Configuration*.
- MCASP_AMUTE: Leave this register at default state.
- MCASP_DLBCTL: If loopback mode is desired, configure this register according to Section 11.9.4.14, *McASP Loopback Modes*, otherwise leave this register at default.
- MCASP_DITCTL: DITEN must be left at default 0 to select TDM mode (transmitters only).
- MCASP_RMASK/MCASP_XMASK: Mask desired bits according to Section 11.9.4.4, *Format Units*.
- MCASP_RFMT/MCASP_XFMT: Program all fields according to data format desired, see Section 11.9.4.4, *Format Units*.
- MCASP_AFSRCTL/MCASP_AFSXCTL: Set RMOD/XMOD bits to (0x2–0x20) for Rx/Tx (2–32 slots) TDM mode. In addition, set RMOD to 0x180 if 384-slot TDM stream has to be received by McASP. Configure other fields as desired.
- MCASP_ACLKRCTL/MCASP_ACLKXCTL: Program all fields according to bit clock desired. For more inforamtion, refer to Section 11.9.4.2.
- MCASP_AHCLKRCTL/MCASP_AHCLKXCTL: Program all fields according to high-frequency clock desired. For more details, refer to Section 11.9.4.2.
- MCASP_RTDM/MCASP_XTDM: Program all fields according to the time slot characteristics desired.
- MCASP_XINTCTL : Program all fields according to transmit interrupts desired.
- MCASP_RCLKCHK/MCASP_XCLKCHK: Program all fields according to clock checking desired.
- MCASP_SRCTL0 through MCASP_SRCTL15: Program all fields according to serializer operation desired.

NOTE: The MCASP_DITCSRA0 through MCASP_DITCSRA5, MCASP_DITCSRB0 through MCASP_DITCSRB5, MCASP_DITUDRA0 through MCASP_DITUDRA5, MCASP_DITUDRB0 through MCASP_DITUDRB5 settings are NOT applicable in TDM transfer modes. They have to be kept at their default values.

11.9.4.9.2.1 TDM Time Slots Generation and Processing

TDM mode on the McASP can extend to support multiprocessor applications, with up to 32 time slots per frame. For each of the time slots, the McASP may be configured to participate or to be inactive by configuring MCASP_XTDM and/or MCASP_RTDM registers.

The TDM sequencer (separate ones for transmit and receive) functions in this mode. The TDM sequencer counts the slots beginning with the frame sync. For each slot, the TDM sequencer checks the respective bit in either MCASP_XTDM or MCASP_RTDM to determine if the McASP transmits/receives in that time slot.

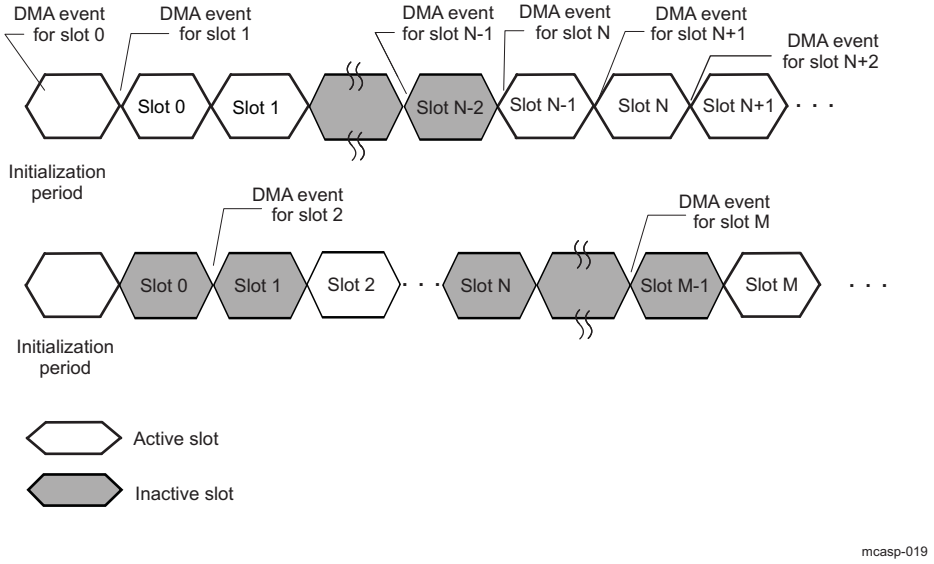
NOTE: If a MCASP_XTDM/MCASP_RTDM bit defines an active slot (number of slot matches the bit position), the McASP functions normally during that time slot; otherwise, the McASP is inactive during that time slot; no update to the buffer occurs, and no event is generated. McASP (transmit only) data pins are automatically set to a high-impedance state, 0, or 1 during that slot, as determined by bitfield MCASP_SRCTL0 through MCASP_SRCTL15[3-2] DISMOD.

Figure 11-591 shows when the transmit DMA event – XEVENT is generated. See Section 11.9.4.10.1, *Data Ready Status and Event/Interrupt Generation*, for details on data ready and the initialization period indication. The transmit DMA event for an active time slot (slot N) is generated during the previous time slot (slot N - 1), regardless of the previous time slot (slot N - 1) being active or inactive.

During an active transmit time slot (slot N), if the next time slot (slot N + 1) is configured to be active, the copy from XRBUFn to XRSRn generates the DMA event for time slot N + 1. If the next time slot (slot N + 1) is configured to be inactive, then the DMA event will be delayed to time slot M - 1. In this case, slot M is the next active time slot. The DMA event for time slot M is generated during the first bit time of slot M - 1.

The receive DMA event is generated after data is received in the buffer (looks back in time). If a time slot is disabled, then no data is copied to the buffer for that time slot and no DMA event is generated.

Figure 11-591. McASP Transmit DMA Event (XEVENT) Generation in TDM Time Slots



11.9.4.9.2.2 Special 384-Slot TDM Mode for Connection to External DIR

The McASP receiver also supports a 384 time slot TDM mode (DIR mode), to support S/PDIF receiver ICs whose natural block (block corresponds to McASP frame) size is 384 samples. The receive TDM time slot register (MCASP_RTDM) should be programmed to all 1s during reception of a DIR block. **Other TDM functionalities (for example, inactive slots) are not supported (only the slot counter counts the 384 subframes in a block).** To receive data in DIR mode, the following pins are typically needed:

- ACLKR – receive bit clock
- AFSR – receive frame sync (or commonly called left/right clock)
- In this mode, AFSR should be connected to a DIR which outputs a start of block signal, instead of LRCLK
- One or more serial data pins, AXRn, whose serializers have been configured to receive

- For this special DIR mode, the control registers can be configured just as for TDM mode, except set RMOD in MCASP_AFSRCTL to 384 (0x180) to receive 384 time slots

11.9.4.9.3 McASP DIT Transfer Mode

The DIT transfer mode of the McASP also supports transmission of audio data in S/PDIF, AES-3, and IEC-60958 formats. These formats are designed to carry audio data between different systems through an optical or coaxial cable. **The DIT mode applies only to a serializer configured as transmitter, NOT receiver.** For a description of the S/PDIF format, see Section 11.9.2.2.5, *S/PDIF Coding Format*.

11.9.4.9.3.1 Transmit DIT Encoding

When the McASP operates in DIT mode, the data transmitted is output as a biphase-mark encoded bitstream, with preamble, channel status, user data, validity, and parity automatically stuffed into the bitstream by the McASP. The McASP includes separate validity bits for even/odd subframes and two 384-bit RAM modules to hold channel status and user data bits.

NOTE: The transmit TDM time slot register (MCASP_XTDM) should be programmed to all 1s during DIT mode. TDM functionality is not supported in DIT mode, except that the TDM slot counter counts the DIT subframes.

To transmit data in DIT mode, the following pins are typically required:

- AHCLKX – transmit high-frequency master clock (The internal clock source can be used instead.)
- One serial data pin (AXRn) of a serializer n configured to transmit.

For DIT Mode Transmission Data Alignment Settings see Section 11.9.4.4.1.2, *DIT Mode Transmission Data Alignment Settings*.

If the McASP is configured to transmit in the DIT mode on more than one serial data pin, the bit streams on all pins will be synchronized. In addition, although they will carry unique audio data, they will carry the same channel status, user data, and validity information.

The actual 24-bit audio data must always be in bit positions 23–0 after passing through the first three stages of the transmit format unit.

11.9.4.9.3.2 Transmit DIT Clock and Frame-Sync Generation

The DIT transmitter works only in the following configuration:

- In the transmit frame control register (MCASP_AFSXCTL):
  - Internally generated transmit frame sync, FSXM = 1
  - Rising-edge frame sync, FSXP = 0
  - Bit-width frame sync, FXWID = 0
  - 384-slot TDM, XMOD = 1 1000 0000b
- In the transmit clock control register (MCASP_ACLKXCTL), ASYNC = 1
- In the transmit bitstream format register (MCASP_XFMT), XSSZ = 1111 (32-bit slot size)

All combinations of AHCLKX and ACLKX are supported.

The following summarizes the register configurations required for DIT mode. DIT mode-specific bit fields are in bold face:

- MCASP_PFUNC: The data pin – AXRn must be configured for McASP function. If AHCLKX is used, it must also be configured for McASP function. Other pins can be configured to function as GPIOs, if desired.
- MCASP_PDIR: The data pin must be configured as output. If internal clock source AUXCLK is used as the reference clock, it may be output as the AHCLKX device level signal by configuring AHCLKX pin as an output.
- MCASP_GBLCTL: Global initialization
- MCASP_AMUTE: Leave this register at default state.

- MCASP_DITCTL: The DITEN bit must be set to 0b1 to enable DIT mode. Configure other bits as desired.
- MCASP_XMASK: Mask the desired bits, depending upon left-aligned or right-aligned internal data.
- MCASP_XFMT: XDATDLY = 0. XRVRS = 0. XPAD = 0. XSSZ = Fh (32-bit slot). XBUSEL = configured as desired. The XROT bit is configured, as described in Section 11.9.4.4.1.2, *DIT Mode Transmission Data Alignment Settings* .
- MCASP_AFSXCTL: Configure the bits according to former discussions.
- MCASP_ACLKXCTL: ASYNC = 1. Program the CLKXDIV bits to obtain the bit clock rate desired. CLKXM = 1.
- MCASP_AHCLKXCTL: Program the HCLKXDIV bits to obtain the high-frequency bit clock rate desired.
- MCASP_XTDM: Set to FFFF FFFFh for all active slots for DIT transfers.
- MCASP_XINTCTL: Program all fields according to the interrupts desired.
- MCASP_XCLKCHK: Program all fields according to the clock checking desired.
- MCASP_SRCTL0 through MCASP_SRCTL15: Set SRMOD = 1 (transmitter) for the DIT pins.
- MCASP_DITCSRA0 through MCASP_DITCSRA5 and MCASP_DITCSRB0 through MCASP_DITCSRB5: Program the channel status bits as desired.
- MCASP_DITUDRA0 through MCASP_DITUDRA5 and MCASP_DITUDRB0 through MCASP_DITUDRB5: Program the user data bits as desired.

NOTE: In DIT mode, the transmitter can support a 192 kHz frame rate (stereo) on up to 2 serial data pins simultaneously (note that the internal bit clock for DIT runs two times faster than the equivalent bit clock for TDM (I^2S) mode, due to the need to generate Biphase Mark Encoded Data, see Section 11.9.2.2.5.1, *Biphase-Mark Code*).

11.9.4.9.3.3 DIT Channel Status and User Data Register Files

The channel status registers (MCASP_DITCSRA0 through MCASP_DITCSRA5 and MCASP_DITCSRB0 through MCASP_DITCSRB5) and user data registers (MCASP_DITUDRA0 through MCASP_DITUDRA5 and MCASP_DITUDRB0 through MCASP_DITUDRB5) are not double-buffered. Typically, programmers use one of the synchronizing interrupts, such as the last slot, to create an event at a safe time so the register may be updated. In addition, the CPU reads the transmit TDM slot counter to determine which word of the register is being used.

It is a software requirement to avoid writing to the word of user data and channel status that are being used to encode the current time slot; otherwise, it is undetermined whether old or new data is used to encode the bitstream.

The DIT subframe format is defined in Section 11.9.2.2.5.2, *S/PDIF Subframe Format*. The channel status information (C) and user data (U) are defined in the following DIT control registers:

- MCASP_DITCSRA0 to MCASP_DITCSRA5: The 192 bits in these six registers contain the channel status information for the left channel within each frame.
- MCASP_DITCSRB0 to MCASP_DITCSRB5: The 192 bits in these six registers contain the channel status information for the right channel within each frame.
- MCASP_DITUDRA0 to MCASP_DITUDRA5: The 192 bits in these six registers contain the user data information for the left channel within each frame.
- MCASP_DITUDRB0 to MCASP_DITUDRB5: The 192 bits in these six registers contain the user data information for the right channel within each frame.
- The S/PDIF block format is shown in Figure 11-581. There are 192 frames within a block (frame 0 to frame 191). There are two subframes within each frame (subframes 1 and 2 for the left and right channels, respectively).

The channel status and user data information sent on each subframe is summarized in Table 11-1242.

Table 11-1242. McASP Channel Status and User Data for Each DIT Block

Frame	Subframe	Preamble	Channel Status Defined in:	User Data Defined in:
Defined by MCASP_DITCSRA0, MCASP_DITCSRB0, MCASP_DITUDRA0, MCASP_DITUDRB0				
0	1 (L)	B	MCASP_DITCSRA0[0]	MCASP_DITUDRA0[0]
0	2 (R)	W	MCASP_DITCSRB0[0]	MCASP_DITUDRB0[0]
1	1 (L)	M	MCASP_DITCSRA0[1]	MCASP_DITUDRA0[1]
1	2 (R)	W	MCASP_DITCSRB0[1]	MCASP_DITUDRB0[1]
2	1 (L)	M	MCASP_DITCSRA0[2]	MCASP_DITUDRA0[2]
2	2 (R)	W	MCASP_DITCSRB0[2]	MCASP_DITUDRB0[2]
…	…	…	…	…
31	1 (L)	M	MCASP_DITCSRA0[31]	MCASP_DITUDRA0[31]
31	2 (R)	W	MCASP_DITCSRB0[31]	MCASP_DITUDRB0[31]
Defined by MCASP_DITCSRA1, MCASP_DITCSRB1, MCASP_DITUDRA1, MCASP_DITUDRB1				
32	1 (L)	M	MCASP_DITCSRA1[0]	MCASP_DITUDRA1[0]
32	2 (R)	W	MCASP_DITCSRB1[0]	MCASP_DITUDRB1[0]
…	…	…	…	…
63	1 (L)	M	MCASP_DITCSRA1[31]	MCASP_DITUDRA1[31]
63	2 (R)	W	MCASP_DITCSRB1[31]	MCASP_DITUDRB1[31]
Defined by MCASP_DITCSRA2, MCASP_DITCSRB2, MCASP_DITUDRA2, MCASP_DITUDRB2				
64	1 (L)	M	MCASP_DITCSRA2[0]	MCASP_DITUDRA2[0]
64	2 (R)	W	MCASP_DITCSRB2[0]	MCASP_DITUDRB2[0]
…	…	…	…	…
95	1 (L)	M	MCASP_DITCSRA2[31]	MCASP_DITUDRA2[31]
95	2 (R)	W	MCASP_DITCSRB2[31]	MCASP_DITUDRB2[31]
Defined by MCASP_DITCSRA3, MCASP_DITCSRB3, MCASP_DITUDRA3, MCASP_DITUDRB3				
96	1 (L)	M	MCASP_DITCSRA3[0]	MCASP_DITUDRA3[0]
96	2 (R)	W	MCASP_DITCSRB3[0]	MCASP_DITUDRB3[0]
…	…	…	…	…
127	1 (L)	M	MCASP_DITCSRA3[31]	MCASP_DITUDRA3[31]
127	2 (R)	W	MCASP_DITCSRB3[31]	MCASP_DITUDRB3[31]
Defined by MCASP_DITCSRA4, MCASP_DITCSRB4, MCASP_DITUDRA4, MCASP_DITUDRB4				
128	1 (L)	M	MCASP_DITCSRA4[0]	MCASP_DITUDRA4[0]
128	2 (R)	W	MCASP_DITCSRB4[0]	MCASP_DITUDRB4[0]
…	…	…	…	…
159	1 (L)	M	MCASP_DITCSRA4[31]	MCASP_DITUDRA4[31]
159	2 (R)	W	MCASP_DITCSRB4[31]	MCASP_DITUDRB4[31]
Defined by MCASP_DITCSRA5, MCASP_DITCSRB5, MCASP_DITUDRA5, MCASP_DITUDRB5				
160	1 (L)	M	MCASP_DITCSRA5[0]	MCASP_DITUDRA5[0]
160	2 (R)	W	MCASP_DITCSRB5[0]	MCASP_DITUDRB5[0]
…	…	…	…	…
191	1 (L)	M	MCASP_DITCSRA5[31]	MCASP_DITUDRA5[31]
191	2 (R)	W	MCASP_DITCSRB5[31]	MCASP_DITUDRB5[31]

11.9.4.10 McASP Data Transmission and Reception

The CPU services the McASP by writing data to the MCASP_XBUF0 through MCASP_XBUF15 registers for transmit operations, and by reading data from the MCASP_RBUF0 through MCASP_RBUF15 registers for receive operations. The McASP sets status flags and notifies the CPU whenever data is ready to be serviced. Section 11.9.4.10.1, *Data Ready Status and Event/Interrupt Generation*, discusses data-ready status in details.

The McASP transmit/receive XRBUFn buffer can be accessed through one of the two peripheral ports of the device:

- DATA port: This port is dedicated to DMA initiated data transfers on the device for McASP transmit (Tx) purposes.
- Configuration bus (CFG): The configuration bus – CFG port is used for peripheral configuration control and receive/transmit data transfers initiated by the host CPU in the device.

Section 11.9.4.10.1.3, *Transfers Through the Data Port (DATA)*, and Section 11.9.4.10.1.4, *Transfers Through the Configuration Bus (CFG),* discuss how to perform transfers through the data port (DATA) and the configuration port (CFG), respectively.

The CPU and DMA usages are discussed in Section 11.9.4.10.1.5, *Using the device CPUs for McASP Servicing*, and Section 11.9.4.10.1.6, *Using the DMA for McASP Servicing*, respectively.

McASP DATA port allows DMAs to access the McASP transmit buffer more efficiently, using burst transfers. The physical addresses to access these registers are listed in Table 11-1248.

11.9.4.10.1 Data Ready Status and Event/Interrupt Generation

11.9.4.10.1.1 Transmit Data Ready

The transmit data ready flag – XDATA in the MCASP_XSTAT register reflects the data ready status of XRBUFn buffers for all of the active slot transmitting serializers. The XDATA flag is set whenever data is transferred from a transmitting serializer buffer – XRBUFn to its corresponding XRSRn shift register. Thus, the XDATA bit indicates the global event that some of the serializers data buffer – XRBUFn is emptied and ready to accept new data from the host (ARMSS/DSP or DMAs). The transmit data ready event is individually indicated per serializer in its corresponding control register MCASP_SRCTL0 through MCASP_SRCTL15[4] XRDY status bit. When this bit is set to 0b1, it notifies to host that this serializer Tx buffer must be serviced (written). When MCASP_XBUF0 through MCASP_XBUF15 is written to by the host, the MCASP_SRCTL0 through MCASP_SRCTL15[4] XRDY is deasserted to 0b0. As XDATA global flag is an OR-event of all active serializers XRDY flags, it indicates to software the moment, when write service operation has to be initiated by the McASP host (XDATA = 0b1). The XRDY flags have to be sequentially scanned by user software to determine which serializer MCASP_XBUF0 through MCASP_XBUF15 register has to be currently written. Once all requested MCASP_XBUF0 through MCASP_XBUF15 are written, the serializers control XRDY flags are cleared to 0b0. As a consequence, XDATA flag is deasserted to 0b0, to indicate to SW that write operation is completed for all serializers.

The global XDATA flag can be cleared when the MCASP_XSTAT[5] XDATA bit is written to 0b1, or once MCASP_XBUF0 through MCASP_XBUF15 registers of all the serializers, that have previously raised their XRDY flags, are written with corresponding active slot data by the host.

Whenever XDATA is set, the AXEVT event is automatically generated on MCASPi_XEVENT_DREQ line (if enabled in the MCASP_XEVTCTL register) to notify the DMA of the MCASP_XBUF0 through MCASP_XBUF15 empty status. An interrupt – MCASPi_X_INTR_REQ can be also generated if the XDATA interrupt is enabled in the MCASP_XINTCTL register (for details, see Section 11.9.4.12.1, *Transmit Data Ready Interrupt*).

For DMA requests, the McASP does not require that MCASP_XSTAT be read between DMA events. This means that, even if MCASP_XSTAT already has the XDATA flag set to 1 from a previous request, the next transfer triggers another DMA request.

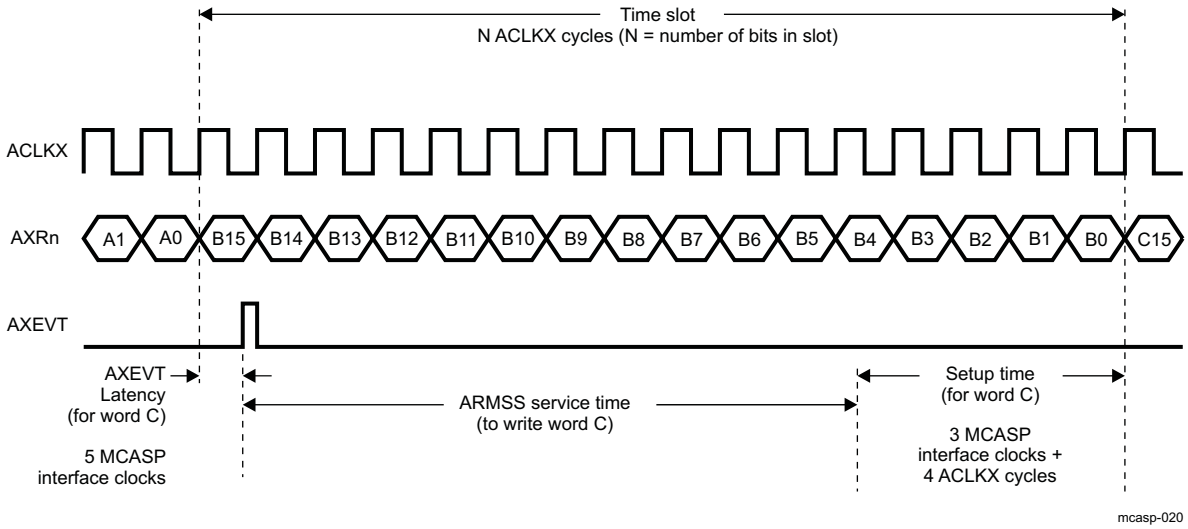
Because the serializer acts in lockstep, only one DMA event is generated to indicate that the transmit serializer is ready to be written to with new data.

Figure 11-592 shows the timing details of when AXEVT is generated at the McASP boundary. In this example, as soon as the last bit (A0) of word A is transmitted, the McASP sets the XDATA flag and generates an AXEVT event. However, it takes up to five McASP interface clocks (AXEVT latency) before AXEVT is active at the McASP boundary. Upon AXEVT, the DSP/ARMSS can begin servicing the McASP by writing word C into the MCASP_XBUF0 through MCASP_XBUF15 (service time). The CPU must write word C into the MCASP_XBUF0 through MCASP_XBUF15 within the setup time required by the McASP (setup time).

The maximum service time (see Figure 11-592) can be calculated as:

Service Time = Time Slot – AXEVT Latency –Setup Time

Figure 11-592. McASP CPU Service Time Upon Transmit DMA Event (AXEVT)



11.9.4.10.1.2 Receive Data Ready

Similarly, the receive data ready flag – RDATA in the MCASP_RSTAT register reflects the data ready status of XRBUFn buffers for all of the **active slot receiving serializers**. The RDATA flag is set whenever data is transferred from a receiving serializer shift register XRSRn to its corresponding XRBUFn data buffer. Thus, the RDATA bit indicates the global event that some of the receivers data buffer – RXBUFn already contains received data (that means that a buffer is full) and is ready to transfer it to the host (ARMSS/DSP). The receive data ready event is individually indicated per serializer in its corresponding control register MCASP_SRCTL0 through MCASP_SRCTL15 [5] RRDY status bit. When this bit is set to 0b1, it notifies to host that this serializer Rx buffer must be serviced (read). When MCASP_RBUF0 through MCASP_RBUF15 is read from the host, the MCASP_SRCTL0 through MCASP_SRCTL15 [5] RRDY is deasserted to 0b0. As RDATA global flag is an OR-event of all active serializers RRDY flags, it indicates to software the moment, when read service operation has to be initiated by the McASP host (RDATA=0b1). The RRDY flags have to be sequentially scanned by user software to determine which serializer MCASP_RBUF0 through MCASP_RBUF15 register has to be currently read. Once all requested MCASP_RBUF0 through MCASP_RBUF15 are read, the serializers control RRDY flags are cleared to 0b0. As a consequence, RDATA flag is deasserted to 0b0, to indicate to SW that read operation is completed for all serializers.

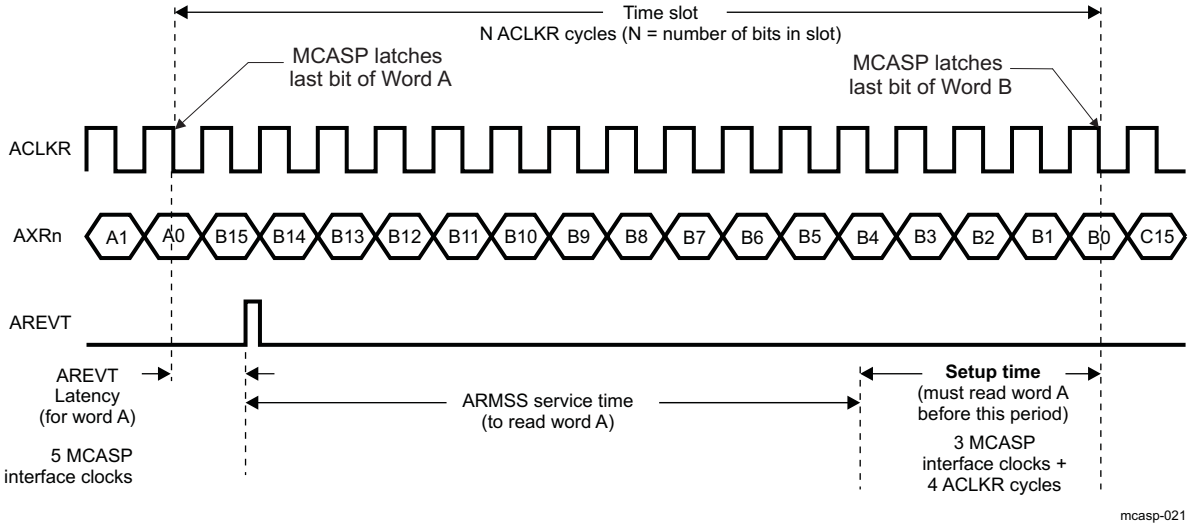
The global RDATA flag can be cleared when the MCASP_RSTAT[5] RDATA bit is written to 0b1, or once MCASP_RBUF0 through MCASP_RBUF15 registers of all the serializers, that have previously raised their RRDY flags, are read by the host.

Whenever RDATA is set, the AREVT event is automatically generated on MCASPi_REVENT_DREQ line (if enabled in the MCASP_REVTCTL register) to notify the DMA of the MCASP_RBUF0 through MCASP_RBUF15 full status. An interrupt – MCASPi_R_INTR_REQ can be also generated if the RDATA interrupt is enabled in the MCASP_RINTCTL register (for details, see Section 11.9.4.12.1, *Receive Data Ready Interrupt*).

Figure 11-593 shows the timing details of when AREVT event is generated at the McASP boundary. In this example, as soon as the last bit (bit A0) of Word A is received, the McASP sets the RDATA flag and generates an AREVT event. However, it takes up to five McASP interface clocks (AREVT Latency) before AREVT is active at the McASP boundary. Upon AREVT, the DSP/ARMSS can begin servicing the McASP by reading Word A from the MCASP_RBUF0 through MCASP_RBUF15 (service time). The DSP/ARMSS must read Word A from the MCASP_RBUF0 through MCASP_RBUF15 register no later than the setup time required by the McASP (Setup Time).

The maximum service time (see Figure 11-593) can be calculated as:
Service Time = Time Slot - AREVT Latency - Setup Time

Figure 11-593. McASP CPU Service Time Upon Receive Event (AREVT)



11.9.4.10.1.3 Transfers Through the Data Port (DATA)

CAUTION

To perform internal transfers through the DATA port, clear the XBUSEL/RBUSEL bit to 0b0 in the MCASP_XFMT/MCASP_RFMT register, respectively. Failure to do so may result in software malfunction.

In a typical McASP transfer scenario, the DMA Controller write accesses the XRBUFn transmit buffer through the McASP data port (DATA). ARMSS and DSP hosts can access both XRBUFn transmit and receive data buffers on their corresponding DATA port address via DATA port corresponding address on main interconect. To perform transfers through the DATA port, simply have the DMA Controller write the McASP Tx buffer through Interconect DATA port location. The ARMSS/DSP write or read access corresponding Tx/Rx buffer through Interconect DATA port location, as specified in the Table 11-1248. Although the transfer is passed through an integrated AFIFO transmit/receive buffer, the host (DMA or CPU) must follow the described below procedure to access the data buffers of each serializer, regardless the AFIFO is enabled or disabled. The AFIFO operation is described in Section 11.9.4.11.

For accesses through the DATA port, the DMAs/ARMSS or DSP service all the serializers through accessing only a single address. In addition, **the same physical DATA port address is used regardless of a read or write access is performed** by device ARMSS/DSP, see Table 11-1248. The McASP automatically cycles through the active slot transmitting/receiving serializers, internally generating the appropriate offsets.

NOTE: DATA port allows the DMAs/ARMSS or DSP to automatically access only the data buffers. There is no way for DMAs/ARMSS or DSP to access the McASP configuration registers addressing their corresponding McASP DATA port.

For transmit operations through the DATA port, **the host must always write to the same transmit buffer DATA port address** (which is same than the receive buffer DATA port adress) to service all of the active slot transmitting serializers. Regardless of McASP serializer 0 being configured inactive or active, the user software must always configure the DMAs/ARMSS or DSP destination address to match the DATA port location of TXBUF buffer (See Table 11-1248).

In addition, the DMAs/ARMSS or DSP must write the buffers of all transmitting serializers in incremental (although not necessarily consecutive) order. For example, if only serializers 1 and 3 are set up as active transmitters, the DMAs/ARMSS or DSP should write to the same transmit buffer DATA port address twice – first data for serializer 1 and second data for serializer 3 upon each transmit data ready event. This exact servicing order must be followed so that data appears in the appropriate serializers.

NOTE: For write transfers through McASP DATA port it is preferable to use DMA on corresponding Interconect. This is because DMAs initiated traffic gets better advantage of the burst transfers supported by DATA port.

For receive operations through the DATA port, **the DMA/ARMSS or DSP must always read from the same receive buffer DATA port address** (which is same than the transmit buffer DATA port adress) to service all of the active slot receiving serializers. Regardless of McASP serializer 0 being configured inactive or active, the user software must always configure the DMA/ARMSS or DSP source address to match the DATA port location of RXBUF buffer (see Table 11-1248).

In addition, reads from the receive buffer for all active slot receiving serializers through the Rx DATA port return data in incremental (although not necessarily consecutive) order. For example, if serializers 0, 1 and 3 are set up as active receivers, the ARMSS or DSP should read from the same receive buffer DATA port address three times to obtain data for serializers 0,1 and 3 in this exact order, upon each receive data ready event.

NOTE: To service a serializer for a transmit or receive operation through the McASP DATA port, the initiator always writes (preferably DMA) and reads from the same address (refer to Table 11-1248), respectively.

See Table 11-1248, *McASP DMA Space Registers*, for more details about XRBUFn buffer physical address corresponding to the McASP DATA port.

NOTE: When transmitting through the DATA port, the DMAs/ARMSS or DSP must write data (at the same address) to each serializer configured as **active** (active slot selected in MCASP_XTDM) and **transmit** (Tx enabled in MCASP_SRCTL0 through MCASP_SRCTL15) within each time slot. Failure to do so results in a buffer underrun condition (see Section 11.9.4.15.1, *Buffer Underrun Error - Transmitter*). Similarly, when DMA/ARMSS or DSP receives, data must be read from each serializer configured as **active** (active slot selected in MCASP_RTDM) and **receive** (Rx enabled in MCASP_SRCTL0 through MCASP_SRCTL15) within each time slot. Failure to do results in a buffer overrun condition (see Section 11.9.4.15.2, *Buffer Overrun Error - Receiver*).

11.9.4.10.1.4 Transfers Through the Configuration Bus (CFG)

CAUTION

To perform internal transfers through the configuration bus, set the XBUSEL/RBUSEL bit to 1 in the MCASP_XFMT/MCASP_RFMT registers, respectively. Failure to do so may result in software malfunction.

In this method, the DMAs/ARMSS or DSP accesses the XRBUFn transmit or receive buffer through corresponding configuration bus (CFG) address.

The exact XRBUFn transmit/receive buffer physical address for any particular serializer is determined by adding the transmit/receive buffer alias register offset for that particular serializer to the base address of McASP CFG port. The XRBUFn buffer of the n-th serializer configured as a transmitter is aliased – MCASP_XBUF0 through MCASP_XBUF15 in the CFG port address space. For example, the XRBUF2 transmit buffer is mapped as the MCASP_XBUF2 register. Similarly, the XRBUFn buffer of the n-th serializer configured as a receiver is aliased – MCASP_RBUF0 through MCASP_RBUF15 in the CFG port address space. For example, the XRBUF3 receive buffer is mapped as the MCASP_RBUF3 register.

Accessing the XRBUF through the DATA port (see Section 11.9.4.10.1.3) is different than CFG port accesses because the DATA port access demands the same physical address, regardless of transfer direction or current channel index , while accessing through the peripheral configuration port – CFG, the DMA, ARMSS or DSP must provide the exact MCASP_XBUF0 through MCASP_XBUF15 or MCASP_RBUF0 through MCASP_RBUF15 address upon accessing n-th serializer TX or RX buffer, respectively. For more details about MCASP_XBUF0 through MCASP_XBUF15 and MCASP_RBUF0 through MCASP_RBUF15 addresses corresponding to McASP CFG port, see Table 11-1246, *McASP Configuration Space Registers*.

11.9.4.10.1.5 Using the device CPUs for McASP Servicing

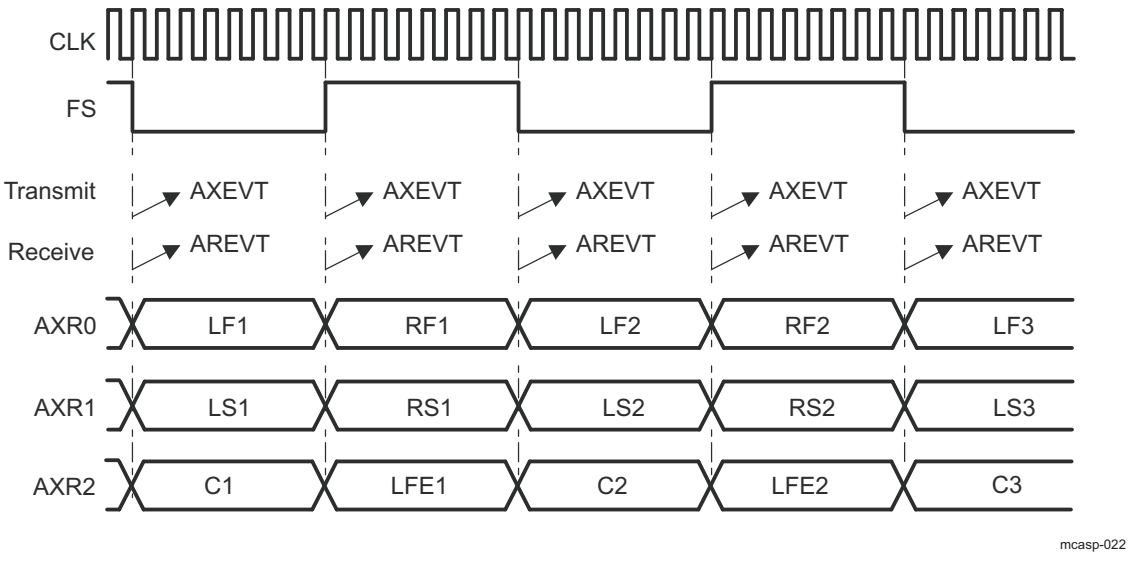
The device CPUs can be used to service the McASP transmit channels through interrupts (upon MCASPi_X_INTR_REQ and MCASPi_R_INTR_REQ interrupts). Another way to service the transmit and receive channels, a polling of the XDATA bit in the MCASP_XSTAT register and RDATA bit in the MCASP_RSTAT register can be performed by device CPUs, respectively. As discussed in Section 11.9.4.10.1.3, *Transfers Through the Data Port (DATA)*, and Section 11.9.4.10.1.4, *Transfers Through the Configuration Bus (CFG)* , the device CPUs can access McASP XRBUF serializer buffer through their corresponding DATA and CFG port locations.

To use the device CPUs to service the McASP through interrupts, the XDATA/RDATA bit must be enabled in the respective MCASP_XINTCTL/MCASP_RINTCTL registers, to generate interrupts MCASPi_X_INTR_REQ /MCASPi_R_INTR_REQ to the device CPUs upon data ready

11.9.4.10.1.6 Using the DMA for McASP Servicing

The typical scenario is to use the DMA to service the McASP transmit and receive logic through the DATA port, although the DMA can also service the McASP through the configuration bus (CFG). The transfer passes through integrated AFIFO transmit/receive buffer. If AFIFO is enabled, DMA requests are collected and fed to DMA system (see Figure 11-583). The data transfer is managed by the AFIFO according to generated transmit and receive events in the McASP and data is fed to transmit buffers and fetched from receive buffers as described in Section 11.9.4.11, *McASP Audio FIFO (AFIFO)*. The generation of transmit and receive request is described below. After generation of transmit/receive DMA events from McASP module, these events are collected in AFIFO and on specific AFIFO conditions described in Section 11.9.4.11, *McASP Audio FIFO (AFIFO)*, the requests (transmit or receive) are forwarded to the DMA system via MCASPi_XEVENT_DREQ and MCASPi_REVENT_DREQ outputs. If the AFIFO is disabled (default state) it is transperrant for the McASP module and all request are directly sent to the DMA system.

Figure 11-594. McASP DMA Transmit and Receive Event in an Audio Example – One Event



In transmit mode, the DMA event – AXEVT (MCASPi_XEVENT_DREQ output), which is triggered upon each XDATA transition from 0 to 1, is used to service the McASP TXBUFn transmit buffers. In receive mode, the DMA event AREVT (MCASPi_REVENT_DREQ output) which is triggered upon each RDATA transition from 0 to 1, is used to service the McASP RXBUFn receive buffers.

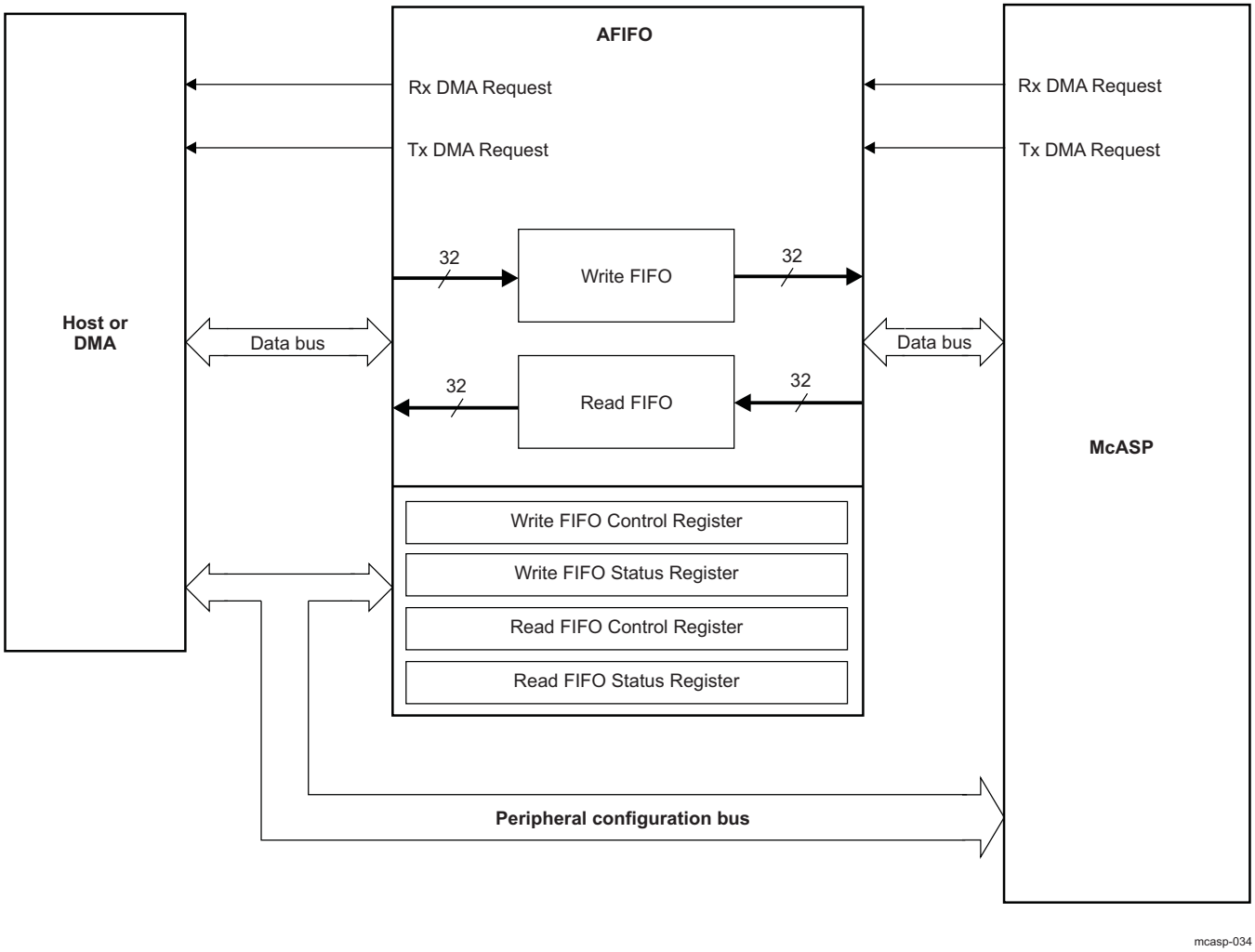
Figure 11-594 is an example of an audio system with six audio channels (LF, RF, LS, RS, C and LFE) transmitted or received through the McASP signals – AXR0, AXR1 and AXR2. It shows the points at which events AXEVT/AREVT are triggered.

In Figure 11-594, a Tx DMA event AXEVT is triggered on each time slot. In the example, AXEVT is triggered for each of the transmit audio channel time slot (time slot for channels LF, LS, and C; and time slot for channels RF, RS, LFE). Transmit DMA events are generated automatically upon transmit data ready, provided that DMA TX requests generation is enabled in the MCASP_XEVTCTL register. Similarly, Rx DMA event AREVT is triggered for each of the receive audio channel time slot. Receive DMA events are generated automatically upon receive data ready, provided that DMA RX requests generation is enabled in the MCASP_REVTCTL register.

11.9.4.11 McASP Audio FIFO (AFIFO)

The AFIFO contains two FIFOs: one Read FIFO (RFIFO), and one Write FIFO (WFIFO). The RFIFO and the WFIFO are the same size: 64 32-bit Words. To ensure backward compatibility with existing software, both the Read and Write FIFOs are disabled by default. See Figure 11-595 for a high-level block diagram of the AFIFO. The AFIFO may be enabled/disabled and configured via the MCASP_WFIFOCTL and MCASP_RFIFOCTL registers. Note that if the Read or Write FIFO is to be enabled, it must be enabled prior to initializing the receive/transmit section of the McASP.

Figure 11-595. McASP Audio FIFO (AFIFO) Block Diagram



11.9.4.11.1 AFIFO Data Transmission

When the Write FIFO is disabled, transmit DMA requests pass through directly from the McASP to the host/DMA controller. Whether the WFIFO is enabled or disabled, the McASP generates transmit DMA requests as needed; the AFIFO is “invisible” to the McASP. When the Write FIFO is enabled, transmit DMA requests from the McASP are sent to the AFIFO, which in turn generates transmit DMA requests to the host/DMA controller. If the Write FIFO is enabled, upon a transmit DMA request from the McASP, the WFIFO writes WNUMDMA 32-bit words to the McASP if and when there are at least WNUMDMA words in the Write FIFO. If there are not, the WFIFO waits until this condition has been satisfied. At that point, it writes WNUMDMA words to the McASP. (See description for MCASP_WFIFOCTL[7-0] WNUMDMA.) If the host CPU writes to the Write FIFO, independent of a transmit DMA request, the WFIFO will accept host writes until full. After this point, excess data will be discarded. Note that when the WFIFO is first enabled, it will immediately issue a transmit DMA request to the host. This is because it begins in an empty state, and is therefore ready to accept data.

11.9.4.11.1.1 Transmit DMA Event Pacer

The AFIFO may be configured to delay making a transmit DMA request to the host until the Write FIFO has enough space for a specified number of words. In this situation, the number of transmit DMA requests to the host or DMA controller is reduced. If the Write FIFO has space to accept WNUMEVT 32-bit words, it generates a transmit DMA request to the host and then waits for a response. Once WNUMEVT words have been written to the FIFO, it checks again to see if there is space for WNUMEVT 32-bit words. If

there is space, it generates another transmit DMA request to the host, and so on. In this fashion, the Write FIFO will attempt to stay filled. Note that if transmit DMA event pacing is desired, MCASP_WFIFOCTL[15-8] WNUMEVT should be set to a non-zero integer multiple of the value in MCASP_WFIFOCTL[7-0] WNUMDMA. If transmit DMA event pacing is not desired, then the value in MCASP_WFIFOCTL[15-8] WNUMEVT should be set equal to the value in MCASP_WFIFOCTL[7-0] WNUMDMA.

11.9.4.11.2 AFIFO Data Reception

When the Read FIFO is disabled, receive DMA requests pass through directly from McASP to the host/DMA controller. Whether the RFIFO is enabled or disabled, the McASP generates receive DMA requests as needed; the AFIFO is “invisible” to the McASP. When the Read FIFO is enabled, receive DMA requests from the McASP are sent to the AFIFO, which in turn generates receive DMA requests to the host/DMA controller. If the Read FIFO is enabled and the McASP makes a receive DMA request, the RFIFO reads RNUMDMA 32-bit words from the McASP, if and when the RFIFO has space for RNUMDMA words. If it does not, the RFIFO waits until this condition has been satisfied; at that point, it reads RNUMDMA words from the McASP. (See description for MCASP_RFIFOCTL[7-0] RNUMDMA.) If the host CPU reads the Read FIFO, independent of a receive DMA request, and the RFIFO at that time contains less than RNUMEVT words, those words will be read correctly, emptying the FIFO.

11.9.4.11.2.1 Receive DMA Event Pacer

The AFIFO may be configured to delay making a receive DMA request to the host until the Read FIFO contains a specified number of words. In this situation, the number of receive DMA requests to the host or DMA controller is reduced. If the Read FIFO contains at least RNUMEVT 32-bit words, it generates a receive DMA request to the host and then waits for a response. Once RNUMEVT 32-bit words have been read from the RFIFO, the RFIFO checks again to see if it contains at least another RNUMEVT words. If it does, it generates another receive DMA request to the host, and so on. In this fashion, the Read FIFO will attempt to stay empty. Note that if receive DMA event pacing is desired, MCASP_RFIFOCTL[15-8] RNUMEVT should be set to a non-zero integer multiple of the value in MCASP_RFIFOCTL[7-0] RNUMDMA. If receive DMA event pacing is not desired, then the value in MCASP_RFIFOCTL[15-8] RNUMEVT should be set equal to the value in MCASP_RFIFOCTL[7-0] RNUMDMA.

11.9.4.11.3 Arbitration Between Transmit and Receive DMA Requests

If both the WFIFO and the RFIFO are enabled and a transmit DMA request and receive DMA request occur simultaneously, priority is given to the transmit DMA request. Once a transfer is in progress, it is allowed to complete. If only the WFIFO is enabled and a transmit DMA request and receive DMA request occur simultaneously, priority is given to the transmit DMA request. Once a transfer is in progress, it is allowed to complete. If only the RFIFO is enabled and a transmit DMA request and receive DMA request occur simultaneously, priority is given to the receive DMA request. Once a transfer is in progress, it is allowed to complete.

11.9.4.12 McASP Events and Interrupt Requests

Table 11-1243 lists all the transmit event flags. Table 11-1244 lists all the receive event flags. Source of each of these TX/RX events can be a TX/RX channel from any McASPi serializer configured as transmitter or receiver respectively.

Table 11-1243. McASP TX Events[(1)]

Event Mask	Event Flag	Map to [(2)]	Description
MCASP_XINTCTL[0] XUNDRN	MCASP_XSTAT[0] XUNDRN	MCASPi_X_INTR_REQ	Transmit buffer underrun
MCASP_XINTCTL[1] XSYNCERR	MCASP_XSTAT[1] XSYNCERR	MCASPi_X_INTR_REQ	Unexpected transmit frame sync
MCASP_XINTCTL[2] XCKFAIL	MCASP_XSTAT[2] XCKFAIL	MCASPi_X_INTR_REQ	Transmit clock failure
MCASP_XINTCTL[3] XDMAERR	MCASP_XSTAT[7] XDMAERR	MCASPi_X_INTR_REQ	DATA port transmit error
MCASP_XINTCTL[4] XLAST	MCASP_XSTAT[4] XLAST	MCASPi_X_INTR_REQ	Transmit last slot interrupt
MCASP_XINTCTL[5] XDATA	MCASP_XSTAT[5] XDATA	MCASPi_X_INTR_REQ	Transmit data-ready interrupt

(1) Global events for all transmitting serializers in a single McASPi module.
(2) Every McASPi module (where i = 0, 1, 2) generates separate IRQ event.

Table 11-1243. McASP TX Events[1] (continued)

Event Mask	Event Flag	Map to [2]	Description
MCASP_XINTCTL[7] XSTAFRM	MCASP_XSTAT[6] XSTAFRM	MCASPi_X_INTR_REQ	Transmit start of frame interrupt
NA	MCASP_XSTAT[8] XERR	NA	OR-event of all Tx-error events: (XDMAERR | XCKFAIL | XUNDRN | XSYNCERR). It is cleared ONLY when all error flags are cleared
NA	MCASP_XSTAT[3] XTDMSLOT	NA	Qualifies the current TDM slot as an odd or an even slot.

Table 11-1244. McASP RX Events[1]

Event Mask	Event Flag	Map to[2]	Description
MCASP_RINTCTL[0] ROVRN	MCASP_RSTAT[0] ROVRN	MCASPi_R_INTR_REQ	Receive buffer overrun
MCASP_RINTCTL[1] RSYNCERR	MCASP_RSTAT[1] RSYNCERR	MCASPi_R_INTR_REQ	Unexpected receive frame sync
MCASP_RINTCTL[2] RCKFAIL	MCASP_RSTAT[2] RCKFAIL	MCASPi_R_INTR_REQ	Receive clock failure
MCASP_RINTCTL[3] RDMAERR	MCASP_RSTAT[7] RDMAERR	MCASPi_R_INTR_REQ	DATA port receive error
MCASP_RINTCTL[4] RLAST	MCASP_RSTAT[4] RLAST	MCASPi_R_INTR_REQ	Receive last slot
MCASP_RINTCTL[5] RDATA	MCASP_RSTAT[5] RDATA	MCASPi_R_INTR_REQ	Receive data-ready
MCASP_RINTCTL[7] RSTAFRM	MCASP_RSTAT[6] RSTAFRM	MCASPi_R_INTR_REQ	Receive start of frame
NA	MCASP_RSTAT[8] RERR	NA	OR-event of all Rx-error events: (RDMAERR | RCKFAIL | ROVRN | RSYNCERR). RERR event is cleared once all error flags are cleared.
NA	MCASP_RSTAT[3] RTDMSLOT	NA	Qualifies the current TDM slot as an odd or an even slot.

(1) Global events for all receiving serializers in a single McASPi module. These events and masks are available in same format for every McASPi module

(2) Every McASPi module (where i = 0, 1, 2) generates separate IRQ event.

Software has to read the MCASP_XSTAT/MCASP_RSTAT register to determine which event occurs at a global level for McASP Tx/Rx logic. In addition user software has to scan the XRDY/RRDY read-only flags in the MCASP_SRCTL0 through MCASP_SRCTL15 registers to determine which active serializer is the actual source of the event.

A Tx interrupt line (MCASPi_X_INTR_REQ, i = 0, 1, 2) is asserted (active high) when one of the MCASP_XSTAT notified events occurs, provided that it is enabled in its corresponding MCASP_XINTCTL bit. Similarly, a Rx interrupt line (MCASPi_R_INTR_REQ, i = 0, 1, 2) is asserted (active high) when one of MCASP_RSTAT notified events occurs, provided that it is enabled in its corresponding MCASP_RINTCTL bit. See also Section 11.9.4.12.5, *Multiple Interrupts*, and Section 11.9.4.10.1, *Data Ready Status and Event/Interrupt Generation*.

11.9.4.12.1 Transmit Data Ready Event and Interrupt

The transmit data-ready interrupt (XDATA) is generated if XDATA is 1 in the MCASP_XSTAT register and XDATA is enabled in MCASP_XINTCTL. Section 11.9.4.10.1, *Data Ready Status and Event/Interrupt Generation*, provides details on when XDATA is set in the MCASP_XSTAT register.

A transmit-start-of-frame interrupt (XSTAFRM) is triggered by the recognition of a transmit frame sync.

A transmit-last-slot interrupt (XLAST) is a qualified version of the data-ready interrupt (XDATA). It has the same behavior than the data-ready interrupt, but is further qualified by having the data requested belonging to the last slot (the slot that just ended is the next-to-last TDM slot, the current slot is the last slot).

11.9.4.12.2 Receive Data Ready Event and Interrupt

The receive data-ready interrupt (RDATA) is generated if RDATA is 1 in the MCASP_RSTAT register and RDATA is enabled in MCASP_RINTCTL. Section 11.9.4.10.1, *Data Ready Status and Event/Interrupt Generation*, provides details on when RDATA flag is set in the MCASP_RSTAT register.

A receiver start of frame (RSTAFRM) interrupt is triggered by the recognition of a receiver frame sync.

A receiver last slot (RLAST) interrupt is a qualified version of the data ready interrupt (RDATA). It has the same behavior as the data ready interrupt, but is further qualified by having the data in the buffer come from the last TDM time slot (the slot that just ended was last TDM slot).

11.9.4.12.3 Error Interrupt

Upon detection, the following error conditions generate interrupt flags:

In the transmit status register (MCASP_XSTAT):

- Transmit underrun (XUNDRN)
- Unexpected transmit frame sync (XSYNCERR)
- Transmit clock failure (XCKFAIL)
- Transmit DATA port error (XDMAERR)

Each interrupt source also has a corresponding enable bit in the transmit interrupt control register (MCASP_XINTCTL). If the enable bit is set, an interrupt is requested when the interrupt flag is set in MCASP_XSTAT. If the enable bit is not set, no interrupt request is generated. However, the interrupt flag may be polled.

In the receive status register (MCASP_RSTAT):

- Receiver overrun (ROVRN)
- Unexpected receive frame sync (RSYNCERR)
- Receive clock failure (RCKFAIL)
- Receive DATA port error (RDMAERR)

Each interrupt source also has a corresponding enable bit in the receive interrupt control register (MCASP_RINTCTL). If the enable bit is set, an interrupt is requested when the interrupt flag is set in MCASP_RSTAT. If the enable bit is not set, no interrupt request is generated. However, the interrupt flag may be polled.

11.9.4.12.4 Audio Mute (MCASP_AMUTE) Function

The McASP includes an automatic audio mute function that asserts in hardware the AMUTE pin to a preprogrammed output state, as selected by the MUTEN bit in the audio mute control register (MCASP_AMUTE). The AMUTE pin is asserted when one of the interrupt flags is set.

When one or more of the errors is detected and enabled, the AMUTE pin is driven to an active state that is selected by MUTEN bit in MCASP_AMUTE register. The active polarity of the MCASP_AMUTE pin is programmable by MUTEN (and the inactive polarity is the opposite of the active polarity). The AMUTE pin remains driven active until software clears all the error interrupt flags that are enabled to mute

11.9.4.12.5 Multiple Interrupts

This only applies to interrupts and not to DMA requests. The following terms are defined:

- **Active Interrupt Request:** A flag in MCASP_XSTAT is set and the interrupt is enabled in MCASP_XINTCTL.
- **Outstanding Interrupt Request:** An interrupt request has been issued on one of the McASP transmit interrupt port, but that request has not yet been serviced.
- **Serviced:** The CPUs write to MCASP_XSTAT to clear one or more of the active interrupt request flags.

The first interrupt request to become active for the serializer with the interrupt flag set in MCASP_XSTAT/MCASP_RSTAT and the interrupt enabled in MCASP_XINTCTL/MCASP_RINTCTL generates a request on the McASP transmit or receive interrupt port.

If more than one interrupt request becomes active in the same cycle, a single interrupt request is generated on the McASP transmit or receive interrupt port. Subsequent interrupt requests that become active while the first interrupt request is outstanding do not immediately generate a new request pulse on the McASP transmit or receive interrupt port.

The interrupt is serviced with the CPU writing to MCASP_XSTAT/MCASP_RSTAT. If any interrupt requests are active after the write, a new request is generated on the McASP transmit or receive interrupt port.

One outstanding interrupt request is allowed on each port, so a transmit and a receive interrupt request may both be outstanding at the same time.

11.9.4.13 McASP DMA Requests

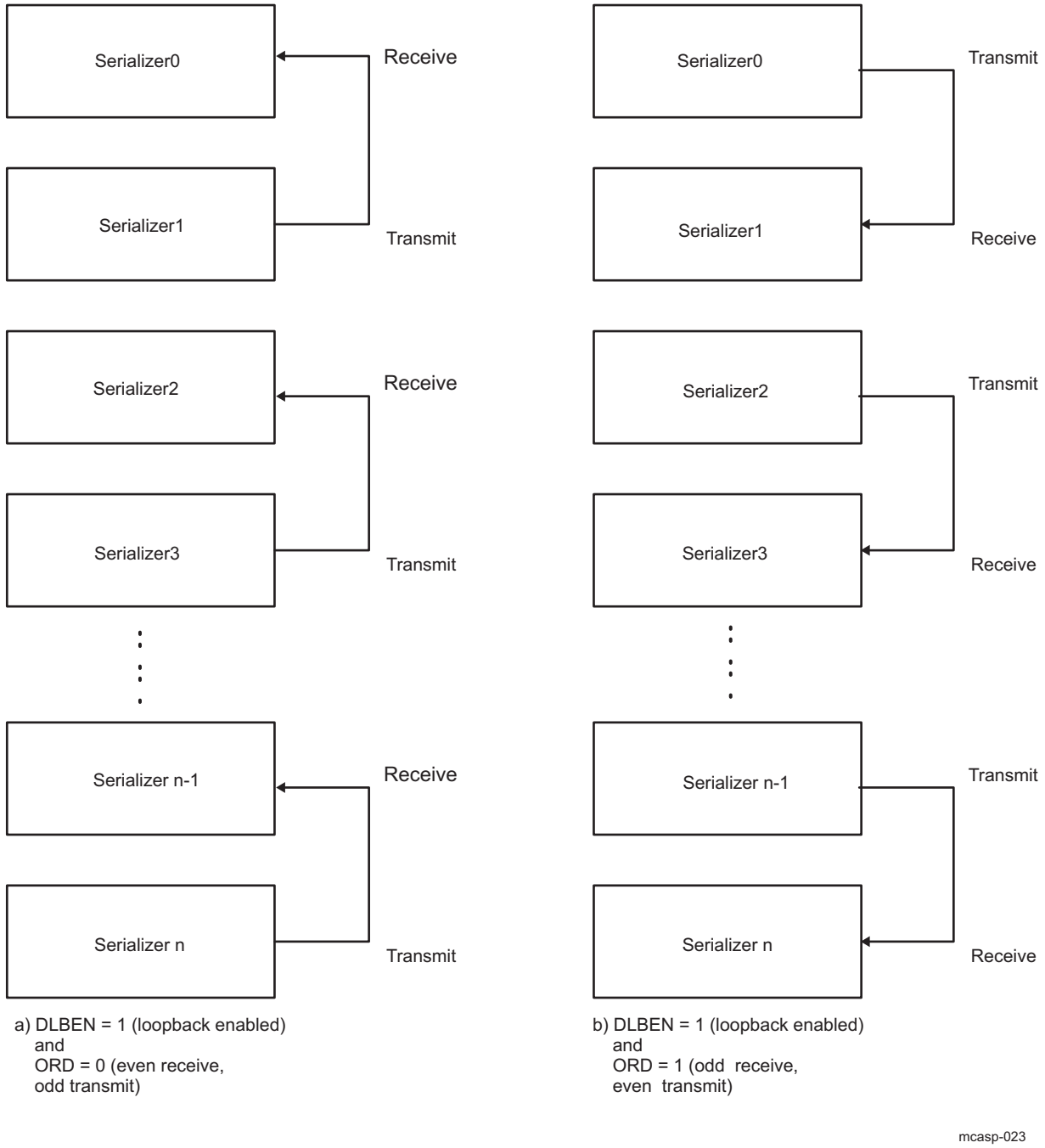
The McASP can generate one DMA request to transmit (MCASPi_XEVENT_DREQ) or receive (MCASPi_REVENT_DREQ) data. A DMA request to transmit data is generated if the XDATDMA bit in the MCASP_XEVTCTL register is cleared. A DMA request to receive data is generated if the RDATDMA bit in the MCASP_REVTCTL register is cleared.

11.9.4.14 McASP Loopback Modes

The McASP features a digital loopback mode (DLB) that allows loopback test transfers in TDM mode between McASP transmitters and receivers within the same device. In loopback mode, the output of a transmit serializer is connected internally to the input of a receive serializer. Therefore, a receiver data can be checked against a transmitter data to ensure that the McASP settings are correct. Digital loopback mode applies to TDM mode only (2 to 32 slots in a frame). It does not apply to DIT mode (XMOD = 0x180) or burst mode (XMOD = 0).

Figure 11-596shows the basic logical connection of the serializers in loopback mode.

Figure 11-596. McASP Serializers Operation in Loopback Mode



Two types of loopback connections are possible, selected by the ORD bit in the digital loopback control register – MCASP_DLBCTL as follows:

- ORD = 0: Outputs of odd serializers are connected to inputs of even serializers. If this mode is selected, the odd serializers must be configured as transmitters and even serializers as receivers.
- ORD = 1: Outputs of even serializers are connected to inputs of odd serializers. If this mode is selected, the even serializers must be configured as transmitters and odd serializers as receivers.

User can choose in software (bit IOLBEN of the MCASP_DLBCTL) between a McASP module internal loopback and a device I/O level loopback.

When a **McASP internal loopback** is selected (MCASP_DLBCTL[4] IOLBEN=0b0), it is NOT necessary to configure MCASP_PFUNC and MCASP_PDIR registers for McASP pin settings. Nevertheless, data can be optionally made externally visible at the I/O pin of the transmit serializer, if the pin is configured as a McASP output pin by setting the corresponding MCASP_PFUNC bit to 0 (that is to function as McASP, not GPIO) and MCASP_PDIR bit to 1 (output).

When a **device I/O level loopback** is selected (MCASP_DLBCTL[4] IOLBEN=0b1), the MCASP_PFUNC and MCASP_PDIR registers must be configured with the appropriate settings for all AXRn pins, according to ORD bit configuration.

In case of device I/O loopback, the connectivity is externally applied between device pads (that is reaching device I/O buffers).

When In loopback mode, the transmit clock and frame sync are used by both the transmit and receive sections of the McASP. The transmit and receive sections operate synchronously. This is achieved by setting the MODE bitfield of the MCASP_DLBCTL register to 0x1 and the ASYNC bit of the MCASP_ACLKXCTL register to 0b0.

11.9.4.14.1 Loopback Mode Configurations

This is a summary of the settings required for digital loopback mode for TDM format:

- The MCASP_DLBCTL[0] DLBEN bit must be set to 0b1 to enable a loopback mode. It must be kept at 0b0 during normal McASP operation.
- The MCASP_DLBCTL[4] IOLBEN bit must be set to select between internal (McASP local) loopback mode or device I/O level loopback mode.
- The MCASP_DLBCTL[3-2] MODE bitfield must be set to 0x1 for both the transmit and receive sections to use the transmit clock and frame sync generator.
- The MCASP_DLBCTL[1] ORD must be programmed appropriately to select odd or even serializers to be transmitters or receivers.
- The corresponding serializers must be configured accordingly.
- The MCASP_ACLKXCTL[6] ASYNC bit must be cleared to 0b0 to ensure synchronous transmit and receive operations.
- The bitfields MCASP_AFSRCTL[15-7] RMOD and MCASP_AFSXCTL[15-7] XMOD must be set within range (0x2–0x20) to indicate TDM mode.

NOTE: Loopback mode does not apply to DIT or burst mode, because McASP receivers do NOT natively support DIR-reception.

11.9.4.15 McASP Error Reporting

The McASP includes error-checking capability for the serial protocol and data underrun. In addition, the McASP includes a timer that continually measures the high-frequency master clock every 32 AHCLKX clock cycles. The value of the timer can be read to get a measurement of the clock frequency and has a minimum and maximum range setting that can set an error flag if the master clock goes out of a specified range.

When one or more errors (software selectable) are detected, an interrupt can be generated if desired, based on one or more error sources.

11.9.4.15.1 Buffer Underrun Error-Transmitter

A buffer underrun occurs when a serializer is instructed by the transmit state-machine to transfer data from XRBUFn buffer to XRSRn shift register, but the corresponding (MCASP_XBUF0 through MCASP_XBUF15) register has not yet been written with new data since the last time the transfer occurred. When this occurs, the transmit state-machine sets the XUNDRN flag.

An underrun is checked only once per time slot. The MCASP_XSTAT[0] XUNDRN flag is set when an underrun condition occurs. Once set, the XUNDRN flag remains set until the host explicitly writes 1 to the XUNDRN bit to clear it.

In DIT mode, a pair of BMC zeros is shifted out when an underrun occurs (four bit times at 128 bfs). By shifting out a pair of zeros, a clock can be recovered on the receiver. To recover, reset the McASP and restart with the proper initialization.

In TDM mode, during an underrun case, a long stream of zeros are shifted out causing the DACs to mute. To recover, reset the McASP and start again with the proper initialization.

11.9.4.15.2 Buffer Overrun Error-Receiver

A buffer overrun occurs when a serializer is instructed to transfer data from XRSRn shift register to XRBUFn receiver buffer, but the corresponding MCASP_RBUF0 through MCASP_RBUF15 register has not yet been read since the last time the transfer occured. When this occurs, the receiver state machine sets the overrun flag – ROVRN. However, the individual serializer writes over the data in the XRBUFn buffer register (destroying the previous sample) and continues shifting.

An overrun is checked only once per time slot. The MCASP_RSTAT[0] ROVRN flag is set when an overrun condition occurs. It is possible that an overrun occurs on one time slot but then the host catches up and does not cause an overrun on the following time slots. However, once the ROVRN flag is set, it remains set until the host explicitly writes a 1 to the ROVRN bit to clear the ROVRN bit.

11.9.4.15.3 DATA Port Error-Transmitter

A transmit DATA port error, as indicated by the XDMAERR flag in the MCASP_XSTAT register, occurs when the DMA or device CPU writes more words to the DATA port of the McASP than it should.

The MCASP_XSTAT[7] XDMAERR=0b1 indicates that the DMA or device CPU wrote too many words to the McASP DATA port for a given transmit DMA event. Writing too few words results in a transmit underrun error setting XUNDRN in MCASP_XSTAT.

While XDMAERR occurs infrequently, an occurrence indicates a serious loss of synchronization between the McASP and the DMA or device CPU. The McASP transmitter and the DMA must be reinitialized to resynchronize them.

11.9.4.15.4 DATA Port Error-Receiver

A receive DATA port error, as indicated by the RDMAERR flag in the MCASP_RSTAT register, occurs when the DMA or device CPU reads more words from the DATA port of the McASP than it should.

The MCASP_RSTAT[7] RDMAERR indicates that the DMA or device CPU read too many words from the McASP DATA port for a given receive AREVT event. Reading too few words results in a receiver overrun error setting ROVRN in MCASP_RSTAT.

While RDMAERR occurs infrequently, an occurrence indicates a serious loss of synchronization between the McASP and the DMA or device CPU. The McASP receiver and the DMA must be reinitialized to resynchronize them.

11.9.4.15.5 Unexpected Frame Sync Error

An unexpected frame sync occurs in when:

- in burst mode and TDM mode, the next active edge of the frame sync occurs early such that the current slot will not be completed by the time the next slot is scheduled to begin.
- in TDM mode, an unexpected frame sync occurs also if the frame sync does NOT occur exactly during the correct bit clock (not a cycle earlier or later) and before slot 0.

When an unexpected frame sync occurs, there are two possible actions depending upon when the unexpected frame sync occurs:

1. **Early:** An early unexpected frame sync occurs when the McASP is in the process of completing the current frame and a new frame sync is detected (not including overlap that occurs due to a 1 or 2 bit frame sync delay). When an early unexpected frame sync occurs:
   - Error event flag is set (XSYNCERR, if an unexpected transmit frame sync occurs; RSYNCERR, if an unexpected receive frame sync occurs).
   - Current frame is not resynchronized. The number of bits in the current frame is completed. The

next frame sync, which occurs after the current frame is completed, will be resynchronized.

2. **Late:** A late unexpected frame sync occurs when there is a gap or delay between the last bit of the previous frame and the first bit of the next frame. When a late unexpected frame sync occurs (as soon as the gap is detected):
   - Error event flag is set (XSYNCERR, if an unexpected transmit frame sync occurs; RSYNCERR, if an unexpected receive frame sync occurs).
   - Resynchronization occurs upon the arrival of the next frame sync.

NOTE: Late frame sync is detected the same way in burst mode and TDM mode. However, in burst mode, late frame sync is not meaningful and its interrupt enable should not be set.

11.9.4.15.6 Clock Failure Detection

11.9.4.15.6.1 Clock Failure Check Startup

It is initially expected of the clock failure circuits to generate an error until at least one measurement is taken. Therefore, the clock failure interrupts, clock switch, and mute functions should not be enabled immediately, but only after a specific startup procedure.

To start the transmit clock failure check procedure:

1. Configure the transmit clock failure detect logic (XMIN, XMAX, XPS) in the transmit clock check control register (MCASP_XCLKCHK).
2. Clear the transmit clock failure flag (XCKFAIL) in the transmit status register (MCASP_XSTAT).
3. Wait until the first measurement is taken (> 32 AHCLKX clock periods).
4. Verify that no clock failure is detected.
5. Repeat Step 2 through Step 4 until the clock is running and is no longer issuing clock failure errors.
6. After the transmit clock is measured and falls within the acceptable range, the following can be enabled:
   1. The transmit clock failure interrupt enable bit (XCKFAIL) in the transmitter interrupt control register (MCASP_XINTCTL)

To start the receive clock failure check procedure:

1. Configure receive clock failure detect logic (RMIN, RMAX, RPS) in the receive clock check control register (MCASP_RCLKCHK).
2. Clear receive clock failure flag (RCKFAIL) in the receive status register (MCASP_RSTAT).
3. Wait until first measurement is taken (> 32 AHCLKR clock periods).
4. Verify no clock failure is detected.
5. Repeat steps 2–4 until clock is running and is no longer issuing clock failure errors.
6. After the receive clock is measured and falls within the acceptable range, the following may be enabled:
   1. the receive clock failure (RCKFAIL) interrupt enable bit in the receive interrupt control register (MCASP_RINTCTL)

11.9.4.15.6.2 Transmit Clock Failure Check and Recovery

The transmit clock failure check circuit (see Figure 11-597) works off the internal McASP interface clock and the external high-frequency serial clock (AHCLKX). It continually counts the number of interface clocks for every 32 high-rate serial clock (AHCLKX) periods, and stores the count in XCNT of the transmit clock check control register (MCASP_XCLKCHK) every 32 high-rate serial clock cycles.

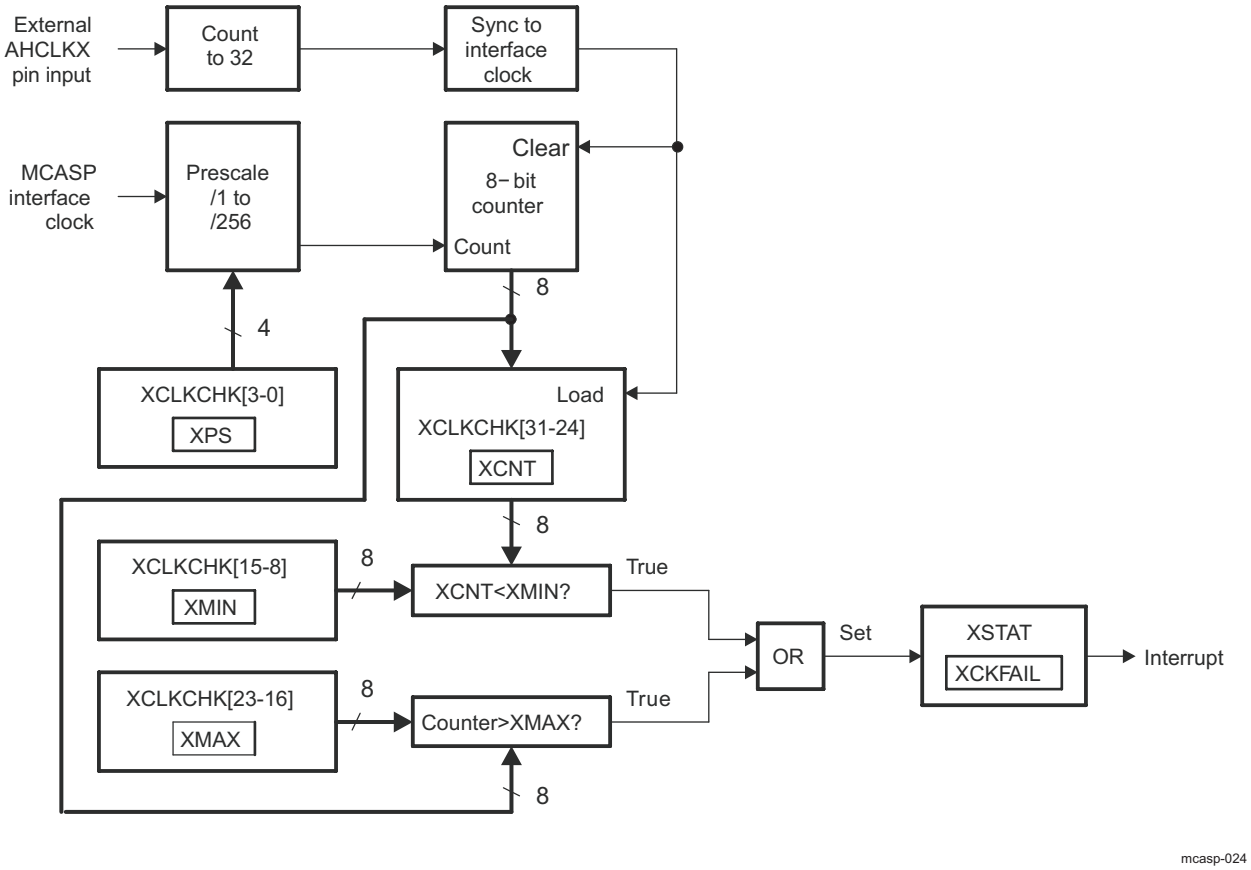
The logic compares the count against a user-defined minimum allowable boundary (XMIN), and automatically flags an interrupt (XCKFAIL in MCASP_XSTAT) when an out-of-range condition occurs. An out-of-range minimum condition occurs when the count is less than XMIN. The logic continually compares the current count (from the running interface clock counter) to the maximum allowable boundary (XMAX). This is so that if the external clock completely stops, the counter value is not copied to XCNT. An out-of-range maximum condition occurs when the count is greater than XMAX. The XMIN and XMAX fields are 8-bit unsigned values, and the comparison is performed using unsigned arithmetic.

An out-of-range count may indicate that an unstable clock was detected or that the audio source has changed and a new sample rate is being used.

For the transmit clock failure check circuit to operate correctly, the high-frequency serial clock divider must be taken out of reset.

If a clock failure is detected, the transmit clock failure flag (XCKFAIL) in MCASP_XSTAT is set. This causes an interrupt if the transmit clock failure interrupt enable bit (XCKFAIL) in MCASP_XINTCTL is set.

Figure 11-597. McASP Transmit Clock Failure Detection Circuit Block Diagram



11.9.4.15.6.3 Receive Clock Failure Check and Recovery

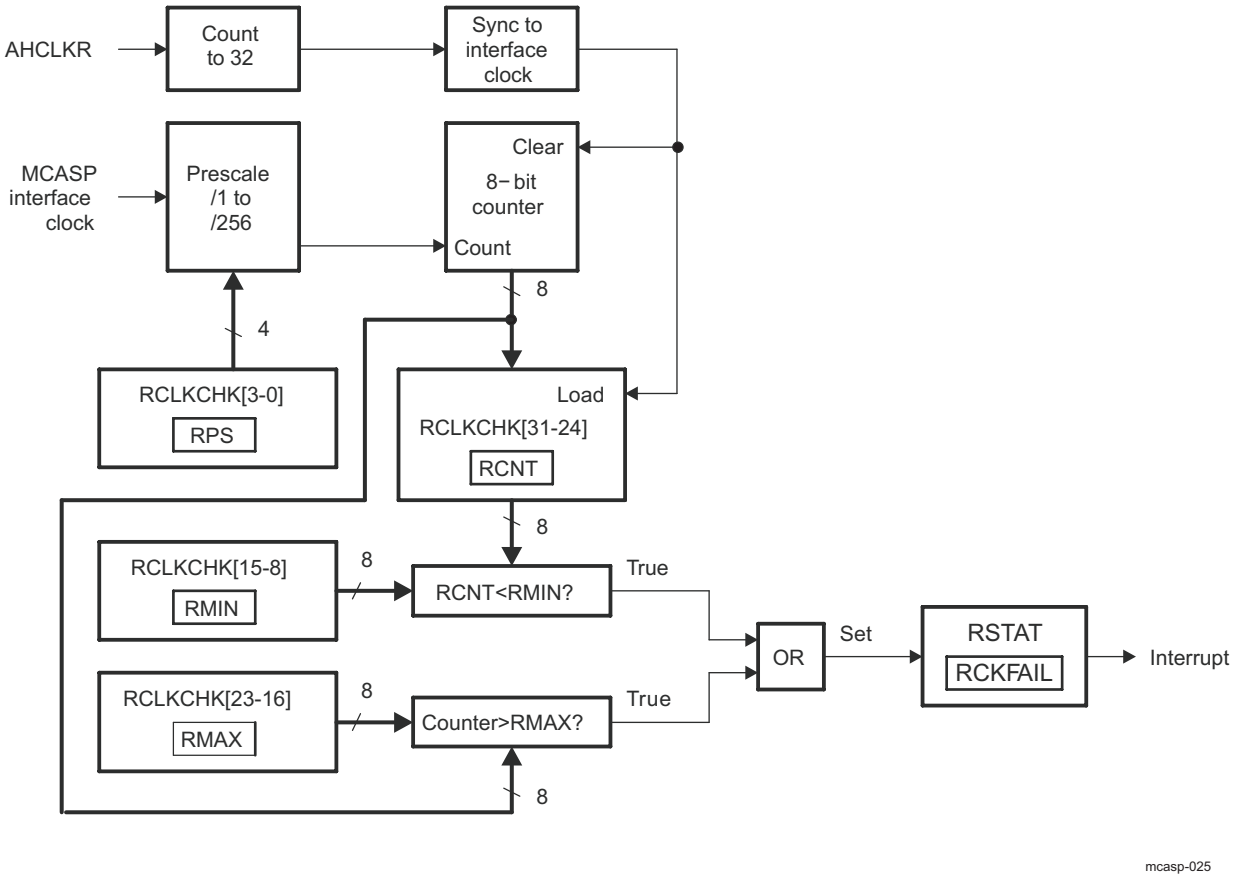
The receive clock failure check circuit (see Figure 11-598) works off both the internal McASP interface clock and the high-frequency serial clock (AHCLKR) coming from the device clock generator. It continually counts the number of interface clocks for every 32 high rate serial clock (AHCLKR) periods, and stores the count in RCNT of the receive clock check control register (MCASP_RCLKCHK) every 32 high rate serial clock cycles.

The logic compares the count against a user-defined minimum allowable boundary (RMIN) and automatically flags an event (RCKFAIL in MCASP_RSTAT) when an out-of-range condition occurs. An out-of-range minimum condition occurs when the count is smaller than RMIN. The logic continually compares the current count (from the running interface clock counter) against the maximum allowable boundary (RMAX). This is in case the external clock completely stops, so that the counter value is not copied to RCNT. An out-of-range maximum condition occurs when the count is greater than RMAX. Note that the RMIN and RMAX fields are 8-bit unsigned values, and the comparison is performed using unsigned arithmetic.

An out-of-range count may indicate either that an unstable clock was detected or that the audio source has changed and a new sample rate is being used.

In order for the receive clock failure check circuit to operate correctly, the high-frequency serial clock divider must be taken out of reset.

Figure 11-598. McASP Receive Clock Failure Detection Circuit Block Diagram



11.9.5 McASP Registers

Table 11-1246 lists the memory-mapped registers for the McASP. All register offset addresses not listed in Table 11-1246 should be considered as reserved locations and the register contents should not be modified.

Table 11-1245. McASP Instances

Instance	Base Address
MCASP_0__CFG	0234 0000h
MCASP_0__FIFO_CFG	0234 0400h
MCASP_0_SLV	2180 4000h
MCASP_1__CFG	0234 2000h

Table 11-1245. McASP Instances (continued)

Instance	Base Address
MCASP_1__FIFO_CFG	0234 2400h
MCASP_1_SLV	2180 4400h
MCASP_2__CFG	0234 4000h
MCASP_2__FIFO_CFG	0234 4400h
MCASP_2_SLV	2180 4800h

Table 11-1246. McASP Configuration Space Registers

Offset	Acronym	Register Name	MCASP_0__CFG Physical Address	MCASP_1__CFG Physical Address	MCASP_2__CFG Physical Address	Section
0h	MCASP_REV	Revision Identification Register	0234 0000h	0234 2000h	0234 4000h	Section 11.9.5.1
4h	MCASP_PWRIDLESYSCONFIG	Power Idle SYSCONFIG Register	0234 0004h	0234 2004h	0234 4004h	Section 11.9.5.2
10h	MCASP_PFUNC	Pin Function Register	0234 0010h	0234 2010h	0234 4010h	Section 11.9.5.3
14h	MCASP_PDIR	Pin Direction Register	0234 0014h	0234 2014h	0234 4014h	Section 11.9.5.4
18h	MCASP_PDOUT	Pin Data Output Register	0234 0018h	0234 2018h	0234 4018h	Section 11.9.5.5
1Ch	MCASP_PDIN	Pin Data Input Register	0234 001Ch	0234 201Ch	0234 401Ch	Section 11.9.5.6
1Ch	MCASP_PDSET	Pin Data Set Register	0234 001Ch	0234 201Ch	0234 401Ch	Section 11.9.5.7
20h	MCASP_PDCLR	Pin Data Clear Register	0234 0020h	0234 2020h	0234 4020h	Section 11.9.5.8
44h	MCASP_GBLCTL	Global Control Register	0234 0044h	0234 2044h	0234 4044h	Section 11.9.5.9
48h	MCASP_AMUTE	Audio Mute Control Register	0234 0048h	0234 2048h	0234 4048h	Section 11.9.5.10
4Ch	MCASP_DLBCTL	Digital Loopback Control Register	0234 004Ch	0234 204Ch	0234 404Ch	Section 11.9.5.11
50h	MCASP_DITCTL	DIT Mode Control Register	0234 0050h	0234 2050h	0234 4050h	Section 11.9.5.12
60h	MCASP_RGBLCTL	Receiver Global Control Register	0234 0060h	0234 2060h	0234 4060h	Section 11.9.5.13
64h	MCASP_RMASK	Receive Format Unit Bit Mask Register	0234 0064h	0234 2064h	0234 4064h	Section 11.9.5.14
68h	MCASP_RFMT	Receive Bit Stream Format Register	0234 0068h	0234 2068h	0234 4068h	Section 11.9.5.15
6Ch	MCASP_AFSRCTL	Receive Frame Sync Control Register	0234 006Ch	0234 206Ch	0234 406Ch	Section 11.9.5.16
70h	MCASP_ACLKRCTL	Receive Clock Control Register	0234 0070h	0234 2070h	0234 4070h	Section 11.9.5.17
74h	MCASP_AHCLKRCTL	Receive High-Frequency Clock Control Register	0234 0074h	0234 2074h	0234 4074h	Section 11.9.5.18
78h	MCASP_RTDM	Receive TDM Time Slot 0-31 Register	0234 0078h	0234 2078h	0234 4078h	Section 11.9.5.19
7Ch	MCASP_RINTCTL	Receiver Interrupt Control Register	0234 007Ch	0234 207Ch	0234 407Ch	Section 11.9.5.20
80h	MCASP_RSTAT	Receiver Status Register	0234 0080h	0234 2080h	0234 4080h	Section 11.9.5.21
84h	MCASP_RSLOT	Current Receive TDM Time Slot Register	0234 0084h	0234 2084h	0234 4084h	Section 11.9.5.22
88h	MCASP_RCLKCHK	Receive Clock Check Control Register	0234 0088h	0234 2088h	0234 4088h	Section 11.9.5.23
8Ch	MCASP_REVTCTL	Receiver DMA Event Control Register	0234 008Ch	0234 208Ch	0234 408Ch	Section 11.9.5.24
A0h	MCASP_XGBLCTL	Transmitter Global Control Register	0234 00A0h	0234 20A0h	0234 40A0h	Section 11.9.5.25

Table 11-1246. McASP Configuration Space Registers (continued)

Offset	Acronym	Register Name	MCASP_0__ CFG Physical Address	MCASP_1__ CFG Physical Address	MCASP_2__ CFG Physical Address	Section
A4h	MCASP_XMASK	Transmit Format Unit Bit Mask Register	0234 00A4h	0234 20A4h	0234 40A4h	Section 11.9.5.26
A8h	MCASP_XFMT	Transmit Bit Stream Format Register	0234 00A8h	0234 20A8h	0234 40A8h	Section 11.9.5.27
ACh	MCASP_AFSXCTL	Transmit Frame Sync Control Register	0234 00ACh	0234 20ACh	0234 40ACh	Section 11.9.5.28
B0h	MCASP_ACLKXCTL	Transmit Clock Control Register	0234 00B0h	0234 20B0h	0234 40B0h	Section 11.9.5.29
B4h	MCASP_AHCLKXCTL	Transmit High-Frequency Clock Control Register	0234 00B4h	0234 20B4h	0234 40B4h	Section 11.9.5.30
B8h	MCASP_XTDM	Transmit TDM Time Slot 0-31 Register	0234 00B8h	0234 20B8h	0234 40B8h	Section 11.9.5.31
BCh	MCASP_XINTCTL	Transmitter Interrupt Control Register	0234 00BCh	0234 20BCh	0234 40BCh	Section 11.9.5.32
C0h	MCASP_XSTAT	Transmitter Status Register	0234 00C0h	0234 20C0h	0234 40C0h	Section 11.9.5.33
C4h	MCASP_XSLOT	Current Transmit TDM Time Slot Register	0234 00C4h	0234 20C4h	0234 40C4h	Section 11.9.5.34
C8h	MCASP_XCLKCHK	Transmit Clock Check Control Register	0234 00C8h	0234 20C8h	0234 40C8h	Section 11.9.5.35
CCh	MCASP_XEVTCTL	Transmitter DMA Event Control Register	0234 00CCh	0234 20CCh	0234 40CCh	Section 11.9.5.36
100h	MCASP_DITCSRA0	Left (Even TDM Time Slot) Channel Status Registers (DIT Mode)	0234 0100h	0234 2100h	0234 4100h	Section 11.9.5.37
104h	MCASP_DITCSRA1	Left (Even TDM Time Slot) Channel Status Registers (DIT Mode)	0234 0104h	0234 2104h	0234 4104h	Section 11.9.5.38
108h	MCASP_DITCSRA2	Left (Even TDM Time Slot) Channel Status Registers (DIT Mode)	0234 0108h	0234 2108h	0234 4108h	Section 11.9.5.39
10Ch	MCASP_DITCSRA3	Left (Even TDM Time Slot) Channel Status Registers (DIT Mode)	0234 010Ch	0234 210Ch	0234 410Ch	Section 11.9.5.40
110h	MCASP_DITCSRA4	Left (Even TDM Time Slot) Channel Status Registers (DIT Mode)	0234 0110h	0234 2110h	0234 4110h	Section 11.9.5.41
114h	MCASP_DITCSRA5	Left (Even TDM Time Slot) Channel Status Registers (DIT Mode)	0234 0114h	0234 2114h	0234 4114h	Section 11.9.5.42
118h	MCASP_DITCSRB0	Right (Odd TDM Time Slot) Channel Status Registers (DIT Mode)	0234 0118h	0234 2118h	0234 4118h	Section 11.9.5.43
11Ch	MCASP_DITCSRB1	Right (Odd TDM Time Slot) Channel Status Registers (DIT Mode)	0234 011Ch	0234 211Ch	0234 411Ch	Section 11.9.5.44
120h	MCASP_DITCSRB2	Right (Odd TDM Time Slot) Channel Status Registers (DIT Mode)	0234 0120h	0234 2120h	0234 4120h	Section 11.9.5.45
124h	MCASP_DITCSRB3	Right (Odd TDM Time Slot) Channel Status Registers (DIT Mode)	0234 0124h	0234 2124h	0234 4124h	Section 11.9.5.46
128h	MCASP_DITCSRB4	Right (Odd TDM Time Slot) Channel Status Registers (DIT Mode)	0234 0128h	0234 2128h	0234 4128h	Section 11.9.5.47

Table 11-1246. McASP Configuration Space Registers (continued)

Offset	Acronym	Register Name	MCASP_0__CFG Physical Address	MCASP_1__CFG Physical Address	MCASP_2__CFG Physical Address	Section
12Ch	MCASP_DITCSRB5	Right (Odd TDM Time Slot) Channel Status Registers (DIT Mode)	0234 012Ch	0234 212Ch	0234 412Ch	Section 11.9.5.48
130h	MCASP_DITUDRA0	Left (Even TDM Time Slot) Channel User Data Registers (DIT Mode)	0234 0130h	0234 2130h	0234 4130h	Section 11.9.5.49
134h	MCASP_DITUDRA1	Left (Even TDM Time Slot) Channel User Data Registers (DIT Mode)	0234 0134h	0234 2134h	0234 4134h	Section 11.9.5.50
138h	MCASP_DITUDRA2	Left (Even TDM Time Slot) Channel User Data Registers (DIT Mode)	0234 0138h	0234 2138h	0234 4138h	Section 11.9.5.51
13Ch	MCASP_DITUDRA3	Left (Even TDM Time Slot) Channel User Data Registers (DIT Mode)	0234 013Ch	0234 213Ch	0234 413Ch	Section 11.9.5.52
140h	MCASP_DITUDRA4	Left (Even TDM Time Slot) Channel User Data Registers (DIT Mode)	0234 0140h	0234 2140h	0234 4140h	Section 11.9.5.53
144h	MCASP_DITUDRA5	Left (Even TDM Time Slot) Channel User Data Registers (DIT Mode)	0234 0144h	0234 2144h	0234 4144h	Section 11.9.5.54
148h	MCASP_DITUDRB0	Right (Odd TDM Time Slot) Channel User Data Registers (DIT Mode)	0234 0148h	0234 2148h	0234 4148h	Section 11.9.5.55
14Ch	MCASP_DITUDRB1	Right (Odd TDM Time Slot) Channel User Data Registers (DIT Mode)	0234 014Ch	0234 214Ch	0234 414Ch	Section 11.9.5.56
150h	MCASP_DITUDRB2	Right (Odd TDM Time Slot) Channel User Data Registers (DIT Mode)	0234 0150h	0234 2150h	0234 4150h	Section 11.9.5.57
154h	MCASP_DITUDRB3	Right (Odd TDM Time Slot) Channel User Data Registers (DIT Mode)	0234 0154h	0234 2154h	0234 4154h	Section 11.9.5.58
158h	MCASP_DITUDRB4	Right (Odd TDM Time Slot) Channel User Data Registers (DIT Mode)	0234 0158h	0234 2158h	0234 4158h	Section 11.9.5.59
15Ch	MCASP_DITUDRB5	Right (Odd TDM Time Slot) Channel User Data Registers (DIT Mode)	0234 015Ch	0234 215Ch	0234 415Ch	Section 11.9.5.60
180h	MCASP_SRCTL0	Serializer Control Registers	0234 0180h	0234 2180h	0234 4180h	Section 11.9.5.61
184h	MCASP_SRCTL1	Serializer Control Registers	0234 0184h	0234 2184h	0234 4184h	Section 11.9.5.62
188h	MCASP_SRCTL2	Serializer Control Registers	0234 0188h	0234 2188h	0234 4188h	Section 11.9.5.63
18Ch	MCASP_SRCTL3	Serializer Control Registers	0234 018Ch	0234 218Ch	0234 418Ch	Section 11.9.5.64
190h	MCASP_SRCTL4	Serializer Control Registers	0234 0190h	0234 2190h	0234 4190h	Section 11.9.5.65
194h	MCASP_SRCTL5	Serializer Control Registers	0234 0194h	0234 2194h	0234 4194h	Section 11.9.5.66
198h	MCASP_SRCTL6	Serializer Control Registers	0234 0198h	0234 2198h	0234 4198h	Section 11.9.5.67
19Ch	MCASP_SRCTL7	Serializer Control Registers	0234 019Ch	0234 219Ch	0234 419Ch	Section 11.9.5.68

Table 11-1246. McASP Configuration Space Registers (continued)

Offset	Acronym	Register Name	MCASP_0__CFG Physical Address	MCASP_1__CFG Physical Address	MCASP_2__CFG Physical Address	Section
1A0h	MCASP_SRCTL8	Serializer Control Registers	0234 01A0h	0234 21A0h	0234 41A0h	Section 11.9.5.69
1A4h	MCASP_SRCTL9	Serializer Control Registers	0234 01A4h	0234 21A4h	0234 41A4h	Section 11.9.5.70
1A8h	MCASP_SRCTL10	Serializer Control Registers	0234 01A8h	0234 21A8h	0234 41A8h	Section 11.9.5.71
1ACh	MCASP_SRCTL11	Serializer Control Registers	0234 01ACh	0234 21ACh	0234 41ACh	Section 11.9.5.72
1B0h	MCASP_SRCTL12	Serializer Control Registers	0234 01B0h	0234 21B0h	0234 41B0h	Section 11.9.5.73
1B4h	MCASP_SRCTL13	Serializer Control Registers	0234 01B4h	0234 21B4h	0234 41B4h	Section 11.9.5.74
1B8h	MCASP_SRCTL14	Serializer Control Registers	0234 01B8h	0234 21B8h	0234 41B8h	Section 11.9.5.75
1BCh	MCASP_SRCTL15	Serializer Control Registers	0234 01BCh	0234 21BCh	0234 41BCh	Section 11.9.5.76
200h	MCASP_XBUF0	Transmit Buffer Register for Serializers	0234 0200h	0234 2200h	0234 4200h	Section 11.9.5.77
204h	MCASP_XBUF1	Transmit Buffer Register for Serializers	0234 0204h	0234 2204h	0234 4204h	Section 11.9.5.78
208h	MCASP_XBUF2	Transmit Buffer Register for Serializers	0234 0208h	0234 2208h	0234 4208h	Section 11.9.5.79
20Ch	MCASP_XBUF3	Transmit Buffer Register for Serializers	0234 020Ch	0234 220Ch	0234 420Ch	Section 11.9.5.80
210h	MCASP_XBUF4	Transmit Buffer Register for Serializers	0234 0210h	0234 2210h	0234 4210h	Section 11.9.5.81
214h	MCASP_XBUF5	Transmit Buffer Register for Serializers	0234 0214h	0234 2214h	0234 4214h	Section 11.9.5.82
218h	MCASP_XBUF6	Transmit Buffer Register for Serializers	0234 0218h	0234 2218h	0234 4218h	Section 11.9.5.83
21Ch	MCASP_XBUF7	Transmit Buffer Register for Serializers	0234 021Ch	0234 221Ch	0234 421Ch	Section 11.9.5.84
220h	MCASP_XBUF8	Transmit Buffer Register for Serializers	0234 0220h	0234 2220h	0234 4220h	Section 11.9.5.85
224h	MCASP_XBUF9	Transmit Buffer Register for Serializers	0234 0224h	0234 2224h	0234 4224h	Section 11.9.5.86
228h	MCASP_XBUF10	Transmit Buffer Register for Serializers	0234 0228h	0234 2228h	0234 4228h	Section 11.9.5.87
22Ch	MCASP_XBUF11	Transmit Buffer Register for Serializers	0234 022Ch	0234 222Ch	0234 422Ch	Section 11.9.5.88
230h	MCASP_XBUF12	Transmit Buffer Register for Serializers	0234 0230h	0234 2230h	0234 4230h	Section 11.9.5.89
234h	MCASP_XBUF13	Transmit Buffer Register for Serializers	0234 0234h	0234 2234h	0234 4234h	Section 11.9.5.90
238h	MCASP_XBUF14	Transmit Buffer Register for Serializers	0234 0238h	0234 2238h	0234 4238h	Section 11.9.5.91
23Ch	MCASP_XBUF15	Transmit Buffer Register for Serializers	0234 023Ch	0234 223Ch	0234 423Ch	Section 11.9.5.92
280h	MCASP_RBUF0	Receive Buffer Register for Serializers	0234 0280h	0234 2280h	0234 4280h	Section 11.9.5.93
284h	MCASP_RBUF1	Receive Buffer Register for Serializers	0234 0284h	0234 2284h	0234 4284h	Section 11.9.5.94
288h	MCASP_RBUF2	Receive Buffer Register for Serializers	0234 0288h	0234 2288h	0234 4288h	Section 11.9.5.95

Table 11-1246. McASP Configuration Space Registers (continued)

Offset	Acronym	Register Name	MCASP_0__CFG Physical Address	MCASP_1__CFG Physical Address	MCASP_2__CFG Physical Address	Section
28Ch	MCASP_RBUF3	Receive Buffer Register for Serializers	0234 028Ch	0234 228Ch	0234 428Ch	Section 11.9.5.96
290h	MCASP_RBUF4	Receive Buffer Register for Serializers	0234 0290h	0234 2290h	0234 4290h	Section 11.9.5.97
294h	MCASP_RBUF5	Receive Buffer Register for Serializers	0234 0294h	0234 2294h	0234 4294h	Section 11.9.5.98
298h	MCASP_RBUF6	Receive Buffer Register for Serializers	0234 0298h	0234 2298h	0234 4298h	Section 11.9.5.99
29Ch	MCASP_RBUF7	Receive Buffer Register for Serializers	0234 029Ch	0234 229Ch	0234 429Ch	Section 11.9.5.100
2A0h	MCASP_RBUF8	Receive Buffer Register for Serializers	0234 02A0h	0234 22A0h	0234 42A0h	Section 11.9.5.101
2A4h	MCASP_RBUF9	Receive Buffer Register for Serializers	0234 02A4h	0234 22A4h	0234 42A4h	Section 11.9.5.102
2A8h	MCASP_RBUF10	Receive Buffer Register for Serializers	0234 02A8h	0234 22A8h	0234 42A8h	Section 11.9.5.103
2ACh	MCASP_RBUF11	Receive Buffer Register for Serializers	0234 02ACh	0234 22ACh	0234 42ACh	Section 11.9.5.104
2B0h	MCASP_RBUF12	Receive Buffer Register for Serializers	0234 02B0h	0234 22B0h	0234 42B0h	Section 11.9.5.105
2B4h	MCASP_RBUF13	Receive Buffer Register for Serializers	0234 02B4h	0234 22B4h	0234 42B4h	Section 11.9.5.106
2B8h	MCASP_RBUF14	Receive Buffer Register for Serializers	0234 02B8h	0234 22B8h	0234 42B8h	Section 11.9.5.107
2BCh	MCASP_RBUF15	Receive Buffer Register for Serializers	0234 02BCh	0234 22BCh	0234 42BCh	Section 11.9.5.108

Table 11-1247. McASP FIFO Configuration Space Registers

Offset	Acronym	Register Name	MCASP_0__FIFO_CFG Physical Address	MCASP_1__FIFO_CFG Physical Address	MCASP_2__FIFO_CFG Physical Address	Section
1000h	MCASP_WFIFOCTL	Write FIFO Control Register	0234 1400h	0234 3400h	0234 5400h	Section 11.9.5.109
1004h	MCASP_WFIFOSTS	Write FIFO Status Register	0234 1404h	0234 3404h	0234 5404h	Section 11.9.5.110
1008h	MCASP_RFIFOCTL	Read FIFO Control Register	0234 1408h	0234 3408h	0234 5408h	Section 11.9.5.111
100Ch	MCASP_RFIFOSTS	Read FIFO Status Register	0234 140Ch	0234 340Ch	0234 540Ch	Section 11.9.5.112

Table 11-1248. McASP DMA Space Registers

Offset	Acronym	Register Name	MCASP_0_SLV Physical Address	MCASP_1_SLV Physical Address	MCASP_2_SLV Physical Address	Section
N/A[1]	MCASP_XBUF	Tx Buffer Data	2180 4000h	2180 4400h	2180 4800h	Section 11.9.5.113
N/A[1]	MCASP_RBUF	Rx Buffer Data	2180 4000h	2180 4400h	2180 4800h	Section 11.9.5.114

(1) N/A denotes an example for DATA port offset value. Actually, whatever the offset value is added to base address, it is ignored (don't care) when ARMSS / DSP performs accesses to XRBUFn RX/TX buffers through the McASP DATA port.

NOTE: For MCASP_XBUF and MCASP_RBUF buffer accesses through the McASP DATA port, the destination physical address is always the same regardless of current channel index or transfer direction. The MCASP_XFMT[3] XBUSEL bit must be set to 0b0, to allow write transfers through the DATA port. The MCASP_RFMT[3] RBUSEL bit must be set to 0b0, to allow read transfers through the DATA port.

NOTE: The McASP DATA port is exclusively assigned for DMAs / device CPUs accesses to the McASP channels transmit and receive buffer registers. All other McASP module registers must be accessed through the McASP CFG (peripheral) port.

11.9.5.1 MCASP_REV Register (Offset = 0h) [reset = 44300A02h]

MCASP_REV is shown in Figure 11-599 and described in Table 11-1250.

The revision identification register (MCASP_REV) contains identification data for the peripheral.

Table 11-1249. MCASP_REV Instances

Instance	Physical Address
MCASP_0__CFG	0234 0000h
MCASP_1__CFG	0234 2000h
MCASP_2__CFG	0234 4000h

Figure 11-599. MCASP_REV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-44300A02h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1250. MCASP_REV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	44300A02h	Identifies revision of peripheral.

Table 11-1251. Register Call Summary for MCASP_REV

McASP Registers
- McASP Registers: [0]
- MCASP_REV Register (Offset = 0h) [reset = 44300A02h]: [0][1]

11.9.5.2 MCASP_PWRIDLESYSCONFIG Register (Offset = 4h) [reset = 2h]

MCASP_PWRIDLESYSCONFIG is shown in Figure 11-600 and described in Table 11-1253.

Table 11-1252. MCASP_PWRIDLESYSCONFIG Instances

Instance	Physical Address
MCASP_0__CFG	0234 0004h
MCASP_1__CFG	0234 2004h
MCASP_2__CFG	0234 4004h

Figure 11-600. MCASP_PWRIDLESYSCONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			OTHER			IDLEMODE	
R-0h			R/W-0h			R/W-2h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1253. MCASP_PWRIDLESYSCONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5-2	OTHER	R/W	0h	Reserved for future programming.
1-0	IDLEMODE	R/W	2h	Power management Configuration of the local target state management mode. By definition, target can handle read/write transaction as long as it is out of IDLE state. 0h (R/W) = Force-idle mode: local target's idle state follows (acknowledges) the system's idle requests unconditionally, i.e. regardless of the IP module's internal requirements. Backup mode, for debug only. 1h (R/W) = No-idle mode: local target never enters idle state. Backup mode, for debug only. 2h (R/W) = Smart-idle mode: local target's idle state eventually follows (acknowledges) the system's idle requests, depending on the IP module's internal requirements. IP module shall not generate (IRQ- or DMA-request-related) wakeup events. 3h (R/W) = Reserved.

Table 11-1254. Register Call Summary for MCASP_PWRIDLESYSCONFIG

McASP Functional Description • McASP Power Management: [0][1][2]
McASP Registers • McASP Registers: [0] • MCASP_PWRIDLESYSCONFIG Register (Offset = 4h) [reset = 2h]: [0]

11.9.5.3 MCASP_PFUNC Register (Offset = 10h) [reset = 0h]

MCASP_PFUNC is shown in Figure 11-601 and described in Table 11-1256.

The pin function register (MCASP_PFUNC) specifies the function of AXRn, ACLKX, AHCLKX, AFSX, ACLKR, AHCLKR, and AFSR pins as either a McASP pin or a general-purpose input/output (GPIO) pin. CAUTION: Writing a value other than 0 to reserved bits in this register may cause improper device operation.

Table 11-1255. MCASP_PFUNC Instances

Instance	Physical Address
MCASP_0__CFG	0234 0010h
MCASP_1__CFG	0234 2010h
MCASP_2__CFG	0234 4010h

Figure 11-601. MCASP_PFUNC Register

31	30	29	28	27	26	25	24
AFSR	AHCLKR	ACLKR	AFSX	AHCLKX	ACLKX	AMUTE	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				AXR			
R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1256. MCASP_PFUNC Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AFSR	R/W	0h	Determines if AFSR pin functions as McASP or GPIO. 0h (R/W) = Pin functions as McASP pin. 1h (R/W) = Pin functions as GPIO pin.
30	AHCLKR	R/W	0h	Determines if AHCLKR pin functions as McASP or GPIO. 0h (R/W) = Pin functions as McASP pin. 1h (R/W) = Pin functions as GPIO pin.
29	ACLKR	R/W	0h	Determines if ACLKR pin functions as McASP or GPIO. 0h (R/W) = Pin functions as McASP pin. 1h (R/W) = Pin functions as GPIO pin.
28	AFSX	R/W	0h	Determines if AFSX pin functions as McASP or GPIO. 0h (R/W) = Pin functions as McASP pin. 1h (R/W) = Pin functions as GPIO pin.
27	AHCLKX	R/W	0h	Determines if AHCLKX pin functions as McASP or GPIO. 0h (R/W) = Pin functions as McASP pin. 1h (R/W) = Pin functions as GPIO pin.
26	ACLKX	R/W	0h	Determines if ACLKX pin functions as McASP or GPIO. 0h (R/W) = Pin functions as McASP pin. 1h (R/W) = Pin functions as GPIO pin.

Table 11-1256. MCASP_PFUNC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	AMUTE	R/W	0h	Determines if AMUTE pin functions as McASP or GPIO. 0h (R/W) = Pin functions as McASP pin. 1h (R/W) = Pin functions as GPIO pin.
24-4	RESERVED	R	0h	Reserved
3-0	AXR	R/W	0h	Determines if AXRn pin functions as McASP or GPIO. 0h (R/W) = Pin functions as McASP pin. 1h (R/W) = Pin functions as GPIO pin.

Table 11-1257. Register Call Summary for MCASP_PFUNC

McASP Functional Description • McASP Loopback Modes: [0][1][2] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit DIT Clock and Frame-Sync Generation: [0]
McASP Registers • MCASP_SRCTL12 Register (Offset = 1B0h) [reset = 0h]: [0] • MCASP_PDIR Register (Offset = 14h) [reset = 0h]: [0][1][2][3] • MCASP_SRCTL10 Register (Offset = 1A8h) [reset = 0h]: [0] • MCASP_SRCTL11 Register (Offset = 1ACh) [reset = 0h]: [0] • MCASP_PDOUT Register (Offset = 18h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9] • MCASP_SRCTL14 Register (Offset = 1B8h) [reset = 0h]: [0] • MCASP_SRCTL15 Register (Offset = 1BCh) [reset = 0h]: [0] • MCASP_PDSET Register (Offset = 1Ch) [reset = 0h]: [0] • MCASP_PDIN Register (Offset = 1Ch) [reset = 0h]: [0] • MCASP_SRCTL13 Register (Offset = 1B4h) [reset = 0h]: [0] • MCASP_PFUNC Register (Offset = 10h) [reset = 0h]: [0][1] • McASP Registers: [0] • MCASP_PDCLR Register (Offset = 20h) [reset = 0h]: [0] • MCASP_SRCTL0 Register (Offset = 180h) [reset = 0h]: [0] • MCASP_SRCTL1 Register (Offset = 184h) [reset = 0h]: [0] • MCASP_SRCTL2 Register (Offset = 188h) [reset = 0h]: [0] • MCASP_SRCTL3 Register (Offset = 18Ch) [reset = 0h]: [0] • MCASP_SRCTL4 Register (Offset = 190h) [reset = 0h]: [0] • MCASP_SRCTL5 Register (Offset = 194h) [reset = 0h]: [0] • MCASP_SRCTL6 Register (Offset = 198h) [reset = 0h]: [0] • MCASP_SRCTL7 Register (Offset = 19Ch) [reset = 0h]: [0] • MCASP_SRCTL8 Register (Offset = 1A0h) [reset = 0h]: [0] • MCASP_SRCTL9 Register (Offset = 1A4h) [reset = 0h]: [0] • MCASP_AMUTE Register (Offset = 48h) [reset = 0h]: [0] • MCASP_GBLCTL Register (Offset = 44h) [reset = 0h]: [0]

11.9.5.4 MCASP_PDIR Register (Offset = 14h) [reset = 0h]

MCASP_PDIR is shown in Figure 11-602 and described in Table 11-1259.

The pin direction register (MCASP_PDIR) specifies the direction of AXRn, ACLKX, AHCLKX, AFSX, ACLKR, AHCLKR, and AFSR pins as either an input or an output pin. Regardless of the pin function register (MCASP_PFUNC) setting, each MCASP_PDIR bit must be set to 1 for the specified pin to be enabled as an output and each MCASP_PDIR bit must be cleared to 0 for the specified pin to be an input. For example, if the McASP is configured to use an internally-generated bit clock and the clock is to be driven out to the system, the MCASP_PFUNC bit must be cleared to 0 (McASP function) and the MCASP_PDIR bit must be set to 1 (an output). When AXRn is configured to transmit, the MCASP_PFUNC bit must be cleared to 0 (McASP function) and the MCASP_PDIR bit must be set to 1 (an output). Similarly, when AXRn is configured to receive, the MCASP_PFUNC bit must be cleared to 0 (McASP function) and the MCASP_PDIR bit must be cleared to 0 (an input). CAUTION: Writing a value other than 0 to reserved bits in this register may cause improper device operation.

Table 11-1258. MCASP_PDIR Instances

Instance	Physical Address
MCASP_0__CFG	0234 0014h
MCASP_1__CFG	0234 2014h
MCASP_2__CFG	0234 4014h

Figure 11-602. MCASP_PDIR Register

31	30	29	28	27	26	25	24
AFSR	AHCLKR	ACLKR	AFSX	AHCLKX	ACLKX	AMUTE	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				AXR			
R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1259. MCASP_PDIR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AFSR	R/W	0h	Determines if AFSR pin functions as an input or output. 0h (R/W) = Pin functions as input. 1h (R/W) = Pin functions as output.
30	AHCLKR	R/W	0h	Determines if AHCLKR pin functions as an input or output. 0h (R/W) = Pin functions as input. 1h (R/W) = Pin functions as output.
29	ACLKR	R/W	0h	Determines if ACLKR pin functions as an input or output. 0h (R/W) = Pin functions as input. 1h (R/W) = Pin functions as output.
28	AFSX	R/W	0h	Determines if AFSX pin functions as an input or output. 0h (R/W) = Pin functions as input. 1h (R/W) = Pin functions as output.

Table 11-1259. MCASP_PDIR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	AHCLKX	R/W	0h	Determines if AHCLKX pin functions as an input or output. 0h (R/W) = Pin functions as input. 1h (R/W) = Pin functions as output.
26	ACLKX	R/W	0h	Determines if ACLKX pin functions as an input or output. 0h (R/W) = Pin functions as input. 1h (R/W) = Pin functions as output.
25	AMUTE	R/W	0h	Determines if AMUTE pin functions as an input or output. 0h (R/W) = Pin functions as input. 1h (R/W) = Pin functions as output.
24-4	RESERVED	R	0h	Reserved
3-0	AXR	R/W	0h	Determines if AXRn pin functions as an input or output. 0h (R/W) = Pin functions as input. 1h (R/W) = Pin functions as output.

Table 11-1260. Register Call Summary for MCASP_PDIR

McASP Functional Description • McASP Loopback Modes: [0][1][2] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit DIT Clock and Frame-Sync Generation: [0]
McASP Registers • MCASP_PDSET Register (Offset = 1Ch) [reset = 0h]: [0] • MCASP_PDIN Register (Offset = 1Ch) [reset = 0h]: [0] • MCASP_PDIR Register (Offset = 14h) [reset = 0h]: [0][1][2][3][4][5][6] • McASP Registers: [0] • MCASP_AMUTE Register (Offset = 48h) [reset = 0h]: [0] • MCASP_PDCLR Register (Offset = 20h) [reset = 0h]: [0] • MCASP_PDOUT Register (Offset = 18h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9] • MCASP_GBLCTL Register (Offset = 44h) [reset = 0h]: [0]
McASP Environment • McASP Signals: [0]

11.9.5.5 MCASP_PDOUT Register (Offset = 18h) [reset = 0h]

MCASP_PDOUT is shown in Figure 11-603 and described in Table 11-1262.

The pin data output register (MCASP_PDOUT) holds a value for data out at all times, and may be read back at all times. The value held by MCASP_PDOUT is not affected by writing to MCASP_PDIR and MCASP_PFUNC. However, the data value in MCASP_PDOUT is driven out onto the McASP pin only if the corresponding bit in MCASP_PFUNC is set to 1 (GPIO function) and the corresponding bit in MCASP_PDIR is set to 1 (output). When reading data, returns the corresponding bit value in MCASP_PDOUT[n], does not return input from I/O pin; when writing data, writes to the corresponding MCASP_PDOUT[n] bit. MCASP_PDOUT has these aliases or alternate addresses: PDSET When written to at this address, writing a 1 to a bit in PDSET sets the corresponding bit in MCASP_PDOUT to 1; writing a 0 has no effect and keeps the bits in MCASP_PDOUT unchanged. MCASP_PDCLR When written to at this address, writing a 1 to a bit in MCASP_PDCLR clears the corresponding bit in MCASP_PDOUT to 0; writing a 0 has no effect and keeps the bits in MCASP_PDOUT unchanged. There is only one set of data out bits, MCASP_PDOUT[31-0]. The other registers, PDSET and MCASP_PDCLR, are just different addresses for the same control bits, with different behaviors during writes. CAUTION: Writing a value other than 0 to reserved bits in this register may cause improper device operation.

Table 11-1261. MCASP_PDOUT Instances

Instance	Physical Address
MCASP_0__CFG	0234 0018h
MCASP_1__CFG	0234 2018h
MCASP_2__CFG	0234 4018h

Figure 11-603. MCASP_PDOUT Register

31	30	29	28	27	26	25	24
AFSR	AHCLKR	ACLKR	AFSX	AHCLKX	ACLKX	AMUTE	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				AXR			
R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1262. MCASP_PDOUT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AFSR	R/W	0h	Determines drive on AFSR output pin when the corresponding MCASP_PFUNC[31] and MCASP_PDIR[31] bits are set to 1. 0h (R/W) = Pin drives low. 1h (R/W) = Pin drives high.
30	AHCLKR	R/W	0h	Determines drive on AHCLKR output pin when the corresponding MCASP_PFUNC[30] and MCASP_PDIR[30] bits are set to 1. 0h (R/W) = Pin drives low. 1h (R/W) = Pin drives high.
29	ACLKR	R/W	0h	Determines drive on ACLKR output pin when the corresponding MCASP_PFUNC[29] and MCASP_PDIR[29] bits are set to 1. 0h (R/W) = Pin drives low. 1h (R/W) = Pin drives high.

Table 11-1262. MCASP_PDOUT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
28	AFSX	R/W	0h	Determines drive on AFSX output pin when the corresponding MCASP_PFUNC[28] and MCASP_PDIR[28] bits are set to 1. 0h (R/W) = Pin drives low. 1h (R/W) = Pin drives high.
27	AHCLKX	R/W	0h	Determines drive on AHCLKX output pin when the corresponding MCASP_PFUNC[27] and MCASP_PDIR[27] bits are set to 1. 0h (R/W) = Pin drives low. 1h (R/W) = Pin drives high.
26	ACLKX	R/W	0h	Determines drive on ACLKX output pin when the corresponding MCASP_PFUNC[26] and MCASP_PDIR[26] bits are set to 1. 0h (R/W) = Pin drives low. 1h (R/W) = Pin drives high.
25	AMUTE	R/W	0h	Determines drive on AMUTE output pin when the corresponding MCASP_PFUNC[25] and MCASP_PDIR[25] bits are set to 1. 0h (R/W) = Pin drives low. 1h (R/W) = Pin drives high.
24-4	RESERVED	R	0h	Reserved
3-0	AXR	R/W	0h	Determines drive on AXR[n] output pin when the corresponding MCASP_PFUNC[n] and MCASP_PDIR[n] bits are set to 1. 0h (R/W) = Pin drives low. 1h (R/W) = Pin drives high.

Table 11-1263. Register Call Summary for MCASP_PDOUT

McASP Functional Description • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0]
McASP Registers • McASP Registers: [0] • MCASP_PDCLR Register (Offset = 20h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17] • MCASP_PDOUT Register (Offset = 18h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11] • MCASP_PDSET Register (Offset = 1Ch) [reset = 0h]: [0][1]

11.9.5.6 MCASP_PDIN Register (Offset = 1Ch) [reset = 0h]

MCASP_PDIN is shown in Figure 11-604 and described in Table 11-1265.

The pin data input register (MCASP_PDIN) holds the I/O pin state of each of the McASP pins. MCASP_PDIN allows the actual value of the pin to be read, regardless of the state of MCASP_PFUNC and MCASP_PDIR. The value after reset for registers 1 through 15 and 24 through 31 depends on how the pins are being driven. CAUTION: Writing a value other than 0 to reserved bits in this register may cause improper device operation.

Table 11-1264. MCASP_PDIN Instances

Instance	Physical Address
MCASP_0__CFG	0234 001Ch
MCASP_1__CFG	0234 201Ch
MCASP_2__CFG	0234 401Ch

Figure 11-604. MCASP_PDIN Register

31	30	29	28	27	26	25	24
AFSR	AHCLKR	ACLKR	AFSX	AHCLKX	ACLKX	AMUTE	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				AXR			
R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1265. MCASP_PDIN Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AFSR	R/W	0h	Logic level on AFSR pin. 0h (R/W) = Pin is logic low. 1h (R/W) = Pin is logic high.
30	AHCLKR	R/W	0h	Logic level on AHCLKR pin. 0h (R/W) = Pin is logic low. 1h (R/W) = Pin is logic high.
29	ACLKR	R/W	0h	Logic level on ACLKR pin. 0h (R/W) = Pin is logic low. 1h (R/W) = Pin is logic high.
28	AFSX	R/W	0h	Logic level on AFSX pin. 0h (R/W) = Pin is logic low. 1h (R/W) = Pin is logic high.
27	AHCLKX	R/W	0h	Logic level on AHCLKX pin. 0h (R/W) = Pin is logic low. 1h (R/W) = Pin is logic high.
26	ACLKX	R/W	0h	Logic level on ACLKX pin. 0h (R/W) = Pin is logic low. 1h (R/W) = Pin is logic high.

Table 11-1265. MCASP_PDIN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	AMUTE	R/W	0h	Logic level on AMUTE pin. 0h (R/W) = Pin is logic low. 1h (R/W) = Pin is logic high.
24-4	RESERVED	R	0h	Reserved
3-0	AXR	R/W	0h	Logic level on AXR[n] pin. 0h (R/W) = Pin is logic low. 1h (R/W) = Pin is logic high.

Table 11-1266. Register Call Summary for MCASP_PDIN

McASP Functional Description • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0]
McASP Registers • McASP Registers: [0] • MCASP_PDIN Register (Offset = 1Ch) [reset = 0h]: [0][1][2]

11.9.5.7 MCASP_PDSET Register (Offset = 1Ch) [reset = 0h]

MCASP_PDSET is shown in Figure 11-605 and described in Table 11-1265.

The pin data set register (MCASP_PDSET) is an alias of the pin data output register (MCASP_PDOUT) for writes only. Writing a 1 to the MCASP_PDSET bit sets the corresponding bit in MCASP_PDOUT and, if MCASP_PFUNC = 1 (GPIO function) and MCASP_PDIR = 1 (output), drives a logic high on the pin. CAUTION: Writing a value other than 0 to reserved bits in this register may cause improper device operation.

Table 11-1267. MCASP_PDSET Instances

Instance	Physical Address
MCASP_0__CFG	0234 001Ch
MCASP_1__CFG	0234 201Ch
MCASP_2__CFG	0234 401Ch

Figure 11-605. MCASP_PDSET Register

31	30	29	28	27	26	25	24
AFSR	AHCLKR	ACLKR	AFSX	AHCLKX	ACLKX	AMUTE	RESERVED
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				AXR			
R-0h				W-0h			

LEGEND: R = Read Only; R/W = Read/Write; W = Write Only; -*n* = value after reset

Table 11-1268. MCASP_PDSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AFSR	W	0h	Logic level on AFSR pin. 0h (W) = No effect 1h (W) = MACSP_PDOUT[31] bit is set to 1
30	AHCLKR	W	0h	Logic level on AHCLKR pin. 0h (W) = No effect 1h (W) = MACSP_PDOUT[30] bit is set to 1
29	ACLKR	W	0h	Logic level on ACLKR pin. 0h (W) = No effect 1h (W) = MACSP_PDOUT[29] bit is set to 1
28	AFSX	W	0h	Logic level on AFSX pin. 0h (W) = No effect 1h (W) = MACSP_PDOUT[28] bit is set to 1
27	AHCLKX	R/W	0h	Logic level on AHCLKX pin. 0h (W) = No effect 1h (W) = MACSP_PDOUT[27] bit is set to 1
26	ACLKX	W	0h	Logic level on ACLKX pin. 0h (W) = No effect 1h (W) = MACSP_PDOUT[26] bit is set to 1

Table 11-1268. MCASP_PDSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	AMUTE	W	0h	Logic level on AMUTE pin. 0h (W) = No effect 1h (W) = MACSP_PDOUT[25] bit is set to 1
24-4	RESERVED	R	0h	Reserved
3-0	AXR	W	0h	Logic level on AXR[n] pin. 0h (W) = No effect 1h (W) = MACSP_PDOUT[3-0] bits is set to 1

Table 11-1269. Register Call Summary for MCASP_PDSET

McASP Functional Description • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0]
McASP Registers • McASP Registers: [0] • MCASP_PDSET Register (Offset = 1Ch) [reset = 0h]: [0][1][2]

11.9.5.8 MCASP_PDCLR Register (Offset = 20h) [reset = 0h]

MCASP_PDCLR is shown in Figure 11-606 and described in Table 11-1271.

The pin data clear register (MCASP_PDCLR) is an alias of the pin data output register (MCASP_PDOUT) for writes only. Writing a 1 to the MCASP_PDCLR bit clears the corresponding bit in MCASP_PDOUT and, if MCASP_PFUNC = 1 (GPIO function) and MCASP_PDIR = 1 (output), drives a logic low on the pin. MCASP_PDCLR is useful for a multitasking system because it allows clearing to a logic low only the desired pin(s) within a system without affecting other I/O pins controlled by the same McASP. CAUTION: Writing a value other than 0 to reserved bits in this register may cause improper device operation.

Table 11-1270. MCASP_PDCLR Instances

Instance	Physical Address
MCASP_0__CFG	0234 0020h
MCASP_1__CFG	0234 2020h
MCASP_2__CFG	0234 4020h

Figure 11-606. MCASP_PDCLR Register

31	30	29	28	27	26	25	24
AFSR	AHCLKR	ACLKR	AFSX	AHCLKX	ACLKX	AMUTE	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				AXR			
R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1271. MCASP_PDCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AFSR	R/W	0h	Allows the corresponding AFSR bit in MCASP_PDOUT to be cleared to a logic low without affecting other I/O pins controlled by the same port. 0h (R/W) = No effect. 1h (R/W) = MCASP_PDOUT[31] bit is cleared to 0.
30	AHCLKR	R/W	0h	Allows the corresponding AHCLKR bit in MCASP_PDOUT to be cleared to a logic low without affecting other I/O pins controlled by the same port. 0h (R/W) = No effect. 1h (R/W) = MCASP_PDOUT[30] bit is cleared to 0.
29	ACLKR	R/W	0h	Allows the corresponding ACLKR bit in MCASP_PDOUT to be cleared to a logic low without affecting other I/O pins controlled by the same port. 0h (R/W) = No effect. 1h (R/W) = MCASP_PDOUT[29] bit is cleared to 0.
28	AFSX	R/W	0h	Allows the corresponding AFSX bit in MCASP_PDOUT to be cleared to a logic low without affecting other I/O pins controlled by the same port. 0h (R/W) = No effect. 1h (R/W) = MCASP_PDOUT[28] bit is cleared to 0.

Table 11-1271. MCASP_PDCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	AHCLKX	R/W	0h	Allows the corresponding AHCLKX bit in MCASP_PDOUT to be cleared to a logic low without affecting other I/O pins controlled by the same port. 0h (R/W) = No effect. 1h (R/W) = MCASP_PDOUT[27] bit is cleared to 0.
26	ACLKX	R/W	0h	Allows the corresponding ACLKX bit in MCASP_PDOUT to be cleared to a logic low without affecting other I/O pins controlled by the same port. 0h (R/W) = No effect. 1h (R/W) = MCASP_PDOUT[26] bit is cleared to 0.
25	AMUTE	R/W	0h	Allows the corresponding AMUTE bit in MCASP_PDOUT to be cleared to a logic low without affecting other I/O pins controlled by the same port. 0h (R/W) = No effect. 1h (R/W) = MCASP_PDOUT[25] bit is cleared to 0.
24-4	RESERVED	R	0h	Reserved
3-0	AXR	R/W	0h	Allows the corresponding AXR[n] bit in MCASP_PDOUT to be cleared to a logic low without affecting other I/O pins controlled by the same port. 0h (R/W) = No effect. 1h (R/W) = MCASP_PDOUT[n] bit is cleared to 0.

Table 11-1272. Register Call Summary for MCASP_PDCLR

McASP Functional Description • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0]
McASP Registers • McASP Registers: [0] • MCASP_PDCLR Register (Offset = 20h) [reset = 0h]: [0][1][2][3] • MCASP_PDOUT Register (Offset = 18h) [reset = 0h]: [0][1][2]

11.9.5.9 MCASP_GBLCTL Register (Offset = 44h) [reset = 0h]

MCASP_GBLCTL is shown in Figure 11-607 and described in Table 11-1274.

The global control register (MCASP_GBLCTL) provides initialization of the transmit and receive sections. The bit fields in MCASP_GBLCTL are synchronized and latched by the corresponding clocks (ACLKX for bits 12-8 and ACLKR for bits 4-0). Before MCASP_GBLCTL is programmed, SW must ensure that serial clocks are running. If the corresponding external serial clocks, ACLKX and ACLKR, are not yet running, SW should select the internal serial clock source in MCASP_AHCLKXCTL, MCASP_AHCLKRCTL, MCASP_ACLKXCTL, and MCASP_ACLKRCTL before MCASP_GBLCTL is programmed. Also, after programming any bits in MCASP_GBLCTL SW should not proceed until a read back from MCASP_GBLCTL is performed and it is verified that the bits are latched in MCASP_GBLCTL.

Table 11-1273. MCASP_GBLCTL Instances

Instance	Physical Address
MCASP_0__CFG	0234 0044h
MCASP_1__CFG	0234 2044h
MCASP_2__CFG	0234 4044h

Figure 11-607. MCASP_GBLCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			XFRST	XSMRST	XSRCLR	XHCLKRST	XCLKRST
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED			RFRST	RSMRST	RSRCLR	RHCLKRST	RCLKRST
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1274. MCASP_GBLCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12	XFRST	R/W	0h	Transmit frame sync generator reset enable bit. 0h (R/W) = Transmit frame sync generator is reset. 1h (R/W) = Transmit frame sync generator is active. When released from reset, the transmit frame sync generator begins counting serial clocks and generating frame sync as programmed.
11	XSMRST	R/W	0h	Transmit state machine reset enable bit. 0h (R/W) = Transmit state machine is held in reset. AXRn pin state: If MCASP_PFUNC[n] = 0 and MCASP_PDIR[n] = 1; then the serializer drives the AXRn pin to the state specified for inactive time slot (as determined by DISMOD bits in SRCTL). 1h (R/W) = Transmit state machine is released from reset. When released from reset, the transmit state machine immediately transfers data from XRBUF[n] to XRSR[n]. The transmit state machine sets the underrun flag (XUNDRN) in MCASP_XSTAT, if XRBUF[n] have not been preloaded with data before reset is released. The transmit state machine also immediately begins detecting frame sync and is ready to transmit. Transmit TDM time slot begins at slot 0 after reset is released.

Table 11-1274. MCASP_GBLCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	XSRCLR	R/W	0h	Transmit serializer clear enable bit. By clearing then setting this bit, the transmit buffer is flushed to an empty state (XDATA = 1). If XSMRST = 1, XSRCLR = 1, XDATA = 1, and MCASP_XBUF is not loaded with new data before the start of the next active time slot, an underrun will occur. 0h (R/W) = Transmit serializers are cleared. 1h (R/W) = Transmit serializers are active. When the transmit serializers are first taken out of reset (XSRCLR changes from 0 to 1), the transmit data ready bit (XDATA) in MCASP_XSTAT is set to indicate MCASP_XBUF is ready to be written.
9	XHCLKRST	R/W	0h	Transmit high-frequency clock divider reset enable bit. 0h (R/W) = Transmit high-frequency clock divider is held in reset and passes through its input as divide-by-1. 1h (R/W) = Transmit high-frequency clock divider is running.
8	XCLKRST	R/W	0h	Transmit clock divider reset enable bit. 0h (R/W) = Transmit clock divider is held in reset. When the clock divider is in reset, it passes through a divide-by-1 of its input. 1h (R/W) = Transmit clock divider is running.
7-5	RESERVED	R	0h	Reserved
4	RFRST	R/W	0h	Receive frame sync generator reset enable bit. 0h (R/W) = Receive frame sync generator is reset. 1h (R/W) = Receive frame sync generator is active. When released from reset, the receive frame sync generator begins counting serial clocks and generating frame sync as programmed.
3	RSMRST	R/W	0h	Receive state machine reset enable bit. 0h (R/W) = Receive state machine is held in reset. 1h (R/W) = Receive state machine is released from reset. When released from reset, the receive state machine immediately begins detecting frame sync and is ready to receive. Receive TDM time slot begins at slot 0 after reset is released.
2	RSRCLR	R/W	0h	Receive serializer clear enable bit. By clearing then setting this bit, the receive buffer is flushed. 0h (R/W) = Receive serializers are cleared. 1h (R/W) = Receive serializers are active.
1	RHCLKRST	R/W	0h	Receive high-frequency clock divider reset enable bit. 0h (R/W) = Receive high-frequency clock divider is held in reset and passes through its input as divide-by-1. 1h (R/W) = Receive high-frequency clock divider is running.
0	RCLKRST	R/W	0h	Receive high-frequency clock divider reset enable bit. 0h (R/W) = Receive clock divider is held in reset. When the clock divider is in reset, it passes through a divide-by-1 of its input. 1h (R/W) = Receive clock divider is running.

Table 11-1275. Register Call Summary for MCASP_GBLCTL

McASP Functional Description
- McASP Software Reset: [0][1]
- Time-Division Multiplexed (TDM) Transfer Mode: [0]
- Transmit DIT Clock and Frame-Sync Generation: [0]

Table 11-1275. Register Call Summary for MCASP_GBLCTL (continued)

McASP Registers
• MCASP_SRCTL13 Register (Offset = 1B4h) [reset = 0h]: [0] • MCASP_RGBLCTL Register (Offset = 60h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12] • MCASP_SRCTL12 Register (Offset = 1B0h) [reset = 0h]: [0] • McASP Registers: [0] • MCASP_SRCTL10 Register (Offset = 1A8h) [reset = 0h]: [0] • MCASP_SRCTL11 Register (Offset = 1ACh) [reset = 0h]: [0] • MCASP_DITCTL Register (Offset = 50h) [reset = 0h]: [0][1] • MCASP_SRCTL14 Register (Offset = 1B8h) [reset = 0h]: [0] • MCASP_XGBLCTL Register (Offset = A0h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14] • MCASP_SRCTL0 Register (Offset = 180h) [reset = 0h]: [0] • MCASP_SRCTL1 Register (Offset = 184h) [reset = 0h]: [0] • MCASP_SRCTL2 Register (Offset = 188h) [reset = 0h]: [0] • MCASP_SRCTL3 Register (Offset = 18Ch) [reset = 0h]: [0] • MCASP_SRCTL4 Register (Offset = 190h) [reset = 0h]: [0] • MCASP_SRCTL5 Register (Offset = 194h) [reset = 0h]: [0] • MCASP_SRCTL6 Register (Offset = 198h) [reset = 0h]: [0] • MCASP_SRCTL7 Register (Offset = 19Ch) [reset = 0h]: [0] • MCASP_SRCTL8 Register (Offset = 1A0h) [reset = 0h]: [0] • MCASP_SRCTL9 Register (Offset = 1A4h) [reset = 0h]: [0] • MCASP_SRCTL15 Register (Offset = 1BCh) [reset = 0h]: [0] • MCASP_GBLCTL Register (Offset = 44h) [reset = 0h]: [0][1][2][3][4][5][6][7]

11.9.5.10 MCASP_AMUTE Register (Offset = 48h) [reset = 0h]

MCASP_AMUTE is shown in Figure 11-608 and described in Table 11-1277.

The audio mute control register (MCASP_AMUTE) controls the McASP audio mute (AMUTE) output pin. The value after reset for register 4 depends on how the pins are being driven.

Table 11-1276. MCASP_AMUTE Instances

Instance	Physical Address
MCASP_0__CFG	0234 0048h
MCASP_1__CFG	0234 2048h
MCASP_2__CFG	0234 4048h

Figure 11-608. MCASP_AMUTE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			XDMAERR	RDMAERR	XCKFAIL	RCKFAIL	XSYNCERR
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RSYNCERR	XUNDRN	ROVRN	INSTAT	INEN	INPOL	MUTEN	
R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1277. MCASP_AMUTE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12	XDMAERR	R/W	0h	If transmit DMA error (XDMAERR), drive MCASP_AMUTE active enable bit. 0h (R/W) = Drive is disabled. Detection of transmit DMA error is ignored by MCASP_AMUTE. 1h (R/W) = Drive is enabled (active). Upon detection of transmit DMA error, MCASP_AMUTE is active and is driven according to MUTEN bit.
11	RDMAERR	R/W	0h	If receive DMA error (RDMAERR), drive MCASP_AMUTE active enable bit. 0h (R/W) = Drive is disabled. Detection of receive DMA error is ignored by MCASP_AMUTE. 1h (R/W) = Drive is enabled (active). Upon detection of receive DMA error, MCASP_AMUTE is active and is driven according to MUTEN bit.
10	XCKFAIL	R/W	0h	If transmit clock failure (XCKFAIL), drive MCASP_AMUTE active enable bit. 0h (R/W) = Drive is disabled. Detection of transmit clock failure is ignored by MCASP_AMUTE. 1h (R/W) = Drive is enabled (active). Upon detection of transmit clock failure, MCASP_AMUTE is active and is driven according to MUTEN bit

Table 11-1277. MCASP_AMUTE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	RCKFAIL	R/W	0h	If receive clock failure (RCKFAIL), drive MCASP_AMUTE active enable bit. 0h (R/W) = Drive is disabled. Detection of receive clock failure is ignored by MCASP_AMUTE. 1h (R/W) = Drive is enabled (active). Upon detection of receive clock failure, MCASP_AMUTE is active and is driven according to MUTEN bit.
8	XSYNCERR	R/W	0h	If unexpected transmit frame sync error (XSYNCERR), drive MCASP_AMUTE active enable bit. 0h (R/W) = Drive is disabled. Detection of unexpected transmit frame sync error is ignored by MCASP_AMUTE. 1h (R/W) = Drive is enabled (active). Upon detection of unexpected transmit frame sync error, MCASP_AMUTE is active and is driven according to MUTEN bit.
7	RSYNCERR	R/W	0h	If unexpected receive frame sync error (RSYNCERR), drive MCASP_AMUTE active enable bit. 0h (R/W) = Drive is disabled. Detection of unexpected receive frame sync error is ignored by MCASP_AMUTE. 1h (R/W) = Drive is enabled (active). Upon detection of unexpected receive frame sync error, MCASP_AMUTE is active and is driven according to MUTEN bit.
6	XUNDRN	R/W	0h	If transmit underrun error (XUNDRN), drive MCASP_AMUTE active enable bit. 0h (R/W) = Drive is disabled. Detection of transmit underrun error is ignored by MCASP_AMUTE. 1h (R/W) = Drive is enabled (active). Upon detection of transmit underrun error, MCASP_AMUTE is active and is driven according to MUTEN bit.
5	ROVRN	R/W	0h	If receiver overrun error (ROVRN), drive MCASP_AMUTE active enable bit. 0h (R/W) = Drive is disabled. Detection of receiver overrun error is ignored by MCASP_AMUTE. 1h (R/W) = Drive is enabled (active). Upon detection of receiver overrun error, MCASP_AMUTE is active and is driven according to MUTEN bit.
4	INSTAT	R	0h	Determines drive on AXRn pin when MCASP_PFUNC[n] and MCASP_PDIR[n] bits are set to 1. 0h (R/W) = AMUTEIN pin is inactive. 1h (R/W) = AMUTEIN pin is active. Audio mute in error is detected.
3	INEN	R/W	0h	Drive MCASP_AMUTE active when AMUTEIN error is active (INSTAT = 1). 0h (R/W) = Drive is disabled. AMUTEIN is ignored by MCASP_AMUTE. 1h (R/W) = Drive is enabled (active). INSTAT = 1 drives MCASP_AMUTE active.
2	INPOL	R/W	0h	Audio mute in (AMUTEIN) polarity select bit. 0h (R/W) = Polarity is active high. A high on AMUTEIN sets INSTAT to 1. 1h (R/W) = Polarity is active low. A low on AM UTEIN sets INSTAT to 1.
1-0	MUTEN	R/W	0h	MCASP_AMUTE pin enable bit (unless overridden by GPIO registers). 0h (R/W) = MCASP_AMUTE pin is disabled, pin goes to tri-state condition. 1h (R/W) = MCASP_AMUTE pin is driven high if error is detected. 2h (R/W) = MCASP_AMUTE pin is driven low if error is detected. 3h (R/W) = Reserved

Table 11-1278. Register Call Summary for MCASP_AMUTE

McASP Functional Description • Audio Mute (MCASP_AMUTE) Function: [0][1][2] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit DIT Clock and Frame-Sync Generation: [0]
McASP Registers • McASP Registers: [0] • MCASP_AMUTE Register (Offset = 48h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25][26][27][28][29][30][31][32]

11.9.5.11 MCASP_DLBCTL Register (Offset = 4Ch) [reset = 0h]

MCASP_DLBCTL is shown in Figure 11-609 and described in Table 11-1280.

The digital loopback control register (MCASP_DLBCTL) controls the internal loopback settings of the McASP in TDM mode.

Table 11-1279. MCASP_DLBCTL Instances

Instance	Physical Address
MCASP_0__CFG	0234 004Ch
MCASP_1__CFG	0234 204Ch
MCASP_2__CFG	0234 404Ch

Figure 11-609. MCASP_DLBCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MODE		ORD	DLBEN
R-0h				R/W-0h		R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1280. MCASP_DLBCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-2	MODE	R/W	0h	Loopback generator mode bits. Applies only when loopback mode is enabled (DLBEN = 1). 0h (R/W) = Default and reserved on loopback mode (DLBEN = 1). When in non-loopback mode (DLBEN = 0), MODE should be left at default (00). When in loopback mode (DLBEN = 1), MODE = 00 is reserved and is not applicable. 1h (R/W) = Transmit clock and frame sync generators used by both transmit and receive sections. When in loopback mode (DLBEN = 1), MODE must be 01. 2h (R/W) = Reserved. 3h (R/W) = Reserved.
1	ORD	R/W	0h	Loopback order bit when loopback mode is enabled (DLBEN = 1). 0h (R/W) = Odd serializers N + 1 transmit to even serializers N that receive. The corresponding serializers must be programmed properly. 1h (R/W) = Even serializers N transmit to odd serializers N + 1 that receive. The corresponding serializers must be programmed properly.
0	DLBEN	R/W	0h	Loopback mode enable bit. 0h (R/W) = Loopback mode is disabled. 1h (R/W) = Loopback mode is enabled.

Table 11-1281. Register Call Summary for MCASP_DLBCTL

McASP Functional Description • McASP Loopback Modes: [0][1][2][3][4] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Loopback Mode Configurations: [0][1][2][3]
McASP Registers • McASP Registers: [0] • MCASP_DLBCTL Register (Offset = 4Ch) [reset = 0h]: [0][1]

11.9.5.12 MCASP_DITCTL Register (Offset = 50h) [reset = 0h]

MCASP_DITCTL is shown in Figure 11-610 and described in Table 11-1283.

The DIT mode control register (MCASP_DITCTL) controls DIT operations of the McASP.

Table 11-1282. MCASP_DITCTL Instances

Instance	Physical Address
MCASP_0__CFG	0234 0050h
MCASP_1__CFG	0234 2050h
MCASP_2__CFG	0234 4050h

Figure 11-610. MCASP_DITCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				VB	VA	RESERVED	DITEN
R-0h				R/W-0h	R/W-0h	R-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1283. MCASP_DITCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	VB	R/W	0h	Valid bit for odd time slots (DIT right subframe). 0h (R/W) = V bit is 0 during odd DIT subframes. 1h (R/W) = V bit is 1 during odd DIT subframes.
2	VA	R/W	0h	Valid bit for even time slots (DIT left subframe). 0h (R/W) = V bit is 0 during even DIT subframes. 1h (R/W) = V bit is 1 during even DIT subframes.
1	RESERVED	R	0h	Reserved
0	DITEN	R/W	0h	DIT mode enable bit. DITEN should only be changed while the XSMRST bit in MCASP_GBLCTL is in reset (and for startup, XSRCLR also in reset). However, it is not necessary to reset the XCLKRST or XHCLKRST bits in MCASP_GBLCTL to change DITEN. 0h (R/W) = DIT mode is disabled. Transmitter operates in TDM or burst mode. 1h (R/W) = DIT mode is enabled. Transmitter operates in DIT encoded mode.

Table 11-1284. Register Call Summary for MCASP_DITCTL

McASP Functional Description
- McASP Synchronous and Asynchronous Transmit and Receive Operations: [0][1]
- Burst Transfer Mode: [0]
- Time-Division Multiplexed (TDM) Transfer Mode: [0]
- Transmit DIT Clock and Frame-Sync Generation: [0]

Table 11-1284. Register Call Summary for MCASP_DITCTL (continued)

McASP Registers
• McASP Registers: [0] • MCASP_DITCTL Register (Offset = 50h) [reset = 0h]: [0][1]

11.9.5.13 MCASP_RGBLCTL Register (Offset = 60h) [reset = 0h]

MCASP_RGBLCTL is shown in Figure 11-611 and described in Table 11-1286.

Alias of the global control register (MCASP_GBLCTL). Writing to the receiver global control register (MCASP_RGBLCTL) affects only the receive bits of MCASP_GBLCTL (bits 4-0). Reads from MCASP_RGBLCTL return the value of MCASP_GBLCTL. MCASP_RGBLCTL allows the receiver to be reset independently from the transmitter.

Table 11-1285. MCASP_RGBLCTL Instances

Instance	Physical Address
MCASP_0__CFG	0234 0060h
MCASP_1__CFG	0234 2060h
MCASP_2__CFG	0234 4060h

Figure 11-611. MCASP_RGBLCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			XFRST	XSMRST	XSRCLR	XHCLKRST	XCLKRST
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED			RFRST	RSMRST	RSRCLR	RHCLKRST	RCLKRST
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1286. MCASP_RGBLCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12	XFRST	R	0h	Transmit frame sync generator reset enable bit. A read of this bit returns the XFRST bit value of MCASP_GBLCTL. Writes have no effect.
11	XSMRST	R	0h	Transmit state machine reset enable bit. A read of this bit returns the XSMRST bit value of MCASP_GBLCTL. Writes have no effect.
10	XSRCLR	R	0h	Transmit serializer clear enable bit. A read of this bit returns the XSRCLR bit value of MCASP_GBLCTL. Writes have no effect.
9	XHCLKRST	R	0h	Transmit high-frequency clock divider reset enable bit. A read of this bit returns the XHCLKRST bit value of MCASP_GBLCTL. Writes have no effect.
8	XCLKRST	R	0h	Transmit clock divider reset enable bit. A read of this bit returns the XCLKRST bit value of MCASP_GBLCTL. Writes have no effect.
7-5	RESERVED	R	0h	Reserved
4	RFRST	R/W	0h	Receive frame sync generator reset enable bit. A write to this bit affects the RFRST bit of MCASP_GBLCTL. 0h (R/W) = Receive frame sync generator is reset. 1h (R/W) = Receive frame sync generator is active.

Table 11-1286. MCASP_RGBLCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	RSMRST	R/W	0h	Receive state machine reset enable bit. A write to this bit affects the RSMRST bit of MCASP_GBLCTL. 0h (R/W) = Receive state machine is held in reset. 1h (R/W) = Receive state machine is released from reset.
2	RSRCLR	R/W	0h	Receive serializer clear enable bit. A write to this bit affects the RSRCLR bit of MCASP_GBLCTL. 0h (R/W) = Receive serializers are cleared. 1h (R/W) = Receive serializers are active.
1	RHCLKRST	R/W	0h	Receive high-frequency clock divider reset enable bit. A write to this bit affects the RHCLKRST bit of MCASP_GBLCTL. 0h (R/W) = Receive high-frequency clock divider is held in reset and passes through its input as divide-by-1. 1h (R/W) = Receive high-frequency clock divider is running.
0	RCLKRST	R/W	0h	Receive clock divider reset enable bit. A write to this bit affects the RCLKRST bit of MCASP_GBLCTL. 0h (R/W) = Receive clock divider is held in reset. 1h (R/W) = Receive clock divider is running.

Table 11-1287. Register Call Summary for MCASP_RGBLCTL

McASP Registers
- McASP Registers: [0]
- MCASP_RGBLCTL Register (Offset = 60h) [reset = 0h]: [0][1][2][3]

11.9.5.14 MCASP_RMASK Register (Offset = 64h) [reset = 0h]

MCASP_RMASK is shown in Figure 11-612 and described in Table 11-1289.

The receive format unit bit mask register (MCASP_RMASK) determines which bits of the received data are masked off and padded with a known value before being read by the CPU or DMA.

Table 11-1288. MCASP_RMASK Instances

Instance	Physical Address
MCASP_0__CFG	0234 0064h
MCASP_1__CFG	0234 2064h
MCASP_2__CFG	0234 4064h

Figure 11-612. MCASP_RMASK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMASK																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1289. MCASP_RMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RMASK	R/W	0h	Receive data mask n enable bit. 0h (R/W) = Corresponding bit of receive data (after passing through reverse and rotate units) is masked out and then padded with the selected bit pad value (RPAD and RPBIT bits in MCASP_RFMT). 1h (R/W) = Corresponding bit of receive data (after passing through reverse and rotate units) is returned to CPU or DMA.

Table 11-1290. Register Call Summary for MCASP_RMASK

McASP Functional Description • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0]
McASP Registers • McASP Registers: [0] • MCASP_RFMT Register (Offset = 68h) [reset = 0h]: [0] • MCASP_RMASK Register (Offset = 64h) [reset = 0h]: [0][1]

11.9.5.15 MCASP_RFMT Register (Offset = 68h) [reset = 0h]

MCASP_RFMT is shown in Figure 11-613 and described in Table 11-1292.

The receive bit stream format register (MCASP_RFMT) configures the receive data format.

Table 11-1291. MCASP_RFMT Instances

Instance	Physical Address
MCASP_0__CFG	0234 0068h
MCASP_1__CFG	0234 2068h
MCASP_2__CFG	0234 4068h

Figure 11-613. MCASP_RFMT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						RDATDLY	
R-0h						R/W-0h	
15	14	13	12	11	10	9	8
RRVRS	RPAD			RPBIT			
R/W-0h	R/W-0h			R/W-0h			
7	6	5	4	3	2	1	0
RSSZ				RBUSEL	RROT		
R/W-0h				R/W-0h	R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1292. MCASP_RFMT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17-16	RDATDLY	R/W	0h	Receive bit delay. 0h (R/W) = 0-bit delay. The first receive data bit, AXRn, occurs in same ACLKR cycle as the receive frame sync (AFSR). 1h (R/W) = 1-bit delay. The first receive data bit, AXRn, occurs one ACLKR cycle after the receive frame sync (AFSR). 2h (R/W) = 2-bit delay. The first receive data bit, AXRn, occurs two ACLKR cycles after the receive frame sync (AFSR). 3h (R/W) = Reserved.
15	RRVRS	R/W	0h	Receive serial bitstream order. 0h (R/W) = Bitstream is LSB first. No bit reversal is performed in receive format bit reverse unit. 1h (R/W) = Bitstream is MSB first. Bit reversal is performed in receive format bit reverse unit.
14-13	RPAD	R/W	0h	Pad value for extra bits in slot not belonging to the word. This field only applies to bits when MCASP_RMASK[n] = 0. 0h (R/W) = Pad extra bits with 0. 1h (R/W) = Pad extra bits with 1. 2h (R/W) = Pad extra bits with one of the bits from the word as specified by RPBIT bits. 3h (R/W) = Reserved.
12-8	RPBIT	R/W	0h	RPBIT value determines which bit (as read by the CPU or DMA from MCASP_RBUF[n]) is used to pad the extra bits. This field only applies when RPAD = 2h. 0h (R/W) = Pad with bit 0 value. 1h (R/W) = Pad with bit 1 to bit 31 value from 1h to 1Fh.

Table 11-1292. MCASP_RFMT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-4	RSSZ	R/W	0h	Receive slot size. 0h (R/W) = Reseved. 1h (R/W) = Reserved. 2h (R/W) = Reserved. 3h (R/W) = Slot size is 8 bits. 4h (R/W) = Reserved 5h (R/W) = Slot size is 12 bits. 6h (R/W) = Reserved 7h (R/W) = Slot size is 16 bits. 8h (R/W) = Reserved 9h (R/W) = Slot size is 20 bits. Ah (R/W) = Reserved Bh (R/W) = Slot size is 24 bits Ch (R/W) = Reserved Dh (R/W) = Slot size is 28 bits. Eh (R/W) = Reserved Fh (R/W) = Slot size is 32 bits.
3	RBUSEL	R/W	0h	Selects whether reads from serializer buffer XRBUF[n] originate from the configuration bus (CFG) or the data (DAT) port. 0h (R/W) = Reads from XRBUF[n] originate on data port. Reads from XRBUF[n] on configuration bus are ignored. 1h (R/W) = Reads from XRBUF[n] originate on configuration bus. Reads from XRBUF[n] on data port are ignored.
2-0	RROT	R/W	0h	Right-rotation value for receive rotate right format unit. 0h (R/W) = Rotate right by 0 (no rotation). 1h (R/W) = Rotate right by 4 bit positions. 2h (R/W) = Rotate right by 8 bit positions. 3h (R/W) = Rotate right by 12 bit positions. 4h (R/W) = Rotate right by 16 bit positions. 5h (R/W) = Rotate right by 20 bit positions. 6h (R/W) = Rotate right by 24 bit positions. 7h (R/W) = Rotate right by 28 bit positions.

Table 11-1293. Register Call Summary for MCASP_RFMT

McASP Functional Description • Transfers Through the Data Port (DATA): [0] • Frame-Sync Generator: [0] • TDM Mode Reception Data Alignment Settings: [0] • McASP Format Units: [0][1][2] • Receive Format Unit: [0][1] • Transfers Through the Configuration Bus (CFG): [0] • McASP State-Machines: [0] • Burst Transfer Mode: [0][1] • Time-Division Multiplexed (TDM) Transfer Mode: [0]
McASP Registers • McASP Registers: [0][1] • MCASP_RFMT Register (Offset = 68h) [reset = 0h]: [0][1] • MCASP_RBUF Register (Offset = 0h) [reset = 0h]: [0] • MCASP_RMASK Register (Offset = 64h) [reset = 0h]: [0]

11.9.5.16 MCASP_AFSRCTL Register (Offset = 6Ch) [reset = 0h]

MCASP_AFSRCTL is shown in Figure 11-614 and described in Table 11-1295.

The receive frame sync control register (MCASP_AFSRCTL) configures the receive frame sync (AFSR).

Table 11-1294. MCASP_AFSRCTL Instances

Instance	Physical Address
MCASP_0__CFG	0234 006Ch
MCASP_1__CFG	0234 206Ch
MCASP_2__CFG	0234 406Ch

Figure 11-614. MCASP_AFSRCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RMOD							
R/W-0h							
7	6	5	4	3	2	1	0
RMOD	RESERVED		FRWID	RESERVED		FSRM	FSRP
R/W-0h	R-0h		R/W-0h	R-0h		R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1295. MCASP_AFSRCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-7	RMOD	R/W	0h	Receive frame sync mode select bits. 1FFh = Reserved from 181h to 1FFh. 0h (R/W) = Burst mode. 1h (R/W) = Reserved. 2h (R/W) = 2-slot TDM (I2S mode) to 32-slot TDM from 2h to 20h. 21h (R/W) = Reserved from 21h to 17Fh. 180h (R/W) = 384-slot TDM (external DIR IC inputting 384-slot DIR frames to McASP over I2S interface). 181h (R/W) = Reserved from 181h to 1FFh.
6-5	RESERVED	R	0h	Reserved
4	FRWID	R/W	0h	Receive frame sync width select bit indicates the width of the receive frame sync (AFSR) during its active period. 0h (R/W) = Single bit. 1h (R/W) = Single word.
3-2	RESERVED	R	0h	Reserved
1	FSRM	R/W	0h	Receive frame sync generation select bit. 0h (R/W) = Externally-generated receive frame sync. 1h (R/W) = Internally-generated receive frame sync.
0	FSRP	R/W	0h	Receive frame sync polarity select bit. 0h (R/W) = A rising edge on receive frame sync (AFSR) indicates the beginning of a frame. 1h (R/W) = A falling edge on receive frame sync (AFSR) indicates the beginning of a frame.

Table 11-1296. Register Call Summary for MCASP_AFSRCTL

McASP Functional Description • Frame-Sync Generator: [0][1][2][3][4][5][6] • Burst Transfer Mode: [0][1] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Loopback Mode Configurations: [0] • Special 384-Slot TDM Mode for Connection to External DIR: [0]
McASP Registers • McASP Registers: [0] • MCASP_AFSRCTL Register (Offset = 6Ch) [reset = 0h]: [0][1]

11.9.5.17 MCASP_ACLKRCTL Register (Offset = 70h) [reset = 0h]

MCASP_ACLKRCTL is shown in Figure 11-615 and described in Table 11-1298.

The receive clock control register (MCASP_ACLKRCTL) configures the receive bit clock (ACLKR) and the receive clock generator.

Table 11-1297. MCASP_ACLKRCTL Instances

Instance	Physical Address
MCASP_0__CFG	0234 0070h
MCASP_1__CFG	0234 2070h
MCASP_2__CFG	0234 4070h

Figure 11-615. MCASP_ACLKRCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CLKRP	RESERVED	CLKRM	CLKRDIV				
R/W-0h	R-0h	R/W-0h	R/W-0h				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1298. MCASP_ACLKRCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	CLKRP	R/W	0h	Receive bitstream clock polarity select bit. 0h (R/W) = Falling edge. Receiver samples data on the falling edge of the serial clock, so the external transmitter driving this receiver must shift data out on the rising edge of the serial clock. 1h (R/W) = Rising edge. Receiver samples data on the rising edge of the serial clock, so the external transmitter driving this receiver must shift data out on the falling edge of the serial clock.
6	RESERVED	R	0h	Reserved
5	CLKRM	R/W	0h	Receive bit clock source bit. Note that this bit does not have any effect, if MCASP_ACLKXCTL.ASYNC = 0. 0h (R/W) = External receive clock source from ACLKR pin. 1h (R/W) = Internal receive clock source from output of programmable bit clock divider.
4-0	CLKRDIV	R/W	0h	Receive bit clock divide ratio bits determine the divide-down ratio from AHCLKR to ACLKR. Note that this bit does not have any effect, if MCASP_ACLKXCTL.ASYNC = 0. 0h (R/W) = Divide-by-1. 1h (R/W) = Divide-by-2. 2h (R/W) = Divide-by-3 to divide-by-32 from 2h to 1Fh.

Table 11-1299. Register Call Summary for MCASP_ACLKRCTL

McASP Functional Description • Receive Clock: [0][1][2][3] • McASP Synchronous and Asynchronous Transmit and Receive Operations: [0] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0]
McASP Registers • McASP Registers: [0] • MCASP_ACLKRCTL Register (Offset = 70h) [reset = 0h]: [0][1] • MCASP_GBLCTL Register (Offset = 44h) [reset = 0h]: [0] • MCASP_AHCLKRCTL Register (Offset = 74h) [reset = 0h]: [0][1]

11.9.5.18 MCASP_AHCLKRCTL Register (Offset = 74h) [reset = 0h]

MCASP_AHCLKRCTL is shown in Figure 11-616 and described in Table 11-1301.

The receive high-frequency clock control register (MCASP_AHCLKRCTL) configures the receive high-frequency master clock (AHCLKR) and the receive clock generator.

Table 11-1300. MCASP_AHCLKRCTL Instances

Instance	Physical Address
MCASP_0__CFG	0234 0074h
MCASP_1__CFG	0234 2074h
MCASP_2__CFG	0234 4074h

Figure 11-616. MCASP_AHCLKRCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
HCLKRM	HCLKRP	RESERVED		HCLKRDIV			
R/W-0h	R/W-0h	R-0h		R/W-0h			
7	6	5	4	3	2	1	0
HCLKRDIV							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1301. MCASP_AHCLKRCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	HCLKRM	R/W	0h	Receive high-frequency clock source bit. 0h (R/W) = External receive high-frequency clock source from AHCLKR pin. 1h (R/W) = Internal receive high-frequency clock source from output of programmable high clock divider.
14	HCLKRP	R/W	0h	Receive bitstream high-frequency clock polarity select bit. 0h (R/W) = AHCLKR is not inverted before programmable bit clock divider. In the special case where the receive bit clock (ACLKR) is internally generated and the programmable bit clock divider is set to divide-by-1 (CLKRDIV = 0 in MCASP_ACLKRCTL), AHCLKR is directly passed through to the ACLKR pin. 1h (R/W) = AHCLKR is inverted before programmable bit clock divider. In the special case where the receive bit clock (ACLKR) is internally generated and the programmable bit clock divider is set to divide-by-1 (CLKRDIV = 0 in MCASP_ACLKRCTL), AHCLKR is directly passed through to the ACLKR pin.
13-12	RESERVED	R	0h	Reserved
11-0	HCLKRDIV	R/W	0h	Receive high-frequency clock divide ratio bits determine the divide-down ratio from AUXCLK to AHCLKR. 0h (R/W) = Divide-by-1. 1h (R/W) = Divide-by-2. 2h (R/W) = Divide-by-3 to divide-by-4096 from 2h to FFFh.

Table 11-1302. Register Call Summary for MCASP_AHCLKRCTL

McASP Functional Description • Receive Clock: [0][1][2][3] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0]
McASP Registers • McASP Registers: [0] • MCASP_AHCLKRCTL Register (Offset = 74h) [reset = 0h]: [0][1] • MCASP_GBLCTL Register (Offset = 44h) [reset = 0h]: [0]

11.9.5.19 MCASP_RTDM Register (Offset = 78h) [reset = 0h]

MCASP_RTDM is shown in Figure 11-617 and described in Table 11-1304.

The receive TDM time slot register (MCASP_RTDM) specifies which TDM time slot the receiver is active.

Table 11-1303. MCASP_RTDM Instances

Instance	Physical Address
MCASP_0__CFG	0234 0078h
MCASP_1__CFG	0234 2078h
MCASP_2__CFG	0234 4078h

Figure 11-617. MCASP_RTDM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTDMS																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1304. MCASP_RTDM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RTDMS	R/W	0h	Receiver mode during TDM time slot n. 0h (R/W) = Receive TDM time slot n is inactive. The receive serializer does not shift in data during this slot. 1h (R/W) = Receive TDM time slot n is active. The receive serializer shifts in data during this slot.

Table 11-1305. Register Call Summary for MCASP_RTDM

McASP Functional Description • TDM Time Slots Generation and Processing: [0][1][2] • Transfers Through the Data Port (DATA): [0] • McASP TDM Sequencers: [0] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Special 384-Slot TDM Mode for Connection to External DIR: [0]
McASP Registers • McASP Registers: [0] • MCASP_RTDM Register (Offset = 78h) [reset = 0h]: [0][1]

11.9.5.20 MCASP_RINTCTL Register (Offset = 7Ch) [reset = 0h]

MCASP_RINTCTL is shown in Figure 11-618 and described in Table 11-1307.

The receiver interrupt control register (MCASP_RINTCTL) controls generation of the McASP receive interrupt (RINT). When the register bit(s) is set to 1, the occurrence of the enabled McASP condition(s) generates RINT. See the MCASP_RSTAT register for a description of the interrupt conditions.

Table 11-1306. MCASP_RINTCTL Instances

Instance	Physical Address
MCASP_0__CFG	0234 007Ch
MCASP_1__CFG	0234 207Ch
MCASP_2__CFG	0234 407Ch

Figure 11-618. MCASP_RINTCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RSTAFRM	RESERVED	RDATA	RLAST	RDMAERR	RCKFAIL	RSYNCERR	ROVRN
R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1307. MCASP_RINTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	RSTAFRM	R/W	0h	Receive start of frame interrupt enable bit. 0h (R/W) = Interrupt is disabled. A receive start of frame interrupt does not generate a McASP receive interrupt (RINT). 1h (R/W) = Interrupt is enabled. A receive start of frame interrupt generates a McASP receive interrupt (RINT).
6	RESERVED	R	0h	Reserved
5	RDATA	R/W	0h	Receive data ready interrupt enable bit. 0h (R/W) = Interrupt is disabled. A receive data ready interrupt does not generate a McASP receive interrupt (RINT). 1h (R/W) = Interrupt is enabled. A receive data ready interrupt generates a McASP receive interrupt (RINT).
4	RLAST	R/W	0h	Receive last slot interrupt enable bit. 0h (R/W) = Interrupt is disabled. A receive last slot interrupt does not generate a McASP receive interrupt (RINT). 1h (R/W) = Interrupt is enabled. A receive last slot interrupt generates a McASP receive interrupt (RINT).
3	RDMAERR	R/W	0h	Receive DMA error interrupt enable bit. 0h (R/W) = Interrupt is disabled. A receive DMA error interrupt does not generate a McASP receive interrupt (RINT). 1h (R/W) = Interrupt is enabled. A receive DMA error interrupt generates a McASP receive interrupt (RINT).

Table 11-1307. MCASP_RINTCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	RCKFAIL	R/W	0h	Receive clock failure interrupt enable bit. 0h (R/W) = Interrupt is disabled. A receive clock failure interrupt does not generate a McASP receive interrupt (RINT). 1h (R/W) = Interrupt is enabled. A receive clock failure interrupt generates a McASP receive interrupt (RINT).
1	RSYNCERR	R/W	0h	Unexpected receive frame sync interrupt enable bit. 0h (R/W) = Interrupt is disabled. An unexpected receive frame sync interrupt does not generate a McASP receive interrupt (RINT). 1h (R/W) = Interrupt is enabled. An unexpected receive frame sync interrupt generates a McASP receive interrupt (RINT).
0	ROVRN	R/W	0h	Receiver overrun interrupt enable bit. 0h (R/W) = Interrupt is disabled. A receiver overrun interrupt does not generate a McASP receive interrupt (RINT). 1h (R/W) = Interrupt is enabled. A receiver overrun interrupt generates a McASP receive interrupt (RINT).

Table 11-1308. Register Call Summary for MCASP_RINTCTL

McASP Functional Description • Using the device CPUs for McASP Servicing: [0] • Multiple Interrupts: [0] • McASP Events and Interrupt Requests: [0][1][2][3][4][5][6][7] • Receive Data Ready: [0] • Error Interrupt: [0] • Receive Data Ready Event and Interrupt: [0] • Clock Failure Check Startup: [0] • Burst Transfer Mode: [0]
McASP Registers • McASP Registers: [0] • MCASP_RINTCTL Register (Offset = 7Ch) [reset = 0h]: [0][1] • MCASP_RSTAT Register (Offset = 80h) [reset = 0h]: [0][1][2][3][4][5][6]

11.9.5.21 MCASP_RSTAT Register (Offset = 80h) [reset = 0h]

MCASP_RSTAT is shown in Figure 11-619 and described in Table 11-1310.

The receiver status register (MCASP_RSTAT) provides the receiver status and receive TDM time slot number. If the McASP logic attempts to set an interrupt flag in the same cycle that the CPU writes to the flag to clear it, the McASP logic has priority and the flag remains set. This also causes a new interrupt request to be generated.

Table 11-1309. MCASP_RSTAT Instances

Instance	Physical Address
MCASP_0__CFG	0234 0080h
MCASP_1__CFG	0234 2080h
MCASP_2__CFG	0234 4080h

Figure 11-619. MCASP_RSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							RERR
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RDMAERR	RSTAFRM	RDATA	RLAST	RTDMSLOT	RCKFAIL	RSYNCERR	ROVRN
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W = Read/Write; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-1310. MCASP_RSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	RERR	R/W	0h	RERR bit always returns a logic-OR of: ROVRN OR RSYNCERR OR RCKFAIL OR RDMAERR. Allows a single bit to be checked to determine if a receiver error interrupt has occurred. 0h (R/W) = No errors have occurred. 1h (R/W) = An error has occurred.
7	RDMAERR	R/W1C	0h	Receive DMA error flag. RDMAERR is set when the CPU or DMA reads more serializers through the data port in a given time slot than were programmed as receivers. Causes a receive interrupt (RINT), if this bit is set and RDMAERR in MCASP_RINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 to this bit has no effect. 0h (R/W) = Receive DMA error did not occur. 1h (R/W) = Receive DMA error did occur.
6	RSTAFRM	R/W1C	0h	Receive start of frame flag. Causes a receive interrupt (RINT), if this bit is set and RSTAFRM in MCASP_RINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 to this bit has no effect. 0h (R/W) = No new receive frame sync (AFSR) is detected. 1h (R/W) = A new receive frame sync (AFSR) is detected.

Table 11-1310. MCASP_RSTAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	RDATA	R/W1C	0h	Receive data ready flag. Causes a receive interrupt (RINT), if this bit is set and RDATA in MCASP_RINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 to this bit has no effect. 0h (R/W) = No new data in MCASP_RBUF. 1h (R/W) = Data is transferred from XRSR to MCASP_RBUF and ready to be serviced by the CPU or DMA. When RDATA is set, it always causes a DMA event (AREVT).
4	RLAST	R/W1C	0h	Receive last slot flag. RLAST is set along with RDATA, if the current slot is the last slot in a frame. Causes a receive interrupt (RINT), if this bit is set and RLAST in MCASP_RINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 to this bit has no effect. 0h (R/W) = Current slot is not the last slot in a frame. 1h (R/W) = Current slot is the last slot in a frame. RDATA is also set.
3	RTDMSLOT	R	0h	Returns the LSB of MCASP_RSLOT. Allows a single read of MCASP_RSTAT to determine whether the current TDM time slot is even or odd. 0h (R/W) = Current TDM time slot is odd. 1h (R/W) = Current TDM time slot is even.
2	RCKFAIL	R/W1C	0h	Receive clock failure flag. RCKFAIL is set when the receive clock failure detection circuit reports an error. Causes a receive interrupt (RINT), if this bit is set and RCKFAIL in MCASP_RINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 to this bit has no effect. 0h (R/W) = Receive clock failure did not occur. 1h (R/W) = Receive clock failure did occur.
1	RSYNCERR	R/W1C	0h	Unexpected receive frame sync flag. RSYNCERR is set when a new receive frame sync (AFSR) occurs before it is expected. Causes a receive interrupt (RINT), if this bit is set and RSYNCERR in MCASP_RINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 to this bit has no effect. 0h (R/W) = Unexpected receive frame sync did not occur. 1h (R/W) = Unexpected receive frame sync did occur.
0	ROVRN	R/W1C	0h	Receiver overrun flag. ROVRN is set when the receive serializer is instructed to transfer data from XRSR to MCASP_RBUF, but the former data in MCASP_RBUF has not yet been read by the CPU or DMA. Causes a receive interrupt (RINT), if this bit is set and ROVRN in MCASP_RINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 to this bit has no effect. 0h (R/W) = Receiver overrun did not occur. 1h (R/W) = Receiver overrun did occur.

Table 11-1311. Register Call Summary for MCASP_RSTAT

McASP Functional Description

- Multiple Interrupts: [0][1]
- DATA Port Error-Receiver: [0][1][2]
- Clock Failure Check Startup: [0]
- Error Interrupt: [0][1]
- Receive Data Ready Event and Interrupt: [0][1]
- Receive Clock Failure Check and Recovery: [0]
- Using the device CPUs for McASP Servicing: [0]
- McASP Events and Interrupt Requests: [0][1][2][3][4][5][6][7][8][9][10]
- Receive Data Ready: [0][1]
- McASP State-Machines: [0]
- Buffer Overrun Error-Receiver: [0]

Table 11-1311. Register Call Summary for MCASP_RSTAT (continued)

McASP Registers
• McASP Registers: [0] • MCASP_RINTCTL Register (Offset = 7Ch) [reset = 0h]: [0] • MCASP_RCLKCHK Register (Offset = 88h) [reset = 0h]: [0][1] • MCASP_RSTAT Register (Offset = 80h) [reset = 0h]: [0][1][2]

11.9.5.22 MCASP_RSLOT Register (Offset = 84h) [reset = 0h]

MCASP_RSLOT is shown in Figure 11-620 and described in Table 11-1313.

The current receive TDM time slot register (MCASP_RSLOT) indicates the current time slot for the receive data frame.

Table 11-1312. MCASP_RSLOT Instances

Instance	Physical Address
MCASP_0__CFG	0234 0084h
MCASP_1__CFG	0234 2084h
MCASP_2__CFG	0234 4084h

Figure 11-620. MCASP_RSLOT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							RSLOTCNT								
R-0h																							R-0h								

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1313. MCASP_RSLOT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8-0	RSLOTCNT	R	0h	0-17Fh = Current receive time slot count. Legal values: 0 to 383 (17Fh). TDM function is not supported for > 32 time slots. However, TDM time slot counter may count to 383 when used to receive a DIR block (transferred over TDM format).

Table 11-1314. Register Call Summary for MCASP_RSLOT

McASP Registers
- McASP Registers: [0]
- MCASP_RSLOT Register (Offset = 84h) [reset = 0h]: [0][1]
- MCASP_RSTAT Register (Offset = 80h) [reset = 0h]: [0]

11.9.5.23 MCASP_RCLKCHK Register (Offset = 88h) [reset = 0h]

MCASP_RCLKCHK is shown in Figure 11-621 and described in Table 11-1316.

The receive clock check control register (MCASP_RCLKCHK) configures the receive clock failure detection circuit.

Table 11-1315. MCASP_RCLKCHK Instances

Instance	Physical Address
MCASP_0__CFG	0234 0088h
MCASP_1__CFG	0234 2088h
MCASP_2__CFG	0234 4088h

Figure 11-621. MCASP_RCLKCHK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RCNT								RMAX							
R-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMIN								RESERVED				RPS			
R/W-0h								R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1316. MCASP_RCLKCHK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RCNT	R	0h	Receive clock count value (from previous measurement). The clock circuit continually counts the number of system clocks for every 32 receive high-frequency master clock (AHCLKR) signals, and stores the count in RCNT until the next measurement is taken.
23-16	RMAX	R/W	0h	Receive clock maximum boundary. This 8 bit unsigned value sets the maximum allowed boundary for the clock check counter after 32 receive high-frequency master clock (AHCLKR) signals have been received. If the current counter value is greater than RMAX after counting 32 AHCLKR signals, RCKFAIL in MCASP_RSTAT is set. The comparison is performed using unsigned arithmetic.
15-8	RMIN	R/W	0h	Receive clock minimum boundary. This 8 bit unsigned value sets the minimum allowed boundary for the clock check counter after 32 receive high-frequency master clock (AHCLKR) signals have been received. If RCNT is less than RMIN after counting 32 AHCLKR signals, RCKFAIL in MCASP_RSTAT is set. The comparison is performed using unsigned arithmetic.
7-4	RESERVED	R	0h	Reserved
3-0	RPS	R/W	0h	Receive clock check prescaler value. 0h (R/W) = McASP system clock divided by 1. 1h (R/W) = McASP system clock divided by 2. 2h (R/W) = McASP system clock divided by 4. 3h (R/W) = McASP system clock divided by 8. 4h (R/W) = McASP system clock divided by 16. 5h (R/W) = McASP system clock divided by 32. 6h (R/W) = McASP system clock divided by 64. 7h (R/W) = McASP system clock divided by 128. 8h (R/W) = McASP system clock divided by 256. 9h (R/W) = Reserved from 9h to Fh.

Table 11-1317. Register Call Summary for MCASP_RCLKCHK

McASP Functional Description • Clock Failure Check Startup: [0] • Receive Clock Failure Check and Recovery: [0] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0]
McASP Registers • McASP Registers: [0] • MCASP_RCLKCHK Register (Offset = 88h) [reset = 0h]: [0][1]

11.9.5.24 MCASP_REVTCTL Register (Offset = 8Ch) [reset = 0h]

MCASP_REVTCTL is shown in Figure 11-622 and described in Table 11-1319.

The receiver DMA event control register (MCASP_REVTCTL) contains a disable bit for the receiver DMA event. Note for device-specific registers: Accessing MCASP_REVTCTL not implemented on a specific device may cause improper operation.

Table 11-1318. MCASP_REVTCTL Instances

Instance	Physical Address
MCASP_0__CFG	0234 008Ch
MCASP_1__CFG	0234 208Ch
MCASP_2__CFG	0234 408Ch

Figure 11-622. MCASP_REVTCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RDATDMA
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1319. MCASP_REVTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	RDATDMA	R/W	0h	Receive data DMA request enable bit. If writing to this bit, always write the default value of 0. 0h (R/W) = Receive data DMA request is enabled. 1h (R/W) = Reserved

Table 11-1320. Register Call Summary for MCASP_REVTCTL

McASP Functional Description • McASP DMA Requests: [0] • Using the DMA for McASP Servicing: [0] • Receive Data Ready: [0]
McASP Registers • McASP Registers: [0] • MCASP_REVTCTL Register (Offset = 8Ch) [reset = 0h]: [0][1][2]

11.9.5.25 MCASP_XGBLCTL Register (Offset = A0h) [reset = 0h]

MCASP_XGBLCTL is shown in Figure 11-623 and described in Table 11-1322.

Alias of the global control register (MCASP_GBLCTL). Writing to the transmitter global control register (MCASP_XGBLCTL) affects only the transmit bits of MCASP_GBLCTL (bits 12-8). Reads from MCASP_XGBLCTL return the value of MCASP_GBLCTL. MCASP_XGBLCTL allows the transmitter to be reset independently from the receiver. See the MCASP_GBLCTL register for a detailed description of MCASP_GBLCTL.

Table 11-1321. MCASP_XGBLCTL Instances

Instance	Physical Address
MCASP_0__CFG	0234 00A0h
MCASP_1__CFG	0234 20A0h
MCASP_2__CFG	0234 40A0h

Figure 11-623. MCASP_XGBLCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			XFRST	XSMRST	XSRCLR	XHCLKRST	XCLKRST
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED			RFRST	RSMRST	RSRCLR	RHCLKRST	RCLKRST
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1322. MCASP_XGBLCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12	XFRST	R/W	0h	Transmit frame sync generator reset enable bit. A write to this bit affects the XFRST bit of MCASP_GBLCTL. 0h (R/W) = Transmit frame sync generator is reset. 1h (R/W) = Transmit frame sync generator is active.
11	XSMRST	R/W	0h	Transmit state machine reset enable bit. A write to this bit affects the XSMRST bit of MCASP_GBLCTL. 0h (R/W) = Transmit state machine is held in reset. 1h (R/W) = Transmit state machine is released from reset.
10	XSRCLR	R/W	0h	Transmit serializer clear enable bit. A write to this bit affects the XSRCLR bit of MCASP_GBLCTL. 0h (R/W) = Transmit serializers are cleared. 1h (R/W) = Transmit serializers are active.
9	XHCLKRST	R/W	0h	Transmit high-frequency clock divider reset enable bit. A write to this bit affects the XHCLKRST bit of MCASP_GBLCTL. 0h (R/W) = Transmit high-frequency clock divider is held in reset. 1h (R/W) = Transmit high-frequency clock divider is running.
8	XCLKRST	R/W	0h	Transmit clock divider reset enable bit. A write to this bit affects the XCLKRST bit of MCASP_GBLCTL. 0h (R/W) = Transmit clock divider is held in reset. 1h (R/W) = Transmit clock divider is running.

Table 11-1322. MCASP_XGBLCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-5	RESERVED	R	0h	Reserved
4	RFRST	R	0h	Receive frame sync generator reset enable bit. A read of this bit returns the RFRST bit value of MCASP_GBLCTL. Writes have no effect.
3	RSMRST	R	0h	Receive state machine reset enable bit. A read of this bit returns the RSMRST bit value of MCASP_GBLCTL. Writes have no effect.
2	RSRCLR	R	0h	Receive serializer clear enable bit. A read of this bit returns the RSRSCLR bit value of MCASP_GBLCTL. Writes have no effect.
1	RHCLKRST	R	0h	Receive high-frequency clock divider reset enable bit. A read of this bit returns the RHCLKRST bit value of MCASP_GBLCTL. Writes have no effect.
0	RCLKRST	R	0h	Receive clock divider reset enable bit. A read of this bit returns the RCLKRST bit value of MCASP_GBLCTL. Writes have no effect.

Table 11-1323. Register Call Summary for MCASP_XGBLCTL

McASP Registers

- McASP Registers: [0]
- MCASP_XGBLCTL Register (Offset = A0h) [reset = 0h]: [0][1][2][3]

11.9.5.26 MCASP_XMASK Register (Offset = A4h) [reset = 0h]

MCASP_XMASK is shown in Figure 11-624 and described in Table 11-1325.

The transmit format unit bit mask register (MCASP_XMASK) determines which bits of the transmitted data are masked off and padded with a known value before being shifted out the McASP.

Table 11-1324. MCASP_XMASK Instances

Instance	Physical Address
MCASP_0__CFG	0234 00A4h
MCASP_1__CFG	0234 20A4h
MCASP_2__CFG	0234 40A4h

Figure 11-624. MCASP_XMASK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XMASK																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1325. MCASP_XMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XMASK	R/W	0h	Transmit data mask n enable bit. 0h (R/W) = Corresponding bit of transmit data (before passing through reverse and rotate units) is masked out and then padded with the selected bit pad value (XPAD and XPBIT bits in MCASP_XFMT), which is transmitted out the McASP in place of the original bit. 1h (R/W) = Corresponding bit of transmit data (before passing through reverse and rotate units) is transmitted out the McASP.

Table 11-1326. Register Call Summary for MCASP_XMASK

McASP Functional Description • DIT Mode Transmission Data Alignment Settings: [0][1][2][3][4][5][6] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit DIT Clock and Frame-Sync Generation: [0] • Transmit Format Unit: [0][1]
McASP Registers • McASP Registers: [0] • MCASP_XMASK Register (Offset = A4h) [reset = 0h]: [0][1] • MCASP_XFMT Register (Offset = A8h) [reset = 0h]: [0][1]

11.9.5.27 MCASP_XFMT Register (Offset = A8h) [reset = 0h]

MCASP_XFMT is shown in Figure 11-625 and described in Table 11-1328.

The transmit bit stream format register (MCASP_XFMT) configures the transmit data format.

Table 11-1327. MCASP_XFMT Instances

Instance	Physical Address
MCASP_0__CFG	0234 00A8h
MCASP_1__CFG	0234 20A8h
MCASP_2__CFG	0234 40A8h

Figure 11-625. MCASP_XFMT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						XDATDLY	
R-0h						R/W-0h	
15	14	13	12	11	10	9	8
XRVRS	XPAD			XPBIT			
R/W-0h	R/W-0h			R/W-0h			
7	6	5	4	3	2	1	0
XSSZ				XBUSEL	XROT		
R/W-0h				R/W-0h	R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1328. MCASP_XFMT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17-16	XDATDLY	R/W	0h	Transmit sync bit delay. 0h (R/W) = 0-bit delay. The first transmit data bit, AXRn, occurs in same ACLKX cycle as the transmit frame sync (AFSX). 1h (R/W) = 1-bit delay. The first transmit data bit, AXRn, occurs one ACLKX cycle after the transmit frame sync (AFSX). 2h (R/W) = 2-bit delay. The first transmit data bit, AXRn, occurs two ACLKX cycles after the transmit frame sync (AFSX). 3h (R/W) = Reserved.
15	XRVRS	R/W	0h	Transmit serial bitstream order. 0h (R/W) = Bitstream is LSB first. No bit reversal is performed in transmit format bit reverse unit. 1h (R/W) = Bitstream is MSB first. Bit reversal is performed in transmit format bit reverse unit.
14-13	XPAD	R/W	0h	Pad value for extra bits in slot not belonging to word defined by MCASP_XMASK. This field only applies to bits when MCASP_XMASK[n] = 0. 0h (R/W) = Pad extra bits with 0. 1h (R/W) = Pad extra bits with 1. 2h (R/W) = Pad extra bits with one of the bits from the word as specified by XPBIT bits. 3h (R/W) = Reserved.

Table 11-1328. MCASP_XFMT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12-8	XPBIT	R/W	0h	XPBIT value determines which bit (as written by the CPU or DMA to MCASP_XBUF[n]) is used to pad the extra bits before shifting. This field only applies when XPAD = 2h. 0h (R/W) = Pad with bit 0 value. 1h (R/W) = Pad with bit 1 to bit 31 value from 1h to 1Fh.
7-4	XSSZ	R/W	0h	Transmit slot size. 0h (R/W) = Reserved. 1h (R/W) = Reserved. 2h (R/W) = Reserved. 3h (R/W) = Slot size is 8 bits. 4h (R/W) = Reserved. 5h (R/W) = Slot size is 12 bits. 6h (R/W) = Reserved. 7h (R/W) = Slot size is 16 bits. 8h (R/W) = Reserved. 9h (R/W) = Slot size is 20 bits. Ah (R/W) = Reserved. Bh (R/W) = Slot size is 24 bits. Ch (R/W) = Reserved. Dh (R/W) = Slot size is 28 bits. Eh (R/W) = Reserved. Fh (R/W) = Slot size is 32 bits.
3	XBUSEL	R/W	0h	Selects whether writes to serializer buffer XRBUF[n] originate from the configuration bus (CFG) or the data (DAT) port. 0h (R/W) = Writes to XRBUF[n] originate from the data port. Writes to XRBUF[n] from the configuration bus are ignored with no effect to the McASP. 1h (R/W) = Writes to XRBUF[n] originate from the configuration bus. Writes to XRBUF[n] from the data port are ignored with no effect to the McASP.
2-0	XROT	R/W	0h	Right-rotation value for transmit rotate right format unit. 0h (R/W) = Rotate right by 0 (no rotation). 1h (R/W) = Rotate right by 4 bit positions. 2h (R/W) = Rotate right by 8 bit positions. 3h (R/W) = Rotate right by 12 bit positions. 4h (R/W) = Rotate right by 16 bit positions. 5h (R/W) = Rotate right by 20 bit positions. 6h (R/W) = Rotate right by 24 bit positions. 7h (R/W) = Rotate right by 28 bit positions.

Table 11-1329. Register Call Summary for MCASP_XFMT

McASP Functional Description

- McASP Format Units: [0][1][2]
- Transfers Through the Configuration Bus (CFG): [0]
- Burst Transfer Mode: [0][1]
- DIT Mode Transmission Data Alignment Settings: [0][1][2][3][4][5][6]
- Frame-Sync Generator: [0]
- Transfers Through the Data Port (DATA): [0]
- Transmit Format Unit: [0][1][2]
- TDM Mode Transmission Data Alignment Settings: [0]
- McASP State-Machines: [0]
- Time-Division Multiplexed (TDM) Transfer Mode: [0]
- Transmit DIT Clock and Frame-Sync Generation: [0][1]

Table 11-1329. Register Call Summary for MCASP_XFMT (continued)

McASP Registers
• McASP Registers: [0][1] • MCASP_XBUF Register (Offset = 0h) [reset = 0h]: [0] • MCASP_XMASK Register (Offset = A4h) [reset = 0h]: [0] • MCASP_XFMT Register (Offset = A8h) [reset = 0h]: [0][1]

11.9.5.28 MCASP_AFSXCTL Register (Offset = ACh) [reset = 0h]

MCASP_AFSXCTL is shown in Figure 11-626 and described in Table 11-1331.

The transmit frame sync control register (MCASP_AFSXCTL) configures the transmit frame sync (AFSX).

Table 11-1330. MCASP_AFSXCTL Instances

Instance	Physical Address
MCASP_0__CFG	0234 00ACh
MCASP_1__CFG	0234 20ACh
MCASP_2__CFG	0234 40ACh

Figure 11-626. MCASP_AFSXCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
XMOD							
R/W-0h							
7	6	5	4	3	2	1	0
XMOD	RESERVED		FXWID	RESERVED		FSXM	FSXP
R/W-0h	R-0h		R/W-0h	R-0h		R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1331. MCASP_AFSXCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-7	XMOD	R/W	0h	Transmit frame sync mode select bits. 1FFh = Reserved from 181h to 1FFh. 0h (R/W) = Burst mode. 1h (R/W) = Reserved. 2h (R/W) = 2-slot TDM (I2S mode) to 32-slot TDM from 2h to 20h. 21h (R/W) = Reserved from 21h to 17Fh. 180h (R/W) = 384-slot DIT mode. 181h (R/W) = Reserved from 181h to 1FFh.
6-5	RESERVED	R	0h	Reserved
4	FXWID	R/W	0h	Transmit frame sync width select bit indicates the width of the transmit frame sync (AFSX) during its active period. 0h (R/W) = Single bit. 1h (R/W) = Single word.
3-2	RESERVED	R	0h	Reserved
1	FSXM	R/W	0h	Transmit frame sync generation select bit. 0h (R/W) = Externally-generated transmit frame sync. 1h (R/W) = Internally-generated transmit frame sync.
0	FSXP	R/W	0h	Transmit frame sync polarity select bit. 0h (R/W) = A rising edge on transmit frame sync (AFSX) indicates the beginning of a frame. 1h (R/W) = A falling edge on transmit frame sync (AFSX) indicates the beginning of a frame.

Table 11-1332. Register Call Summary for MCASP_AFSXCTL

McASP Functional Description • Frame-Sync Generator: [0][1][2][3][4][5][6] • Burst Transfer Mode: [0][1] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit DIT Clock and Frame-Sync Generation: [0][1] • Loopback Mode Configurations: [0]
McASP Registers • MCASP_AFSXCTL Register (Offset = ACh) [reset = 0h]: [0][1] • McASP Registers: [0]

11.9.5.29 MCASP_ACLKXCTL Register (Offset = B0h) [reset = 60h]

MCASP_ACLKXCTL is shown in Figure 11-627 and described in Table 11-1334.

The transmit clock control register (MCASP_ACLKXCTL) configures the transmit bit clock (ACLKX) and the transmit clock generator.

Table 11-1333. MCASP_ACLKXCTL Instances

Instance	Physical Address
MCASP_0__CFG	0234 00B0h
MCASP_1__CFG	0234 20B0h
MCASP_2__CFG	0234 40B0h

Figure 11-627. MCASP_ACLKXCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CLKXP	ASYNC	CLKXM	CLKXDIV				
R/W-0h	R/W-1h	R/W-1h	R/W-0h				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1334. MCASP_ACLKXCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	CLKXP	R/W	0h	Transmit bitstream clock polarity select bit. 0h (R/W) = Rising edge. External receiver samples data on the falling edge of the serial clock, so the transmitter must shift data out on the rising edge of the serial clock. 1h (R/W) = Falling edge. External receiver samples data on the rising edge of the serial clock, so the transmitter must shift data out on the falling edge of the serial clock.
6	ASYNC	R/W	1h	Transmit/receive operation asynchronous enable bit. 0h (R/W) = Synchronous. Transmit clock and frame sync provides the source for both the transmit and receive sections. 1h (R/W) = Asynchronous. Separate clock and frame sync used by transmit and receive sections.
5	CLKXM	R/W	1h	Transmit bit clock source bit. 0h (R/W) = External transmit clock source from ACLKX pin. 1h (R/W) = Internal transmit clock source from output of programmable bit clock divider.
4-0	CLKXDIV	R/W	0h	Transmit bit clock divide ratio bits determine the divide-down ratio from AHCLKX to ACLKX. 0h (R/W) = Divide-by-1. 1h (R/W) = Divide-by-2. 2h (R/W) = Divide-by-3 to divide-by-32 from 2h to 1Fh.

Table 11-1335. Register Call Summary for MCASP_ACLKXCTL

McASP Functional Description • Receive Clock: [0] • Frame-Sync Generator: [0][1] • McASP Synchronous and Asynchronous Transmit and Receive Operations: [0][1][2] • Transmit DIT Clock and Frame-Sync Generation: [0][1] • McASP Loopback Modes: [0] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit Clock: [0][1][2][3][4][5] • Loopback Mode Configurations: [0]
McASP Registers • McASP Registers: [0] • MCASP_ACLKXCTL Register (Offset = B0h) [reset = 60h]: [0][1] • MCASP_ACLKRCTL Register (Offset = 70h) [reset = 0h]: [0][1] • MCASP_GBLCTL Register (Offset = 44h) [reset = 0h]: [0] • MCASP_AHCLKXCTL Register (Offset = B4h) [reset = 0h]: [0][1]

11.9.5.30 MCASP_AHCLKXCTL Register (Offset = B4h) [reset = 0h]

MCASP_AHCLKXCTL is shown in Figure 11-628 and described in Table 11-1337.

The transmit high-frequency clock control register (MCASP_AHCLKXCTL) configures the transmit high-frequency master clock (AHCLKX) and the transmit clock generator.

Table 11-1336. MCASP_AHCLKXCTL Instances

Instance	Physical Address
MCASP_0__CFG	0234 00B4h
MCASP_1__CFG	0234 20B4h
MCASP_2__CFG	0234 40B4h

Figure 11-628. MCASP_AHCLKXCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
HCLKXM	HCLKXP	RESERVED		HCLKXDIV			
R/W-0h	R/W-0h	R-0h		R/W-0h			
7	6	5	4	3	2	1	0
HCLKXDIV							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1337. MCASP_AHCLKXCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	HCLKXM	R/W	0h	Transmit high-frequency clock source bit. 0h (R/W) = External transmit high-frequency clock source from AHCLKX pin. 1h (R/W) = Internal transmit high-frequency clock source from output of programmable high clock divider.
14	HCLKXP	R/W	0h	Transmit bitstream high-frequency clock polarity select bit. 0h (R/W) = AHCLKX is not inverted before programmable bit clock divider. In the special case where the transmit bit clock (ACLKX) is internally generated and the programmable bit clock divider is set to divide-by-1 (CLKXDIV = 0 in MCASP_ACLKXCTL), AHCLKX is directly passed through to the ACLKX pin. 1h (R/W) = AHCLKX is inverted before programmable bit clock divider. In the special case where the transmit bit clock (ACLKX) is internally generated and the programmable bit clock divider is set to divide-by-1 (CLKXDIV = 0 in MCASP_ACLKXCTL), AHCLKX is directly passed through to the ACLKX pin.
13-12	RESERVED	R	0h	Reserved
11-0	HCLKXDIV	R/W	0h	Transmit high-frequency clock divide ratio bits determine the divide-down ratio from AUXCLK to AHCLKX. 0h (R/W) = Divide-by-1. 1h (R/W) = Divide-by-2. 2h (R/W) = Divide-by-3 to divide-by-4096 from 2h to FFFh.

Table 11-1338. Register Call Summary for MCASP_AHCLKXCTL

McASP Functional Description • Transmit DIT Clock and Frame-Sync Generation: [0] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit Clock: [0][1][2][3][4]
McASP Registers • McASP Registers: [0] • MCASP_GBLCTL Register (Offset = 44h) [reset = 0h]: [0] • MCASP_AHCLKXCTL Register (Offset = B4h) [reset = 0h]: [0][1]

11.9.5.31 MCASP_XTDM Register (Offset = B8h) [reset = 0h]

MCASP_XTDM is shown in Figure 11-629 and described in Table 11-1340.

The transmit TDM time slot register (MCASP_XTDM) specifies in which TDM time slot the transmitter is active. TDM time slot counter range is extended to 384 slots (to support SPDIF blocks of 384 subframes). MCASP_XTDM operates modulo 32, that is, XTDMS specifies the TDM activity for time slots 0, 32, 64, 96, 128, and so on.

Table 11-1339. MCASP_XTDM Instances

Instance	Physical Address
MCASP_0__CFG	0234 00B8h
MCASP_1__CFG	0234 20B8h
MCASP_2__CFG	0234 40B8h

Figure 11-629. MCASP_XTDM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XTDMS																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1340. MCASP_XTDM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XTDMS	R/W	0h	Transmitter mode during TDM time slot n. 0h (R/W) = Transmit TDM time slot n is inactive. The transmit serializer does not shift out data during this slot. 1h (R/W) = Transmit TDM time slot n is active. The transmit serializer shifts out data during this slot according to the serializer control register (SRCTL).

Table 11-1341. Register Call Summary for MCASP_XTDM

McASP Functional Description • TDM Time Slots Generation and Processing: [0][1][2] • Transfers Through the Data Port (DATA): [0] • McASP TDM Sequencers: [0][1] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit DIT Clock and Frame-Sync Generation: [0] • Transmit DIT Encoding: [0]
McASP Registers • McASP Registers: [0] • MCASP_XTDM Register (Offset = B8h) [reset = 0h]: [0][1][2]

11.9.5.32 MCASP_XINTCTL Register (Offset = BCh) [reset = 0h]

MCASP_XINTCTL is shown in Figure 11-630 and described in Table 11-1343.

The transmitter interrupt control register (MCASP_XINTCTL) controls generation of the McASP transmit interrupt (XINT). When the register bit(s) is set to 1, the occurrence of the enabled McASP condition(s) generates XINT. See the MCASP_XSTAT register for a description of the interrupt conditions.

Table 11-1342. MCASP_XINTCTL Instances

Instance	Physical Address
MCASP_0__CFG	0234 00BCh
MCASP_1__CFG	0234 20BCh
MCASP_2__CFG	0234 40BCh

Figure 11-630. MCASP_XINTCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
XSTAFRM	RESERVED	XDATA	XLAST	XDMAERR	XCKFAIL	XSYNCERR	XUNDRN
R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1343. MCASP_XINTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	XSTAFRM	R/W	0h	Transmit start of frame interrupt enable bit. 0h (R/W) = Interrupt is disabled. A transmit start of frame interrupt does not generate a McASP transmit interrupt (XINT). 1h (R/W) = Interrupt is enabled. A transmit start of frame interrupt generates a McASP transmit interrupt (XINT).
6	RESERVED	R	0h	Reserved
5	XDATA	R/W	0h	Transmit data ready interrupt enable bit. 0h (R/W) = Interrupt is disabled. A transmit data ready interrupt does not generate a McASP transmit interrupt (XINT). 1h (R/W) = Interrupt is enabled. A transmit data ready interrupt generates a McASP transmit interrupt (XINT).
4	XLAST	R/W	0h	Transmit last slot interrupt enable bit. 0h (R/W) = Interrupt is disabled. A transmit last slot interrupt does not generate a McASP transmit interrupt (XINT). 1h (R/W) = Interrupt is enabled. A transmit last slot interrupt generates a McASP transmit interrupt (XINT).
3	XDMAERR	R/W	0h	Transmit DMA error interrupt enable bit. 0h (R/W) = Interrupt is disabled. A transmit DMA error interrupt does not generate a McASP transmit interrupt (XINT). 1h (R/W) = Interrupt is enabled. A transmit DMA error interrupt generates a McASP transmit interrupt (XINT).

Table 11-1343. MCASP_XINTCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	XCKFAIL	R/W	0h	Transmit clock failure interrupt enable bit. 0h (R/W) = Interrupt is disabled. A transmit clock failure interrupt does not generate a McASP transmit interrupt (XINT). 1h (R/W) = Interrupt is enabled. A transmit clock failure interrupt generates a McASP transmit interrupt (XINT).
1	XSYNCERR	R/W	0h	Unexpected transmit frame sync interrupt enable bit. 0h (R/W) = Interrupt is disabled. An unexpected transmit frame sync interrupt does not generate a McASP transmit interrupt (XINT). 1h (R/W) = Interrupt is enabled. An unexpected transmit frame sync interrupt generates a McASP transmit interrupt (XINT).
0	XUNDRN	R/W	0h	Transmitter underrun interrupt enable bit. 0h (R/W) = Interrupt is disabled. A transmitter underrun interrupt does not generate a McASP transmit interrupt (XINT). 1h (R/W) = Interrupt is enabled. A transmitter underrun interrupt generates a McASP transmit interrupt (XINT).

Table 11-1344. Register Call Summary for MCASP_XINTCTL

McASP Functional Description
- Multiple Interrupts: [0][1]
- Transmit Data Ready Event and Interrupt: [0]
- Transmit Clock Failure Check and Recovery: [0]
- Error Interrupt: [0]
- Burst Transfer Mode: [0]
- Using the device CPUs for McASP Servicing: [0]
- McASP Events and Interrupt Requests: [0][1][2][3][4][5][6][7]
- Transmit Data Ready: [0]
- Clock Failure Check Startup: [0]
- Time-Division Multiplexed (TDM) Transfer Mode: [0]
- Transmit DIT Clock and Frame-Sync Generation: [0]

McASP Registers
- McASP Registers: [0]
- MCASP_XSTAT Register (Offset = C0h) [reset = 0h]: [0][1][2][3][4][5][6]
- MCASP_XINTCTL Register (Offset = BCh) [reset = 0h]: [0][1]

11.9.5.33 MCASP_XSTAT Register (Offset = C0h) [reset = 0h]

MCASP_XSTAT is shown in Figure 11-631 and described in Table 11-1346.

The transmitter status register (MCASP_XSTAT) provides the transmitter status and transmit TDM time slot number. If the McASP logic attempts to set an interrupt flag in the same cycle that the CPU writes to the flag to clear it, the McASP logic has priority and the flag remains set. This also causes a new interrupt request to be generated.

Table 11-1345. MCASP_XSTAT Instances

Instance	Physical Address
MCASP_0__CFG	0234 00C0h
MCASP_1__CFG	0234 20C0h
MCASP_2__CFG	0234 40C0h

Figure 11-631. MCASP_XSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							XERR
R-0h							R/W-0h
7	6	5	4	3	2	1	0
XDMAERR	XSTAFRM	XDATA	XLAST	XTDMSLOT	XCKFAIL	XSYNCERR	XUNDRN
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W = Read/Write; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-1346. MCASP_XSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	XERR	R/W	0h	XERR bit always returns a logic-OR of: XUNDRN OR XSYNCERR OR XCKFAIL OR XDMAERR. Allows a single bit to be checked to determine if a transmitter error interrupt has occurred. 0h (R/W) = No errors have occurred. 1h (R/W) = An error has occurred.
7	XDMAERR	R/W1C	0h	Transmit DMA error flag. XDMAERR is set when the CPU or DMA writes more serializers through the data port in a given time slot than were programmed as transmitters. Causes a transmit interrupt (XINT), if this bit is set and XDMAERR in MCASP_XINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 has no effect. 0h (R/W) = Transmit DMA error did not occur. 1h (R/W) = Transmit DMA error did occur.
6	XSTAFRM	R/W1C	0h	Transmit start of frame flag. Causes a transmit interrupt (XINT), if this bit is set and XSTAFRM in MCASP_XINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 has no effect. 0h (R/W) = No new transmit frame sync (AFSX) is detected. 1h (R/W) = A new transmit frame sync (AFSX) is detected.

Table 11-1346. MCASP_XSTAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	XDATA	R/W1C	0h	Transmit data ready flag. Causes a transmit interrupt (XINT), if this bit is set and XDATA in MCASP_XINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 has no effect. 0h (R/W) = MCASP_XBUF is written and is full. 1h (R/W) = Data is copied from MCASP_XBUF to XRSR. MCASP_XBUF is empty and ready to be written. XDATA is also set when the transmit serializers are taken out of reset. When XDATA is set, it always causes a DMA event (AXEVT).
4	XLAST	R/W1C	0h	Transmit last slot flag. XLAST is set along with XDATA, if the current slot is the last slot in a frame. Causes a transmit interrupt (XINT), if this bit is set and XLAST in MCASP_XINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 has no effect. 0h (R/W) = Current slot is not the last slot in a frame. 1h (R/W) = Current slot is the last slot in a frame. XDATA is also set.
3	XTDMSLOT	R	0h	Returns the LSB of MCASP_XSLOT. Allows a single read of MCASP_XSTAT to determine whether the current TDM time slot is even or odd. 0h (R/W) = Current TDM time slot is odd. 1h (R/W) = Current TDM time slot is even.
2	XCKFAIL	R/W1C	0h	Transmit clock failure flag. XCKFAIL is set when the transmit clock failure detection circuit reports an error. Causes a transmit interrupt (XINT), if this bit is set and XCKFAIL in MCASP_XINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 has no effect. 0h (R/W) = Transmit clock failure did not occur. 1h (R/W) = Transmit clock failure did occur.
1	XSYNCERR	R/W1C	0h	Unexpected transmit frame sync flag. XSYNCERR is set when a new transmit frame sync (AFSX) occurs before it is expected. Causes a transmit interrupt (XINT), if this bit is set and XSYNCERR in MCASP_XINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 has no effect. 0h (R/W) = Unexpected transmit frame sync did not occur. 1h (R/W) = Unexpected transmit frame sync did occur.
0	XUNDRN	R/W1C	0h	Transmitter underrun flag. XUNDRN is set when the transmit serializer is instructed to transfer data from MCASP_XBUF to XRSR, but MCASP_XBUF has not yet been serviced with new data since the last transfer. Causes a transmit interrupt (XINT), if this bit is set and XUNDRN in MCASP_XINTCTL is set. This bit is cleared by writing a 1 to this bit. Writing a 0 has no effect. 0h (R/W) = Transmitter underrun did not occur. 1h (R/W) = Transmitter underrun did occur. For details on McASP action upon underrun conditions, see Buffer Underrun Error - Transmitter.

Table 11-1347. Register Call Summary for MCASP_XSTAT

McASP Functional Description

- DATA Port Error-Transmitter: [0][1][2]
- Multiple Interrupts: [0][1][2][3]
- Buffer Underrun Error-Transmitter: [0]
- Transmit Data Ready Event and Interrupt: [0][1]
- Transmit Clock Failure Check and Recovery: [0][1]
- Error Interrupt: [0][1]
- Using the device CPUs for McASP Servicing: [0]
- McASP Events and Interrupt Requests: [0][1][2][3][4][5][6][7][8][9][10]
- Transmit Data Ready: [0][1][2][3]
- Clock Failure Check Startup: [0]
- McASP State-Machines: [0]

Table 11-1347. Register Call Summary for MCASP_XSTAT (continued)

McASP Registers
• McASP Registers: [0] • MCASP_XSTAT Register (Offset = C0h) [reset = 0h]: [0][1][2] • MCASP_XCLKCHK Register (Offset = C8h) [reset = 0h]: [0][1] • MCASP_XINTCTL Register (Offset = BCh) [reset = 0h]: [0] • MCASP_GBLCTL Register (Offset = 44h) [reset = 0h]: [0][1]

11.9.5.34 MCASP_XSLOT Register (Offset = C4h) [reset = 0h]

MCASP_XSLOT is shown in Figure 11-632 and described in Table 11-1349.

The current transmit TDM time slot register (MCASP_XSLOT) indicates the current time slot for the transmit data frame.

Table 11-1348. MCASP_XSLOT Instances

Instance	Physical Address
MCASP_0__CFG	0234 00C4h
MCASP_1__CFG	0234 20C4h
MCASP_2__CFG	0234 40C4h

Figure 11-632. MCASP_XSLOT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						XSLOTCNT									
R-0h																						R-0h									

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1349. MCASP_XSLOT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9-0	XSLOTCNT	R	0h	Current transmit time slot count. Legal values: 0 to 383 (17Fh). During reset, this counter value is 383 so the next count value, which is used to encode the first DIT group of data, will be 0 and encodes the B preamble. TDM function is not supported for >32 time slots. However, TDM time slot counter may count to 383 when used to transmit a DIT block.

Table 11-1350. Register Call Summary for MCASP_XSLOT

McASP Functional Description • McASP TDM Sequencers: [0]
McASP Registers • McASP Registers: [0] • MCASP_XSTAT Register (Offset = C0h) [reset = 0h]: [0] • MCASP_XSLOT Register (Offset = C4h) [reset = 0h]: [0][1]

11.9.5.35 MCASP_XCLKCHK Register (Offset = C8h) [reset = 0h]

MCASP_XCLKCHK is shown in Figure 11-633 and described in Table 11-1352.

The transmit clock check control register (MCASP_XCLKCHK) configures the transmit clock failure detection circuit.

Table 11-1351. MCASP_XCLKCHK Instances

Instance	Physical Address
MCASP_0__CFG	0234 00C8h
MCASP_1__CFG	0234 20C8h
MCASP_2__CFG	0234 40C8h

Figure 11-633. MCASP_XCLKCHK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
XCNT								XMAX							
R-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XMIN								RESERVED				XPS			
R/W-0h								R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1352. MCASP_XCLKCHK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	XCNT	R	0h	Transmit clock count value (from previous measurement). The clock circuit continually counts the number of system clocks for every 32 transmit high-frequency master clock (AHCLKX) signals, and stores the count in XCNT until the next measurement is taken.
23-16	XMAX	R/W	0h	Transmit clock maximum boundary. This 8 bit unsigned value sets the maximum allowed boundary for the clock check counter after 32 transmit high-frequency master clock (AHCLKX) signals have been received. If the current counter value is greater than XMAX after counting 32 AHCLKX signals, XCKFAIL in MCASP_XSTAT is set. The comparison is performed using unsigned arithmetic.
15-8	XMIN	R/W	0h	Transmit clock minimum boundary. This 8 bit unsigned value sets the minimum allowed boundary for the clock check counter after 32 transmit high-frequency master clock (AHCLKX) signals have been received. If XCNT is less than XMIN after counting 32 AHCLKX signals, XCKFAIL in MCASP_XSTAT is set. The comparison is performed using unsigned arithmetic.
7-4	RESERVED	R	0h	Reserved
3-0	XPS	R/W	0h	Transmit clock check prescaler value. Fh = Reserved from 9h to Fh. 0h (R/W) = McASP system clock divided by 1. 1h (R/W) = McASP system clock divided by 2. 2h (R/W) = McASP system clock divided by 4. 3h (R/W) = McASP system clock divided by 8. 4h (R/W) = McASP system clock divided by 16. 5h (R/W) = McASP system clock divided by 32. 6h (R/W) = McASP system clock divided by 64. 7h (R/W) = McASP system clock divided by 128. 8h (R/W) = McASP system clock divided by 256. 9h (R/W) = Reserved from 9h to Fh.

Table 11-1353. Register Call Summary for MCASP_XCLKCHK

McASP Functional Description • Clock Failure Check Startup: [0] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit DIT Clock and Frame-Sync Generation: [0] • Transmit Clock Failure Check and Recovery: [0]
McASP Registers • McASP Registers: [0] • MCASP_XCLKCHK Register (Offset = C8h) [reset = 0h]: [0][1]

11.9.5.36 MCASP_XEVTCTL Register (Offset = CCh) [reset = 0h]

MCASP_XEVTCTL is shown in Figure 11-634 and described in Table 11-1355.

The transmitter DMA event control register (MCASP_XEVTCTL) contains a disable bit for the transmit DMA event. Note for device-specific registers: Accessing MCASP_XEVTCTL not implemented on a specific device may cause improper device operation.

Table 11-1354. MCASP_XEVTCTL Instances

Instance	Physical Address
MCASP_0__CFG	0234 00CCh
MCASP_1__CFG	0234 20CCh
MCASP_2__CFG	0234 40CCh

Figure 11-634. MCASP_XEVTCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							XDATDMA
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1355. MCASP_XEVTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	XDATDMA	R/W	0h	Transmit data DMA request enable bit. If writing to this bit, always write the default value of 0. 0h (R/W) = Transmit data DMA request is enabled. 1h (R/W) = Reserved

Table 11-1356. Register Call Summary for MCASP_XEVTCTL

McASP Functional Description • McASP DMA Requests: [0] • Using the DMA for McASP Servicing: [0] • Transmit Data Ready: [0]
McASP Registers • McASP Registers: [0] • MCASP_XEVTCTL Register (Offset = CCh) [reset = 0h]: [0][1][2]

11.9.5.37 MCASP_DITCSRA0 Register (Offset = 100h) [reset = 0h]

MCASP_DITCSRA0 is shown in Figure 11-635 and described in Table 11-1358.

The DIT left channel status registers (MCASP_DITCSRA0) provide the status of each left channel (even TDM time slot). Each of the six 32-bit registers can store 192 bits of channel status data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register file in time, if a different set of data need to be sent.

Table 11-1357. MCASP_DITCSRA0 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0100h
MCASP_1__CFG	0234 2100h
MCASP_2__CFG	0234 4100h

Figure 11-635. MCASP_DITCSRA0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITCSRA																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1358. MCASP_DITCSRA0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITCSRA	R/W	0h	DIT left channel status registers.

Table 11-1359. Register Call Summary for MCASP_DITCSRA0

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2][3][4][5][6] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit DIT Clock and Frame-Sync Generation: [0]
McASP Registers • McASP Registers: [0] • MCASP_DITCSRA0 Register (Offset = 100h) [reset = 0h]: [0][1]

11.9.5.38 MCASP_DITCSRA1 Register (Offset = 104h) [reset = 0h]

MCASP_DITCSRA1 is shown in Figure 11-636 and described in Table 11-1361.

The DIT left channel status registers (DITCSRA) provide the status of each left channel (even TDM time slot). Each of the six 32-bit registers can store 192 bits of channel status data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register file in time, if a different set of data need to be sent.

Table 11-1360. MCASP_DITCSRA1 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0104h
MCASP_1__CFG	0234 2104h
MCASP_2__CFG	0234 4104h

Figure 11-636. MCASP_DITCSRA1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITCSRA																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1361. MCASP_DITCSRA1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITCSRA	R/W	0h	DIT left channel status registers.

Table 11-1362. Register Call Summary for MCASP_DITCSRA1

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2]
McASP Registers • McASP Registers: [0] • MCASP_DITCSRA1 Register (Offset = 104h) [reset = 0h]: [0]

11.9.5.39 MCASP_DITCSRA2 Register (Offset = 108h) [reset = 0h]

MCASP_DITCSRA2 is shown in Figure 11-637 and described in Table 11-1364.

The DIT left channel status registers (DITCSRA) provide the status of each left channel (even TDM time slot). Each of the six 32-bit registers can store 192 bits of channel status data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register file in time, if a different set of data need to be sent.

Table 11-1363. MCASP_DITCSRA2 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0108h
MCASP_1__CFG	0234 2108h
MCASP_2__CFG	0234 4108h

Figure 11-637. MCASP_DITCSRA2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITCSRA																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1364. MCASP_DITCSRA2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITCSRA	R/W	0h	DIT left channel status registers.

Table 11-1365. Register Call Summary for MCASP_DITCSRA2

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2]
McASP Registers • MCASP_DITCSRA2 Register (Offset = 108h) [reset = 0h]: [0] • McASP Registers: [0]

11.9.5.40 MCASP_DITCSRA3 Register (Offset = 10Ch) [reset = 0h]

MCASP_DITCSRA3 is shown in Figure 11-638 and described in Table 11-1367.

The DIT left channel status registers (DITCSRA) provide the status of each left channel (even TDM time slot). Each of the six 32-bit registers can store 192 bits of channel status data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register file in time, if a different set of data need to be sent.

Table 11-1366. MCASP_DITCSRA3 Instances

Instance	Physical Address
MCASP_0__CFG	0234 010Ch
MCASP_1__CFG	0234 210Ch
MCASP_2__CFG	0234 410Ch

Figure 11-638. MCASP_DITCSRA3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITCSRA																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1367. MCASP_DITCSRA3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITCSRA	R/W	0h	DIT left channel status registers.

Table 11-1368. Register Call Summary for MCASP_DITCSRA3

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2]
McASP Registers • McASP Registers: [0] • MCASP_DITCSRA3 Register (Offset = 10Ch) [reset = 0h]: [0]

11.9.5.41 MCASP_DITCSRA4 Register (Offset = 110h) [reset = 0h]

MCASP_DITCSRA4 is shown in Figure 11-639 and described in Table 11-1370.

The DIT left channel status registers (DITCSRA) provide the status of each left channel (even TDM time slot). Each of the six 32-bit registers can store 192 bits of channel status data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register file in time, if a different set of data need to be sent.

Table 11-1369. MCASP_DITCSRA4 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0110h
MCASP_1__CFG	0234 2110h
MCASP_2__CFG	0234 4110h

Figure 11-639. MCASP_DITCSRA4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITCSRA																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1370. MCASP_DITCSRA4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITCSRA	R/W	0h	DIT left channel status registers.

Table 11-1371. Register Call Summary for MCASP_DITCSRA4

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2]
McASP Registers • McASP Registers: [0] • MCASP_DITCSRA4 Register (Offset = 110h) [reset = 0h]: [0]

11.9.5.42 MCASP_DITCSRA5 Register (Offset = 114h) [reset = 0h]

MCASP_DITCSRA5 is shown in Figure 11-640 and described in Table 11-1373.

The DIT left channel status registers (DITCSRA) provide the status of each left channel (even TDM time slot). Each of the six 32-bit registers can store 192 bits of channel status data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register file in time, if a different set of data need to be sent.

Table 11-1372. MCASP_DITCSRA5 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0114h
MCASP_1__CFG	0234 2114h
MCASP_2__CFG	0234 4114h

Figure 11-640. MCASP_DITCSRA5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITCSRA																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1373. MCASP_DITCSRA5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITCSRA	R/W	0h	DIT left channel status registers.

Table 11-1374. Register Call Summary for MCASP_DITCSRA5

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2][3][4] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit DIT Clock and Frame-Sync Generation: [0]
McASP Registers • McASP Registers: [0] • MCASP_DITCSRA5 Register (Offset = 114h) [reset = 0h]: [0]

11.9.5.43 MCASP_DITCSRB0 Register (Offset = 118h) [reset = 0h]

MCASP_DITCSRB0 is shown in Figure 11-641 and described in Table 11-1376.

The DIT right channel status registers (MCASP_DITCSRB0) provide the status of each right channel (odd TDM time slot). Each of the six 32-bit registers can store 192 bits of channel status data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register file in time, if a different set of data need to be sent.

Table 11-1375. MCASP_DITCSRB0 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0118h
MCASP_1__CFG	0234 2118h
MCASP_2__CFG	0234 4118h

Figure 11-641. MCASP_DITCSRB0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITCSRB																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1376. MCASP_DITCSRB0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITCSRB	R/W	0h	DIT right channel status registers.

Table 11-1377. Register Call Summary for MCASP_DITCSRB0

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2][3][4][5][6] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit DIT Clock and Frame-Sync Generation: [0]
McASP Registers • MCASP_DITCSRB5 Register (Offset = 12Ch) [reset = 0h]: [0] • MCASP_DITCSRB4 Register (Offset = 128h) [reset = 0h]: [0] • MCASP_DITCSRB3 Register (Offset = 124h) [reset = 0h]: [0] • MCASP_DITCSRB2 Register (Offset = 120h) [reset = 0h]: [0] • MCASP_DITCSRB1 Register (Offset = 11Ch) [reset = 0h]: [0] • MCASP_DITCSRB0 Register (Offset = 118h) [reset = 0h]: [0][1] • McASP Registers: [0]

11.9.5.44 MCASP_DITCSRB1 Register (Offset = 11Ch) [reset = 0h]

MCASP_DITCSRB1 is shown in Figure 11-642 and described in Table 11-1379.

The DIT right channel status registers (MCASP_DITCSRB0) provide the status of each right channel (odd TDM time slot). Each of the six 32-bit registers can store 192 bits of channel status data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register file in time, if a different set of data need to be sent.

Table 11-1378. MCASP_DITCSRB1 Instances

Instance	Physical Address
MCASP_0__CFG	0234 011Ch
MCASP_1__CFG	0234 211Ch
MCASP_2__CFG	0234 411Ch

Figure 11-642. MCASP_DITCSRB1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITCSRB																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1379. MCASP_DITCSRB1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITCSRB	R/W	0h	DIT right channel status registers.

Table 11-1380. Register Call Summary for MCASP_DITCSRB1

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2]
McASP Registers • McASP Registers: [0] • MCASP_DITCSRB1 Register (Offset = 11Ch) [reset = 0h]: [0]

11.9.5.45 MCASP_DITCSRB2 Register (Offset = 120h) [reset = 0h]

MCASP_DITCSRB2 is shown in Figure 11-643 and described in Table 11-1382.

The DIT right channel status registers (MCASP_DITCSRB0) provide the status of each right channel (odd TDM time slot). Each of the six 32-bit registers can store 192 bits of channel status data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register file in time, if a different set of data need to be sent.

Table 11-1381. MCASP_DITCSRB2 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0120h
MCASP_1__CFG	0234 2120h
MCASP_2__CFG	0234 4120h

Figure 11-643. MCASP_DITCSRB2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITCSRB																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1382. MCASP_DITCSRB2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITCSRB	R/W	0h	DIT right channel status registers.

Table 11-1383. Register Call Summary for MCASP_DITCSRB2

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2]
McASP Registers • McASP Registers: [0] • MCASP_DITCSRB2 Register (Offset = 120h) [reset = 0h]: [0]

11.9.5.46 MCASP_DITCSRB3 Register (Offset = 124h) [reset = 0h]

MCASP_DITCSRB3 is shown in Figure 11-644 and described in Table 11-1385.

The DIT right channel status registers (MCASP_DITCSRB0) provide the status of each right channel (odd TDM time slot). Each of the six 32-bit registers can store 192 bits of channel status data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register file in time, if a different set of data need to be sent.

Table 11-1384. MCASP_DITCSRB3 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0124h
MCASP_1__CFG	0234 2124h
MCASP_2__CFG	0234 4124h

Figure 11-644. MCASP_DITCSRB3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITCSRB																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1385. MCASP_DITCSRB3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITCSRB	R/W	0h	DIT right channel status registers.

Table 11-1386. Register Call Summary for MCASP_DITCSRB3

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2]
McASP Registers • McASP Registers: [0] • MCASP_DITCSRB3 Register (Offset = 124h) [reset = 0h]: [0]

11.9.5.47 MCASP_DITCSRB4 Register (Offset = 128h) [reset = 0h]

MCASP_DITCSRB4 is shown in Figure 11-645 and described in Table 11-1388.

The DIT right channel status registers (MCASP_DITCSRB0) provide the status of each right channel (odd TDM time slot). Each of the six 32-bit registers can store 192 bits of channel status data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register file in time, if a different set of data need to be sent.

Table 11-1387. MCASP_DITCSRB4 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0128h
MCASP_1__CFG	0234 2128h
MCASP_2__CFG	0234 4128h

Figure 11-645. MCASP_DITCSRB4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITCSRB																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1388. MCASP_DITCSRB4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITCSRB	R/W	0h	DIT right channel status registers.

Table 11-1389. Register Call Summary for MCASP_DITCSRB4

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2]
McASP Registers • McASP Registers: [0] • MCASP_DITCSRB4 Register (Offset = 128h) [reset = 0h]: [0]

11.9.5.48 MCASP_DITCSRB5 Register (Offset = 12Ch) [reset = 0h]

MCASP_DITCSRB5 is shown in Figure 11-646 and described in Table 11-1391.

The DIT right channel status registers (MCASP_DITCSRB0) provide the status of each right channel (odd TDM time slot). Each of the six 32-bit registers can store 192 bits of channel status data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register file in time, if a different set of data need to be sent.

Table 11-1390. MCASP_DITCSRB5 Instances

Instance	Physical Address
MCASP_0__CFG	0234 012Ch
MCASP_1__CFG	0234 212Ch
MCASP_2__CFG	0234 412Ch

Figure 11-646. MCASP_DITCSRB5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITCSRB																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1391. MCASP_DITCSRB5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITCSRB	R/W	0h	DIT right channel status registers.

Table 11-1392. Register Call Summary for MCASP_DITCSRB5

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2][3][4] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit DIT Clock and Frame-Sync Generation: [0]
McASP Registers • McASP Registers: [0] • MCASP_DITCSRB5 Register (Offset = 12Ch) [reset = 0h]: [0]

11.9.5.49 MCASP_DITUDRA0 Register (Offset = 130h) [reset = 0h]

MCASP_DITUDRA0 is shown in Figure 11-647 and described in Table 11-1394.

The DIT left channel user data registers (MCASP_DITUDRA0) provides the user data of each left channel (even TDM time slot). Each of the six 32-bit registers can store 192 bits of user data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register in time, if a different set of data need to be sent.

Table 11-1393. MCASP_DITUDRA0 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0130h
MCASP_1__CFG	0234 2130h
MCASP_2__CFG	0234 4130h

Figure 11-647. MCASP_DITUDRA0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITUDRA																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1394. MCASP_DITUDRA0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITUDRA	R/W	0h	DIT left channel user data registers.

Table 11-1395. Register Call Summary for MCASP_DITUDRA0

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2][3][4][5][6] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit DIT Clock and Frame-Sync Generation: [0]
McASP Registers • MCASP_DITUDRA5 Register (Offset = 144h) [reset = 0h]: [0] • MCASP_DITUDRA4 Register (Offset = 140h) [reset = 0h]: [0] • MCASP_DITUDRA1 Register (Offset = 134h) [reset = 0h]: [0] • MCASP_DITUDRA0 Register (Offset = 130h) [reset = 0h]: [0][1] • MCASP_DITUDRA3 Register (Offset = 13Ch) [reset = 0h]: [0] • MCASP_DITUDRA2 Register (Offset = 138h) [reset = 0h]: [0] • McASP Registers: [0]

11.9.5.50 MCASP_DITUDRA1 Register (Offset = 134h) [reset = 0h]

MCASP_DITUDRA1 is shown in Figure 11-648 and described in Table 11-1397.

The DIT left channel user data registers (MCASP_DITUDRA0) provides the user data of each left channel (even TDM time slot). Each of the six 32-bit registers can store 192 bits of user data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register in time, if a different set of data need to be sent.

Table 11-1396. MCASP_DITUDRA1 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0134h
MCASP_1__CFG	0234 2134h
MCASP_2__CFG	0234 4134h

Figure 11-648. MCASP_DITUDRA1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITUDRA																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1397. MCASP_DITUDRA1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITUDRA	R/W	0h	DIT left channel user data registers.

Table 11-1398. Register Call Summary for MCASP_DITUDRA1

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2]
McASP Registers • McASP Registers: [0] • MCASP_DITUDRA1 Register (Offset = 134h) [reset = 0h]: [0]

11.9.5.51 MCASP_DITUDRA2 Register (Offset = 138h) [reset = 0h]

MCASP_DITUDRA2 is shown in Figure 11-649 and described in Table 11-1400.

The DIT left channel user data registers (MCASP_DITUDRA0) provides the user data of each left channel (even TDM time slot). Each of the six 32-bit registers can store 192 bits of user data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register in time, if a different set of data need to be sent.

Table 11-1399. MCASP_DITUDRA2 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0138h
MCASP_1__CFG	0234 2138h
MCASP_2__CFG	0234 4138h

Figure 11-649. MCASP_DITUDRA2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITUDRA																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1400. MCASP_DITUDRA2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITUDRA	R/W	0h	DIT left channel user data registers.

Table 11-1401. Register Call Summary for MCASP_DITUDRA2

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2]
McASP Registers • McASP Registers: [0] • MCASP_DITUDRA2 Register (Offset = 138h) [reset = 0h]: [0]

11.9.5.52 MCASP_DITUDRA3 Register (Offset = 13Ch) [reset = 0h]

MCASP_DITUDRA3 is shown in Figure 11-650 and described in Table 11-1403.

The DIT left channel user data registers (MCASP_DITUDRA0) provides the user data of each left channel (even TDM time slot). Each of the six 32-bit registers can store 192 bits of user data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register in time, if a different set of data need to be sent.

Table 11-1402. MCASP_DITUDRA3 Instances

Instance	Physical Address
MCASP_0__CFG	0234 013Ch
MCASP_1__CFG	0234 213Ch
MCASP_2__CFG	0234 413Ch

Figure 11-650. MCASP_DITUDRA3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITUDRA																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1403. MCASP_DITUDRA3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITUDRA	R/W	0h	DIT left channel user data registers.

Table 11-1404. Register Call Summary for MCASP_DITUDRA3

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2]
McASP Registers • McASP Registers: [0] • MCASP_DITUDRA3 Register (Offset = 13Ch) [reset = 0h]: [0]

11.9.5.53 MCASP_DITUDRA4 Register (Offset = 140h) [reset = 0h]

MCASP_DITUDRA4 is shown in Figure 11-651 and described in Table 11-1406.

The DIT left channel user data registers (MCASP_DITUDRA0) provides the user data of each left channel (even TDM time slot). Each of the six 32-bit registers can store 192 bits of user data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register in time, if a different set of data need to be sent.

Table 11-1405. MCASP_DITUDRA4 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0140h
MCASP_1__CFG	0234 2140h
MCASP_2__CFG	0234 4140h

Figure 11-651. MCASP_DITUDRA4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITUDRA																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1406. MCASP_DITUDRA4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITUDRA	R/W	0h	DIT left channel user data registers.

Table 11-1407. Register Call Summary for MCASP_DITUDRA4

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2]
McASP Registers • McASP Registers: [0] • MCASP_DITUDRA4 Register (Offset = 140h) [reset = 0h]: [0]

11.9.5.54 MCASP_DITUDRA5 Register (Offset = 144h) [reset = 0h]

MCASP_DITUDRA5 is shown in Figure 11-652 and described in Table 11-1409.

The DIT left channel user data registers (MCASP_DITUDRA0) provides the user data of each left channel (even TDM time slot). Each of the six 32-bit registers can store 192 bits of user data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register in time, if a different set of data need to be sent.

Table 11-1408. MCASP_DITUDRA5 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0144h
MCASP_1__CFG	0234 2144h
MCASP_2__CFG	0234 4144h

Figure 11-652. MCASP_DITUDRA5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITUDRA																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1409. MCASP_DITUDRA5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITUDRA	R/W	0h	DIT left channel user data registers.

Table 11-1410. Register Call Summary for MCASP_DITUDRA5

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2][3][4] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit DIT Clock and Frame-Sync Generation: [0]
McASP Registers • MCASP_DITUDRA5 Register (Offset = 144h) [reset = 0h]: [0] • McASP Registers: [0]

11.9.5.55 MCASP_DITUDRB0 Register (Offset = 148h) [reset = 0h]

MCASP_DITUDRB0 is shown in Figure 11-653 and described in Table 11-1412.

The DIT right channel user data registers (MCASP_DITUDRB0) provides the user data of each right channel (odd TDM time slot). Each of the six 32-bit registers can store 192 bits of user data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register in time, if a different set of data need to be sent.

Table 11-1411. MCASP_DITUDRB0 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0148h
MCASP_1__CFG	0234 2148h
MCASP_2__CFG	0234 4148h

Figure 11-653. MCASP_DITUDRB0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITUDRB																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1412. MCASP_DITUDRB0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITUDRB	R/W	0h	DIT right channel user data registers.

Table 11-1413. Register Call Summary for MCASP_DITUDRB0

McASP Functional Description
- DIT Channel Status and User Data Register Files: [0][1][2][3][4][5][6]
- Burst Transfer Mode: [0]
- Time-Division Multiplexed (TDM) Transfer Mode: [0]
- Transmit DIT Clock and Frame-Sync Generation: [0]

McASP Registers
- MCASP_DITUDRB5 Register (Offset = 15Ch) [reset = 0h]: [0]
- MCASP_DITUDRB4 Register (Offset = 158h) [reset = 0h]: [0]
- McASP Registers: [0]
- MCASP_DITUDRB0 Register (Offset = 148h) [reset = 0h]: [0][1]
- MCASP_DITUDRB1 Register (Offset = 14Ch) [reset = 0h]: [0]
- MCASP_DITUDRB2 Register (Offset = 150h) [reset = 0h]: [0]
- MCASP_DITUDRB3 Register (Offset = 154h) [reset = 0h]: [0]

11.9.5.56 MCASP_DITUDRB1 Register (Offset = 14Ch) [reset = 0h]

MCASP_DITUDRB1 is shown in Figure 11-654 and described in Table 11-1415.

The DIT right channel user data registers (MCASP_DITUDRB0) provides the user data of each right channel (odd TDM time slot). Each of the six 32-bit registers can store 192 bits of user data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register in time, if a different set of data need to be sent.

Table 11-1414. MCASP_DITUDRB1 Instances

Instance	Physical Address
MCASP_0__CFG	0234 014Ch
MCASP_1__CFG	0234 214Ch
MCASP_2__CFG	0234 414Ch

Figure 11-654. MCASP_DITUDRB1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITUDRB																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1415. MCASP_DITUDRB1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITUDRB	R/W	0h	DIT right channel user data registers.

Table 11-1416. Register Call Summary for MCASP_DITUDRB1

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2]
McASP Registers • McASP Registers: [0] • MCASP_DITUDRB1 Register (Offset = 14Ch) [reset = 0h]: [0]

11.9.5.57 MCASP_DITUDRB2 Register (Offset = 150h) [reset = 0h]

MCASP_DITUDRB2 is shown in Figure 11-655 and described in Table 11-1418.

The DIT right channel user data registers (MCASP_DITUDRB0) provides the user data of each right channel (odd TDM time slot). Each of the six 32-bit registers can store 192 bits of user data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register in time, if a different set of data need to be sent.

Table 11-1417. MCASP_DITUDRB2 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0150h
MCASP_1__CFG	0234 2150h
MCASP_2__CFG	0234 4150h

Figure 11-655. MCASP_DITUDRB2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITUDRB																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1418. MCASP_DITUDRB2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITUDRB	R/W	0h	DIT right channel user data registers.

Table 11-1419. Register Call Summary for MCASP_DITUDRB2

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2]
McASP Registers • McASP Registers: [0] • MCASP_DITUDRB2 Register (Offset = 150h) [reset = 0h]: [0]

11.9.5.58 MCASP_DITUDRB3 Register (Offset = 154h) [reset = 0h]

MCASP_DITUDRB3 is shown in Figure 11-656 and described in Table 11-1421.

The DIT right channel user data registers (MCASP_DITUDRB0) provides the user data of each right channel (odd TDM time slot). Each of the six 32-bit registers can store 192 bits of user data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register in time, if a different set of data need to be sent.

Table 11-1420. MCASP_DITUDRB3 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0154h
MCASP_1__CFG	0234 2154h
MCASP_2__CFG	0234 4154h

Figure 11-656. MCASP_DITUDRB3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITUDRB																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1421. MCASP_DITUDRB3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITUDRB	R/W	0h	DIT right channel user data registers.

Table 11-1422. Register Call Summary for MCASP_DITUDRB3

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2]
McASP Registers • McASP Registers: [0] • MCASP_DITUDRB3 Register (Offset = 154h) [reset = 0h]: [0]

11.9.5.59 MCASP_DITUDRB4 Register (Offset = 158h) [reset = 0h]

MCASP_DITUDRB4 is shown in Figure 11-657 and described in Table 11-1424.

The DIT right channel user data registers (MCASP_DITUDRB0) provides the user data of each right channel (odd TDM time slot). Each of the six 32-bit registers can store 192 bits of user data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register in time, if a different set of data need to be sent.

Table 11-1423. MCASP_DITUDRB4 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0158h
MCASP_1__CFG	0234 2158h
MCASP_2__CFG	0234 4158h

Figure 11-657. MCASP_DITUDRB4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITUDRB																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1424. MCASP_DITUDRB4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITUDRB	R/W	0h	DIT right channel user data registers.

Table 11-1425. Register Call Summary for MCASP_DITUDRB4

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2]
McASP Registers • MCASP_DITUDRB4 Register (Offset = 158h) [reset = 0h]: [0] • McASP Registers: [0]

11.9.5.60 MCASP_DITUDRB5 Register (Offset = 15Ch) [reset = 0h]

MCASP_DITUDRB5 is shown in Figure 11-658 and described in Table 11-1427.

The DIT right channel user data registers (MCASP_DITUDRB0) provides the user data of each right channel (odd TDM time slot). Each of the six 32-bit registers can store 192 bits of user data for a complete block of transmission. The DIT reuses the same data for the next block. It is SW responsibility to update the register in time, if a different set of data need to be sent.

Table 11-1426. MCASP_DITUDRB5 Instances

Instance	Physical Address
MCASP_0__CFG	0234 015Ch
MCASP_1__CFG	0234 215Ch
MCASP_2__CFG	0234 415Ch

Figure 11-658. MCASP_DITUDRB5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DITUDRB																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1427. MCASP_DITUDRB5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DITUDRB	R/W	0h	DIT right channel user data registers.

Table 11-1428. Register Call Summary for MCASP_DITUDRB5

McASP Functional Description • DIT Channel Status and User Data Register Files: [0][1][2][3][4] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit DIT Clock and Frame-Sync Generation: [0]
McASP Registers • McASP Registers: [0] • MCASP_DITUDRB5 Register (Offset = 15Ch) [reset = 0h]: [0]

11.9.5.61 MCASP_SRCTL0 Register (Offset = 180h) [reset = 0h]

MCASP_SRCTL0 is shown in Figure 11-659 and described in Table 11-1430.

Each serializer on the McASP has a serializer control register (MCASP_SRCTL0). There are up to 16 serializers per McASP. Note for device-specific registers: Accessing SRCTL0n not implemented on a specific device may cause improper device operation.

Table 11-1429. MCASP_SRCTL0 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0180h
MCASP_1__CFG	0234 2180h
MCASP_2__CFG	0234 4180h

Figure 11-659. MCASP_SRCTL0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RRDY	XRDY	DISMOD		SRMOD	
R-0h		R-0h	R-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1430. MCASP_SRCTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RRDY	R	0h	Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to MCASP_RBUF. 0h (R/W) = Receive buffer (MCASP_RBUF) is empty. 1h (R/W) = Receive buffer (MCASP_RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY	R	0h	Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in MCASP_GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0). 0h (R/W) = Transmit buffer (MCASP_XBUF) contains data. 1h (R/W) = Transmit buffer (MCASP_XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.

Table 11-1430. MCASP_SRCTL0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	DISMOD	R/W	0h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (MCASP_PFUNC = 0). 0h (R/W) = Drive on pin is 3-state. 1h (R/W) = Reserved. 2h (R/W) = Drive on pin is logic low. 3h (R/W) = Drive on pin is logic high.
1-0	SRMOD	R/W	0h	Serializer mode bit. 0h (R/W) = Serializer is inactive. 1h (R/W) = Serializer is transmitter. 2h (R/W) = Serializer is receiver. 3h (R/W) = Reserved.

Table 11-1431. Register Call Summary for MCASP_SRCTL0

McASP Functional Description • TDM Time Slots Generation and Processing: [0] • McASP Events and Interrupt Requests: [0] • McASP Serializers: [0][1][2][3][4][5][6][7] • Transmit Data Ready: [0][1] • Transfers Through the Data Port (DATA): [0][1] • McASP TDM Sequencers: [0] • Receive Data Ready: [0][1] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit DIT Clock and Frame-Sync Generation: [0]
McASP Registers • MCASP_SRCTL13 Register (Offset = 1B4h) [reset = 0h]: [0] • MCASP_SRCTL12 Register (Offset = 1B0h) [reset = 0h]: [0] • McASP Registers: [0] • MCASP_SRCTL10 Register (Offset = 1A8h) [reset = 0h]: [0] • MCASP_SRCTL11 Register (Offset = 1ACh) [reset = 0h]: [0] • MCASP_SRCTL14 Register (Offset = 1B8h) [reset = 0h]: [0] • MCASP_SRCTL15 Register (Offset = 1BCh) [reset = 0h]: [0] • MCASP_SRCTL0 Register (Offset = 180h) [reset = 0h]: [0][1] • MCASP_SRCTL1 Register (Offset = 184h) [reset = 0h]: [0] • MCASP_SRCTL2 Register (Offset = 188h) [reset = 0h]: [0] • MCASP_SRCTL3 Register (Offset = 18Ch) [reset = 0h]: [0] • MCASP_SRCTL4 Register (Offset = 190h) [reset = 0h]: [0] • MCASP_SRCTL5 Register (Offset = 194h) [reset = 0h]: [0] • MCASP_SRCTL6 Register (Offset = 198h) [reset = 0h]: [0] • MCASP_SRCTL7 Register (Offset = 19Ch) [reset = 0h]: [0] • MCASP_SRCTL8 Register (Offset = 1A0h) [reset = 0h]: [0] • MCASP_SRCTL9 Register (Offset = 1A4h) [reset = 0h]: [0]
McASP Environment • McASP Signals: [0]

11.9.5.62 MCASP_SRCTL1 Register (Offset = 184h) [reset = 0h]

MCASP_SRCTL1 is shown in Figure 11-660 and described in Table 11-1433.

Each serializer on the McASP has a serializer control register (MCASP_SRCTL0). There are up to 16 serializers per McASP. Note for device-specific registers: Accessing SRCTL0n not implemented on a specific device may cause improper device operation.

Table 11-1432. MCASP_SRCTL1 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0184h
MCASP_1__CFG	0234 2184h
MCASP_2__CFG	0234 4184h

Figure 11-660. MCASP_SRCTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RRDY	XRDY	DISMOD		SRMOD	
R-0h		R-0h	R-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1433. MCASP_SRCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RRDY	R	0h	Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to MCASP_RBUF. 0h (R/W) = Receive buffer (MCASP_RBUF) is empty. 1h (R/W) = Receive buffer (MCASP_RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY	R	0h	Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in MCASP_GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0). 0h (R/W) = Transmit buffer (MCASP_XBUF) contains data. 1h (R/W) = Transmit buffer (MCASP_XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.

Table 11-1433. MCASP_SRCTL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	DISMOD	R/W	0h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (MCASP_PFUNC = 0). 0h (R/W) = Drive on pin is 3-state. 1h (R/W) = Reserved. 2h (R/W) = Drive on pin is logic low. 3h (R/W) = Drive on pin is logic high.
1-0	SRMOD	R/W	0h	Serializer mode bit. 0h (R/W) = Serializer is inactive. 1h (R/W) = Serializer is transmitter. 2h (R/W) = Serializer is receiver. 3h (R/W) = Reserved.

Table 11-1434. Register Call Summary for MCASP_SRCTL1

McASP Registers
- McASP Registers: [0]
- MCASP_SRCTL1 Register (Offset = 184h) [reset = 0h]: [0]

11.9.5.63 MCASP_SRCTL2 Register (Offset = 188h) [reset = 0h]

MCASP_SRCTL2 is shown in Figure 11-661 and described in Table 11-1436.

Each serializer on the McASP has a serializer control register (MCASP_SRCTL0). There are up to 16 serializers per McASP. Note for device-specific registers: Accessing SRCTL0n not implemented on a specific device may cause improper device operation.

Table 11-1435. MCASP_SRCTL2 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0188h
MCASP_1__CFG	0234 2188h
MCASP_2__CFG	0234 4188h

Figure 11-661. MCASP_SRCTL2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RRDY	XRDY	DISMOD		SRMOD	
R-0h		R-0h	R-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1436. MCASP_SRCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RRDY	R	0h	Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to MCASP_RBUF. 0h (R/W) = Receive buffer (MCASP_RBUF) is empty. 1h (R/W) = Receive buffer (MCASP_RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY	R	0h	Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in MCASP_GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0). 0h (R/W) = Transmit buffer (MCASP_XBUF) contains data. 1h (R/W) = Transmit buffer (MCASP_XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.

Table 11-1436. MCASP_SRCTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	DISMOD	R/W	0h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (MCASP_PFUNC = 0). 0h (R/W) = Drive on pin is 3-state. 1h (R/W) = Reserved. 2h (R/W) = Drive on pin is logic low. 3h (R/W) = Drive on pin is logic high.
1-0	SRMOD	R/W	0h	Serializer mode bit. 0h (R/W) = Serializer is inactive. 1h (R/W) = Serializer is transmitter. 2h (R/W) = Serializer is receiver. 3h (R/W) = Reserved.

Table 11-1437. Register Call Summary for MCASP_SRCTL2

McASP Registers
- McASP Registers: [0]
- MCASP_SRCTL2 Register (Offset = 188h) [reset = 0h]: [0]

11.9.5.64 MCASP_SRCTL3 Register (Offset = 18Ch) [reset = 0h]

MCASP_SRCTL3 is shown in Figure 11-662 and described in Table 11-1439.

Each serializer on the McASP has a serializer control register (MCASP_SRCTL0). There are up to 16 serializers per McASP. Note for device-specific registers: Accessing SRCTL0n not implemented on a specific device may cause improper device operation.

Table 11-1438. MCASP_SRCTL3 Instances

Instance	Physical Address
MCASP_0__CFG	0234 018Ch
MCASP_1__CFG	0234 218Ch
MCASP_2__CFG	0234 418Ch

Figure 11-662. MCASP_SRCTL3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RRDY	XRDY	DISMOD		SRMOD	
R-0h		R-0h	R-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1439. MCASP_SRCTL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RRDY	R	0h	Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to MCASP_RBUF. 0h (R/W) = Receive buffer (MCASP_RBUF) is empty. 1h (R/W) = Receive buffer (MCASP_RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY	R	0h	Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in MCASP_GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0). 0h (R/W) = Transmit buffer (MCASP_XBUF) contains data. 1h (R/W) = Transmit buffer (MCASP_XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.

Table 11-1439. MCASP_SRCTL3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	DISMOD	R/W	0h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (MCASP_PFUNC = 0). 0h (R/W) = Drive on pin is 3-state. 1h (R/W) = Reserved. 2h (R/W) = Drive on pin is logic low. 3h (R/W) = Drive on pin is logic high.
1-0	SRMOD	R/W	0h	Serializer mode bit. 0h (R/W) = Serializer is inactive. 1h (R/W) = Serializer is transmitter. 2h (R/W) = Serializer is receiver. 3h (R/W) = Reserved.

Table 11-1440. Register Call Summary for MCASP_SRCTL3

McASP Registers
- McASP Registers: [0]
- MCASP_SRCTL3 Register (Offset = 18Ch) [reset = 0h]: [0]

11.9.5.65 MCASP_SRCTL4 Register (Offset = 190h) [reset = 0h]

MCASP_SRCTL4 is shown in Figure 11-663 and described in Table 11-1442.

Each serializer on the McASP has a serializer control register (MCASP_SRCTL0). There are up to 16 serializers per McASP. Note for device-specific registers: Accessing SRCTL0n not implemented on a specific device may cause improper device operation.

Table 11-1441. MCASP_SRCTL4 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0190h
MCASP_1__CFG	0234 2190h
MCASP_2__CFG	0234 4190h

Figure 11-663. MCASP_SRCTL4 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RRDY	XRDY	DISMOD		SRMOD	
R-0h		R-0h	R-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1442. MCASP_SRCTL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RRDY	R	0h	Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to MCASP_RBUF. 0h (R/W) = Receive buffer (MCASP_RBUF) is empty. 1h (R/W) = Receive buffer (MCASP_RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY	R	0h	Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in MCASP_GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0). 0h (R/W) = Transmit buffer (MCASP_XBUF) contains data. 1h (R/W) = Transmit buffer (MCASP_XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.

Table 11-1442. MCASP_SRCTL4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	DISMOD	R/W	0h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (MCASP_PFUNC = 0). 0h (R/W) = Drive on pin is 3-state. 1h (R/W) = Reserved. 2h (R/W) = Drive on pin is logic low. 3h (R/W) = Drive on pin is logic high.
1-0	SRMOD	R/W	0h	Serializer mode bit. 0h (R/W) = Serializer is inactive. 1h (R/W) = Serializer is transmitter. 2h (R/W) = Serializer is receiver. 3h (R/W) = Reserved.

Table 11-1443. Register Call Summary for MCASP_SRCTL4

McASP Registers
- McASP Registers: [0]
- MCASP_SRCTL4 Register (Offset = 190h) [reset = 0h]: [0]

11.9.5.66 MCASP_SRCTL5 Register (Offset = 194h) [reset = 0h]

MCASP_SRCTL5 is shown in Figure 11-664 and described in Table 11-1445.

Each serializer on the McASP has a serializer control register (MCASP_SRCTL0). There are up to 16 serializers per McASP. Note for device-specific registers: Accessing SRCTL0n not implemented on a specific device may cause improper device operation.

Table 11-1444. MCASP_SRCTL5 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0194h
MCASP_1__CFG	0234 2194h
MCASP_2__CFG	0234 4194h

Figure 11-664. MCASP_SRCTL5 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RRDY	XRDY	DISMOD		SRMOD	
R-0h		R-0h	R-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1445. MCASP_SRCTL5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RRDY	R	0h	Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to MCASP_RBUF. 0h (R/W) = Receive buffer (MCASP_RBUF) is empty. 1h (R/W) = Receive buffer (MCASP_RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY	R	0h	Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in MCASP_GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0). 0h (R/W) = Transmit buffer (MCASP_XBUF) contains data. 1h (R/W) = Transmit buffer (MCASP_XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.

Table 11-1445. MCASP_SRCTL5 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	DISMOD	R/W	0h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (MCASP_PFUNC = 0). 0h (R/W) = Drive on pin is 3-state. 1h (R/W) = Reserved. 2h (R/W) = Drive on pin is logic low. 3h (R/W) = Drive on pin is logic high.
1-0	SRMOD	R/W	0h	Serializer mode bit. 0h (R/W) = Serializer is inactive. 1h (R/W) = Serializer is transmitter. 2h (R/W) = Serializer is receiver. 3h (R/W) = Reserved.

Table 11-1446. Register Call Summary for MCASP_SRCTL5

McASP Registers
- McASP Registers: [0]
- MCASP_SRCTL5 Register (Offset = 194h) [reset = 0h]: [0]

11.9.5.67 MCASP_SRCTL6 Register (Offset = 198h) [reset = 0h]

MCASP_SRCTL6 is shown in Figure 11-665 and described in Table 11-1448.

Each serializer on the McASP has a serializer control register (MCASP_SRCTL0). There are up to 16 serializers per McASP. Note for device-specific registers: Accessing SRCTL0n not implemented on a specific device may cause improper device operation.

Table 11-1447. MCASP_SRCTL6 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0198h
MCASP_1__CFG	0234 2198h
MCASP_2__CFG	0234 4198h

Figure 11-665. MCASP_SRCTL6 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RRDY	XRDY	DISMOD		SRMOD	
R-0h		R-0h	R-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1448. MCASP_SRCTL6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RRDY	R	0h	Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to MCASP_RBUF. 0h (R/W) = Receive buffer (MCASP_RBUF) is empty. 1h (R/W) = Receive buffer (MCASP_RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY	R	0h	Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in MCASP_GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0). 0h (R/W) = Transmit buffer (MCASP_XBUF) contains data. 1h (R/W) = Transmit buffer (MCASP_XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.

Table 11-1448. MCASP_SRCTL6 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	DISMOD	R/W	0h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (MCASP_PFUNC = 0). 0h (R/W) = Drive on pin is 3-state. 1h (R/W) = Reserved. 2h (R/W) = Drive on pin is logic low. 3h (R/W) = Drive on pin is logic high.
1-0	SRMOD	R/W	0h	Serializer mode bit. 0h (R/W) = Serializer is inactive. 1h (R/W) = Serializer is transmitter. 2h (R/W) = Serializer is receiver. 3h (R/W) = Reserved.

Table 11-1449. Register Call Summary for MCASP_SRCTL6

McASP Registers
- McASP Registers: [0]
- MCASP_SRCTL6 Register (Offset = 198h) [reset = 0h]: [0]

11.9.5.68 MCASP_SRCTL7 Register (Offset = 19Ch) [reset = 0h]

MCASP_SRCTL7 is shown in Figure 11-666 and described in Table 11-1451.

Each serializer on the McASP has a serializer control register (MCASP_SRCTL0). There are up to 16 serializers per McASP. Note for device-specific registers: Accessing SRCTL0n not implemented on a specific device may cause improper device operation.

Table 11-1450. MCASP_SRCTL7 Instances

Instance	Physical Address
MCASP_0__CFG	0234 019Ch
MCASP_1__CFG	0234 219Ch
MCASP_2__CFG	0234 419Ch

Figure 11-666. MCASP_SRCTL7 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RRDY	XRDY	DISMOD		SRMOD	
R-0h		R-0h	R-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1451. MCASP_SRCTL7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RRDY	R	0h	Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to MCASP_RBUF. 0h (R/W) = Receive buffer (MCASP_RBUF) is empty. 1h (R/W) = Receive buffer (MCASP_RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY	R	0h	Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in MCASP_GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0). 0h (R/W) = Transmit buffer (MCASP_XBUF) contains data. 1h (R/W) = Transmit buffer (MCASP_XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.

Table 11-1451. MCASP_SRCTL7 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	DISMOD	R/W	0h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (MCASP_PFUNC = 0). 0h (R/W) = Drive on pin is 3-state. 1h (R/W) = Reserved. 2h (R/W) = Drive on pin is logic low. 3h (R/W) = Drive on pin is logic high.
1-0	SRMOD	R/W	0h	Serializer mode bit. 0h (R/W) = Serializer is inactive. 1h (R/W) = Serializer is transmitter. 2h (R/W) = Serializer is receiver. 3h (R/W) = Reserved.

Table 11-1452. Register Call Summary for MCASP_SRCTL7

McASP Registers
- McASP Registers: [0]
- MCASP_SRCTL7 Register (Offset = 19Ch) [reset = 0h]: [0]

11.9.5.69 MCASP_SRCTL8 Register (Offset = 1A0h) [reset = 0h]

MCASP_SRCTL8 is shown in Figure 11-667 and described in Table 11-1454.

Each serializer on the McASP has a serializer control register (MCASP_SRCTL0). There are up to 16 serializers per McASP. Note for device-specific registers: Accessing SRCTL0n not implemented on a specific device may cause improper device operation.

Table 11-1453. MCASP_SRCTL8 Instances

Instance	Physical Address
MCASP_0__CFG	0234 01A0h
MCASP_1__CFG	0234 21A0h
MCASP_2__CFG	0234 41A0h

Figure 11-667. MCASP_SRCTL8 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RRDY	XRDY	DISMOD		SRMOD	
R-0h		R-0h	R-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1454. MCASP_SRCTL8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RRDY	R	0h	Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to MCASP_RBUF. 0h (R/W) = Receive buffer (MCASP_RBUF) is empty. 1h (R/W) = Receive buffer (MCASP_RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY	R	0h	Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in MCASP_GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0). 0h (R/W) = Transmit buffer (MCASP_XBUF) contains data. 1h (R/W) = Transmit buffer (MCASP_XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.

Table 11-1454. MCASP_SRCTL8 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	DISMOD	R/W	0h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (MCASP_PFUNC = 0). 0h (R/W) = Drive on pin is 3-state. 1h (R/W) = Reserved. 2h (R/W) = Drive on pin is logic low. 3h (R/W) = Drive on pin is logic high.
1-0	SRMOD	R/W	0h	Serializer mode bit. 0h (R/W) = Serializer is inactive. 1h (R/W) = Serializer is transmitter. 2h (R/W) = Serializer is receiver. 3h (R/W) = Reserved.

Table 11-1455. Register Call Summary for MCASP_SRCTL8

McASP Registers
- MCASP_SRCTL8 Register (Offset = 1A0h) [reset = 0h]: [0]
- McASP Registers: [0]

11.9.5.70 MCASP_SRCTL9 Register (Offset = 1A4h) [reset = 0h]

MCASP_SRCTL9 is shown in Figure 11-668 and described in Table 11-1457.

Each serializer on the McASP has a serializer control register (MCASP_SRCTL0). There are up to 16 serializers per McASP. Note for device-specific registers: Accessing SRCTL0n not implemented on a specific device may cause improper device operation.

Table 11-1456. MCASP_SRCTL9 Instances

Instance	Physical Address
MCASP_0__CFG	0234 01A4h
MCASP_1__CFG	0234 21A4h
MCASP_2__CFG	0234 41A4h

Figure 11-668. MCASP_SRCTL9 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RRDY	XRDY	DISMOD		SRMOD	
R-0h		R-0h	R-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1457. MCASP_SRCTL9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RRDY	R	0h	Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to MCASP_RBUF. 0h (R/W) = Receive buffer (MCASP_RBUF) is empty. 1h (R/W) = Receive buffer (MCASP_RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY	R	0h	Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in MCASP_GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0). 0h (R/W) = Transmit buffer (MCASP_XBUF) contains data. 1h (R/W) = Transmit buffer (MCASP_XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.

Table 11-1457. MCASP_SRCTL9 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	DISMOD	R/W	0h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (MCASP_PFUNC = 0). 0h (R/W) = Drive on pin is 3-state. 1h (R/W) = Reserved. 2h (R/W) = Drive on pin is logic low. 3h (R/W) = Drive on pin is logic high.
1-0	SRMOD	R/W	0h	Serializer mode bit. 0h (R/W) = Serializer is inactive. 1h (R/W) = Serializer is transmitter. 2h (R/W) = Serializer is receiver. 3h (R/W) = Reserved.

Table 11-1458. Register Call Summary for MCASP_SRCTL9

McASP Registers
- McASP Registers: [0]
- MCASP_SRCTL9 Register (Offset = 1A4h) [reset = 0h]: [0]

11.9.5.71 MCASP_SRCTL10 Register (Offset = 1A8h) [reset = 0h]

MCASP_SRCTL10 is shown in Figure 11-669 and described in Table 11-1460.

Each serializer on the McASP has a serializer control register (MCASP_SRCTL0). There are up to 16 serializers per McASP. Note for device-specific registers: Accessing SRCTL0n not implemented on a specific device may cause improper device operation.

Table 11-1459. MCASP_SRCTL10 Instances

Instance	Physical Address
MCASP_0__CFG	0234 01A8h
MCASP_1__CFG	0234 21A8h
MCASP_2__CFG	0234 41A8h

Figure 11-669. MCASP_SRCTL10 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RRDY	XRDY	DISMOD		SRMOD	
R-0h		R-0h	R-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1460. MCASP_SRCTL10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RRDY	R	0h	Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to MCASP_RBUF. 0h (R/W) = Receive buffer (MCASP_RBUF) is empty. 1h (R/W) = Receive buffer (MCASP_RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY	R	0h	Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in MCASP_GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0). 0h (R/W) = Transmit buffer (MCASP_XBUF) contains data. 1h (R/W) = Transmit buffer (MCASP_XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.

Table 11-1460. MCASP_SRCTL10 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	DISMOD	R/W	0h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (MCASP_PFUNC = 0). 0h (R/W) = Drive on pin is 3-state. 1h (R/W) = Reserved. 2h (R/W) = Drive on pin is logic low. 3h (R/W) = Drive on pin is logic high.
1-0	SRMOD	R/W	0h	Serializer mode bit. 0h (R/W) = Serializer is inactive. 1h (R/W) = Serializer is transmitter. 2h (R/W) = Serializer is receiver. 3h (R/W) = Reserved.

Table 11-1461. Register Call Summary for MCASP_SRCTL10

McASP Registers
- McASP Registers: [0]
- MCASP_SRCTL10 Register (Offset = 1A8h) [reset = 0h]: [0]

11.9.5.72 MCASP_SRCTL11 Register (Offset = 1ACh) [reset = 0h]

MCASP_SRCTL11 is shown in Figure 11-670 and described in Table 11-1463.

Each serializer on the McASP has a serializer control register (MCASP_SRCTL0). There are up to 16 serializers per McASP. Note for device-specific registers: Accessing SRCTL0n not implemented on a specific device may cause improper device operation.

Table 11-1462. MCASP_SRCTL11 Instances

Instance	Physical Address
MCASP_0__CFG	0234 01ACh
MCASP_1__CFG	0234 21ACh
MCASP_2__CFG	0234 41ACh

Figure 11-670. MCASP_SRCTL11 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RRDY	XRDY	DISMOD		SRMOD	
R-0h		R-0h	R-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1463. MCASP_SRCTL11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RRDY	R	0h	Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to MCASP_RBUF. 0h (R/W) = Receive buffer (MCASP_RBUF) is empty. 1h (R/W) = Receive buffer (MCASP_RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY	R	0h	Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in MCASP_GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0). 0h (R/W) = Transmit buffer (MCASP_XBUF) contains data. 1h (R/W) = Transmit buffer (MCASP_XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.

Table 11-1463. MCASP_SRCTL11 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	DISMOD	R/W	0h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (MCASP_PFUNC = 0). 0h (R/W) = Drive on pin is 3-state. 1h (R/W) = Reserved. 2h (R/W) = Drive on pin is logic low. 3h (R/W) = Drive on pin is logic high.
1-0	SRMOD	R/W	0h	Serializer mode bit. 0h (R/W) = Serializer is inactive. 1h (R/W) = Serializer is transmitter. 2h (R/W) = Serializer is receiver. 3h (R/W) = Reserved.

Table 11-1464. Register Call Summary for MCASP_SRCTL11

McASP Registers
- McASP Registers: [0]
- MCASP_SRCTL11 Register (Offset = 1ACh) [reset = 0h]: [0]

11.9.5.73 MCASP_SRCTL12 Register (Offset = 1B0h) [reset = 0h]

MCASP_SRCTL12 is shown in Figure 11-671 and described in Table 11-1466.

Each serializer on the McASP has a serializer control register (MCASP_SRCTL0). There are up to 16 serializers per McASP. Note for device-specific registers: Accessing SRCTL0n not implemented on a specific device may cause improper device operation.

Table 11-1465. MCASP_SRCTL12 Instances

Instance	Physical Address
MCASP_0__CFG	0234 01B0h
MCASP_1__CFG	0234 21B0h
MCASP_2__CFG	0234 41B0h

Figure 11-671. MCASP_SRCTL12 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RRDY	XRDY	DISMOD		SRMOD	
R-0h		R-0h	R-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1466. MCASP_SRCTL12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RRDY	R	0h	Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to MCASP_RBUF. 0h (R/W) = Receive buffer (MCASP_RBUF) is empty. 1h (R/W) = Receive buffer (MCASP_RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY	R	0h	Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in MCASP_GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0). 0h (R/W) = Transmit buffer (MCASP_XBUF) contains data. 1h (R/W) = Transmit buffer (MCASP_XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.

Table 11-1466. MCASP_SRCTL12 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	DISMOD	R/W	0h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (MCASP_PFUNC = 0). 0h (R/W) = Drive on pin is 3-state. 1h (R/W) = Reserved. 2h (R/W) = Drive on pin is logic low. 3h (R/W) = Drive on pin is logic high.
1-0	SRMOD	R/W	0h	Serializer mode bit. 0h (R/W) = Serializer is inactive. 1h (R/W) = Serializer is transmitter. 2h (R/W) = Serializer is receiver. 3h (R/W) = Reserved.

Table 11-1467. Register Call Summary for MCASP_SRCTL12

McASP Registers
- MCASP_SRCTL12 Register (Offset = 1B0h) [reset = 0h]: [0]
- McASP Registers: [0]

11.9.5.74 MCASP_SRCTL13 Register (Offset = 1B4h) [reset = 0h]

MCASP_SRCTL13 is shown in Figure 11-672 and described in Table 11-1469.

Each serializer on the McASP has a serializer control register (MCASP_SRCTL0). There are up to 16 serializers per McASP. Note for device-specific registers: Accessing SRCTL0n not implemented on a specific device may cause improper device operation.

Table 11-1468. MCASP_SRCTL13 Instances

Instance	Physical Address
MCASP_0__CFG	0234 01B4h
MCASP_1__CFG	0234 21B4h
MCASP_2__CFG	0234 41B4h

Figure 11-672. MCASP_SRCTL13 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RRDY	XRDY	DISMOD		SRMOD	
R-0h		R-0h	R-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1469. MCASP_SRCTL13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RRDY	R	0h	Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to MCASP_RBUF. 0h (R/W) = Receive buffer (MCASP_RBUF) is empty. 1h (R/W) = Receive buffer (MCASP_RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY	R	0h	Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in MCASP_GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0). 0h (R/W) = Transmit buffer (MCASP_XBUF) contains data. 1h (R/W) = Transmit buffer (MCASP_XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.

Table 11-1469. MCASP_SRCTL13 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	DISMOD	R/W	0h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (MCASP_PFUNC = 0). 0h (R/W) = Drive on pin is 3-state. 1h (R/W) = Reserved. 2h (R/W) = Drive on pin is logic low. 3h (R/W) = Drive on pin is logic high.
1-0	SRMOD	R/W	0h	Serializer mode bit. 0h (R/W) = Serializer is inactive. 1h (R/W) = Serializer is transmitter. 2h (R/W) = Serializer is receiver. 3h (R/W) = Reserved.

Table 11-1470. Register Call Summary for MCASP_SRCTL13

McASP Registers
- McASP Registers: [0]
- MCASP_SRCTL13 Register (Offset = 1B4h) [reset = 0h]: [0]

11.9.5.75 MCASP_SRCTL14 Register (Offset = 1B8h) [reset = 0h]

MCASP_SRCTL14 is shown in Figure 11-673 and described in Table 11-1472.

Each serializer on the McASP has a serializer control register (MCASP_SRCTL0). There are up to 16 serializers per McASP. Note for device-specific registers: Accessing SRCTL0n not implemented on a specific device may cause improper device operation.

Table 11-1471. MCASP_SRCTL14 Instances

Instance	Physical Address
MCASP_0__CFG	0234 01B8h
MCASP_1__CFG	0234 21B8h
MCASP_2__CFG	0234 41B8h

Figure 11-673. MCASP_SRCTL14 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RRDY	XRDY	DISMOD		SRMOD	
R-0h		R-0h	R-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1472. MCASP_SRCTL14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RRDY	R	0h	Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to MCASP_RBUF. 0h (R/W) = Receive buffer (MCASP_RBUF) is empty. 1h (R/W) = Receive buffer (MCASP_RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY	R	0h	Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in MCASP_GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0). 0h (R/W) = Transmit buffer (MCASP_XBUF) contains data. 1h (R/W) = Transmit buffer (MCASP_XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.

Table 11-1472. MCASP_SRCTL14 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	DISMOD	R/W	0h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (MCASP_PFUNC = 0). 0h (R/W) = Drive on pin is 3-state. 1h (R/W) = Reserved. 2h (R/W) = Drive on pin is logic low. 3h (R/W) = Drive on pin is logic high.
1-0	SRMOD	R/W	0h	Serializer mode bit. 0h (R/W) = Serializer is inactive. 1h (R/W) = Serializer is transmitter. 2h (R/W) = Serializer is receiver. 3h (R/W) = Reserved.

Table 11-1473. Register Call Summary for MCASP_SRCTL14

McASP Registers
- McASP Registers: [0]
- MCASP_SRCTL14 Register (Offset = 1B8h) [reset = 0h]: [0]

11.9.5.76 MCASP_SRCTL15 Register (Offset = 1BCh) [reset = 0h]

MCASP_SRCTL15 is shown in Figure 11-674 and described in Table 11-1475.

Each serializer on the McASP has a serializer control register (MCASP_SRCTL0). There are up to 16 serializers per McASP. Note for device-specific registers: Accessing SRCTL0n not implemented on a specific device may cause improper device operation.

Table 11-1474. MCASP_SRCTL15 Instances

Instance	Physical Address
MCASP_0__CFG	0234 01BCh
MCASP_1__CFG	0234 21BCh
MCASP_2__CFG	0234 41BCh

Figure 11-674. MCASP_SRCTL15 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RRDY	XRDY	DISMOD		SRMOD	
R-0h		R-0h	R-0h	R/W-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1475. MCASP_SRCTL15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RRDY	R	0h	Receive buffer ready bit. RRDY indicates the current receive buffer state. Always reads 0 when programmed as a transmitter or as inactive. If SRMOD bit is set to receive (2h), RRDY switches from 0 to 1 whenever data is transferred from XRSR to MCASP_RBUF. 0h (R/W) = Receive buffer (MCASP_RBUF) is empty. 1h (R/W) = Receive buffer (MCASP_RBUF) contains data and needs to be read before the start of the next time slot or a receiver overrun occurs.
4	XRDY	R	0h	Transmit buffer ready bit. XRDY indicates the current transmit buffer state. Always reads 0 when programmed as a receiver or as inactive. If SRMOD bit is set to transmit (1h), XRDY switches from 0 to 1 when XSRCLR in MCASP_GBLCTL is switched from 0 to 1 to indicate an empty transmitter. XRDY remains set until XSRCLR is forced to 0, data is written to the corresponding transmit buffer, or SRMOD bit is changed to receive (2h) or inactive (0). 0h (R/W) = Transmit buffer (MCASP_XBUF) contains data. 1h (R/W) = Transmit buffer (MCASP_XBUF) is empty and needs to be written before the start of the next time slot or a transmit underrun occurs.

Table 11-1475. MCASP_SRCTL15 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	DISMOD	R/W	0h	Serializer pin drive mode bit. Drive on pin when in inactive TDM slot of transmit mode or when serializer is inactive. This field only applies if the pin is configured as a McASP pin (MCASP_PFUNC = 0). 0h (R/W) = Drive on pin is 3-state. 1h (R/W) = Reserved. 2h (R/W) = Drive on pin is logic low. 3h (R/W) = Drive on pin is logic high.
1-0	SRMOD	R/W	0h	Serializer mode bit. 0h (R/W) = Serializer is inactive. 1h (R/W) = Serializer is transmitter. 2h (R/W) = Serializer is receiver. 3h (R/W) = Reserved.

Table 11-1476. Register Call Summary for MCASP_SRCTL15

McASP Functional Description • TDM Time Slots Generation and Processing: [0] • McASP Events and Interrupt Requests: [0] • McASP Serializers: [0][1][2][3][4][5][6][7] • Transmit Data Ready: [0][1] • Transfers Through the Data Port (DATA): [0][1] • McASP TDM Sequencers: [0] • Receive Data Ready: [0][1] • Burst Transfer Mode: [0] • Time-Division Multiplexed (TDM) Transfer Mode: [0] • Transmit DIT Clock and Frame-Sync Generation: [0]
McASP Registers • McASP Registers: [0] • MCASP_SRCTL15 Register (Offset = 1BCh) [reset = 0h]: [0]
McASP Environment • McASP Signals: [0]

11.9.5.77 MCASP_XBUF0 Register (Offset = 200h) [reset = 0h]

MCASP_XBUF0 is shown in Figure 11-675 and described in Table 11-1478.

The transmit buffers for the serializers (MCASP_XBUF0) hold data from the transmit format unit. For transmit operations, the MCASP_XBUF0 is an alias of the XRBUF in the serializer. Accessing MCASP_XBUF0 registers not implemented on a specific device may cause improper device operation.

Table 11-1477. MCASP_XBUF0 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0200h
MCASP_1__CFG	0234 2200h
MCASP_2__CFG	0234 4200h

Figure 11-675. MCASP_XBUF0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1478. MCASP_XBUF0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	R/W	0h	Transmit buffers for serializers.

Table 11-1479. Register Call Summary for MCASP_XBUF0

McASP Functional Description
- Buffer Underrun Error-Transmitter: [0]
- McASP Serializers: [0][1]
- Transmit Data Ready: [0][1][2][3][4][5][6]
- Transfers Through the Configuration Bus (CFG): [0][1][2]
- McASP TDM Sequencers: [0]
- McASP Data Transmission and Reception: [0]

McASP Registers
- MCASP_XBUF0 Register (Offset = 200h) [reset = 0h]: [0][1][2][3]
- MCASP_XBUF1 Register (Offset = 204h) [reset = 0h]: [0][1][2]
- MCASP_XBUF2 Register (Offset = 208h) [reset = 0h]: [0][1][2]
- MCASP_XBUF3 Register (Offset = 20Ch) [reset = 0h]: [0][1][2]
- MCASP_XBUF4 Register (Offset = 210h) [reset = 0h]: [0][1][2]
- MCASP_XBUF5 Register (Offset = 214h) [reset = 0h]: [0][1][2]
- MCASP_XBUF6 Register (Offset = 218h) [reset = 0h]: [0][1][2]
- MCASP_XBUF7 Register (Offset = 21Ch) [reset = 0h]: [0][1][2]
- MCASP_XBUF8 Register (Offset = 220h) [reset = 0h]: [0][1][2]
- McASP Registers: [0]
- MCASP_XBUF14 Register (Offset = 238h) [reset = 0h]: [0][1][2]
- MCASP_XBUF10 Register (Offset = 228h) [reset = 0h]: [0][1][2]
- MCASP_XBUF11 Register (Offset = 22Ch) [reset = 0h]: [0][1][2]
- MCASP_XBUF13 Register (Offset = 234h) [reset = 0h]: [0][1][2]
- MCASP_XBUF9 Register (Offset = 224h) [reset = 0h]: [0][1][2]
- MCASP_XBUF15 Register (Offset = 23Ch) [reset = 0h]: [0][1][2]
- MCASP_XBUF12 Register (Offset = 230h) [reset = 0h]: [0][1][2]

11.9.5.78 MCASP_XBUF1 Register (Offset = 204h) [reset = 0h]

MCASP_XBUF1 is shown in Figure 11-676 and described in Table 11-1481.

The transmit buffers for the serializers (MCASP_XBUF0) hold data from the transmit format unit. For transmit operations, the MCASP_XBUF0 is an alias of the XRBUF in the serializer. Accessing MCASP_XBUF0 registers not implemented on a specific device may cause improper device operation.

Table 11-1480. MCASP_XBUF1 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0204h
MCASP_1__CFG	0234 2204h
MCASP_2__CFG	0234 4204h

Figure 11-676. MCASP_XBUF1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1481. MCASP_XBUF1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	R/W	0h	Transmit buffers for serializers.

Table 11-1482. Register Call Summary for MCASP_XBUF1

McASP Registers
- McASP Registers: [0]
- MCASP_XBUF1 Register (Offset = 204h) [reset = 0h]: [0]

11.9.5.79 MCASP_XBUF2 Register (Offset = 208h) [reset = 0h]

MCASP_XBUF2 is shown in Figure 11-677 and described in Table 11-1484.

The transmit buffers for the serializers (MCASP_XBUF0) hold data from the transmit format unit. For transmit operations, the MCASP_XBUF0 is an alias of the XRBUF in the serializer. Accessing MCASP_XBUF0 registers not implemented on a specific device may cause improper device operation.

Table 11-1483. MCASP_XBUF2 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0208h
MCASP_1__CFG	0234 2208h
MCASP_2__CFG	0234 4208h

Figure 11-677. MCASP_XBUF2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1484. MCASP_XBUF2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	R/W	0h	Transmit buffers for serializers.

Table 11-1485. Register Call Summary for MCASP_XBUF2

McASP Functional Description • Transfers Through the Configuration Bus (CFG): [0]
McASP Registers • McASP Registers: [0] • MCASP_XBUF2 Register (Offset = 208h) [reset = 0h]: [0]

11.9.5.80 MCASP_XBUF3 Register (Offset = 20Ch) [reset = 0h]

MCASP_XBUF3 is shown in Figure 11-678 and described in Table 11-1487.

The transmit buffers for the serializers (MCASP_XBUF0) hold data from the transmit format unit. For transmit operations, the MCASP_XBUF0 is an alias of the XRBUF in the serializer. Accessing MCASP_XBUF0 registers not implemented on a specific device may cause improper device operation.

Table 11-1486. MCASP_XBUF3 Instances

Instance	Physical Address
MCASP_0__CFG	0234 020Ch
MCASP_1__CFG	0234 220Ch
MCASP_2__CFG	0234 420Ch

Figure 11-678. MCASP_XBUF3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1487. MCASP_XBUF3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	R/W	0h	Transmit buffers for serializers.

Table 11-1488. Register Call Summary for MCASP_XBUF3

McASP Registers
- McASP Registers: [0]
- MCASP_XBUF3 Register (Offset = 20Ch) [reset = 0h]: [0]

11.9.5.81 MCASP_XBUF4 Register (Offset = 210h) [reset = 0h]

MCASP_XBUF4 is shown in Figure 11-679 and described in Table 11-1490.

The transmit buffers for the serializers (MCASP_XBUF0) hold data from the transmit format unit. For transmit operations, the MCASP_XBUF0 is an alias of the XRBUF in the serializer. Accessing MCASP_XBUF0 registers not implemented on a specific device may cause improper device operation.

Table 11-1489. MCASP_XBUF4 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0210h
MCASP_1__CFG	0234 2210h
MCASP_2__CFG	0234 4210h

Figure 11-679. MCASP_XBUF4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1490. MCASP_XBUF4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	R/W	0h	Transmit buffers for serializers.

Table 11-1491. Register Call Summary for MCASP_XBUF4

McASP Registers
- McASP Registers: [0]
- MCASP_XBUF4 Register (Offset = 210h) [reset = 0h]: [0]

11.9.5.82 MCASP_XBUF5 Register (Offset = 214h) [reset = 0h]

MCASP_XBUF5 is shown in Figure 11-680 and described in Table 11-1493.

The transmit buffers for the serializers (MCASP_XBUF0) hold data from the transmit format unit. For transmit operations, the MCASP_XBUF0 is an alias of the XRBUF in the serializer. Accessing MCASP_XBUF0 registers not implemented on a specific device may cause improper device operation.

Table 11-1492. MCASP_XBUF5 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0214h
MCASP_1__CFG	0234 2214h
MCASP_2__CFG	0234 4214h

Figure 11-680. MCASP_XBUF5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1493. MCASP_XBUF5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	R/W	0h	Transmit buffers for serializers.

Table 11-1494. Register Call Summary for MCASP_XBUF5

McASP Registers
- McASP Registers: [0]
- MCASP_XBUF5 Register (Offset = 214h) [reset = 0h]: [0]

11.9.5.83 MCASP_XBUF6 Register (Offset = 218h) [reset = 0h]

MCASP_XBUF6 is shown in Figure 11-681 and described in Table 11-1496.

The transmit buffers for the serializers (MCASP_XBUF0) hold data from the transmit format unit. For transmit operations, the MCASP_XBUF0 is an alias of the XRBUF in the serializer. Accessing MCASP_XBUF0 registers not implemented on a specific device may cause improper device operation.

Table 11-1495. MCASP_XBUF6 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0218h
MCASP_1__CFG	0234 2218h
MCASP_2__CFG	0234 4218h

Figure 11-681. MCASP_XBUF6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1496. MCASP_XBUF6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	R/W	0h	Transmit buffers for serializers.

Table 11-1497. Register Call Summary for MCASP_XBUF6

McASP Registers
- McASP Registers: [0]
- MCASP_XBUF6 Register (Offset = 218h) [reset = 0h]: [0]

11.9.5.84 MCASP_XBUF7 Register (Offset = 21Ch) [reset = 0h]

MCASP_XBUF7 is shown in Figure 11-682 and described in Table 11-1499.

The transmit buffers for the serializers (MCASP_XBUF0) hold data from the transmit format unit. For transmit operations, the MCASP_XBUF0 is an alias of the XRBUF in the serializer. Accessing MCASP_XBUF0 registers not implemented on a specific device may cause improper device operation.

Table 11-1498. MCASP_XBUF7 Instances

Instance	Physical Address
MCASP_0__CFG	0234 021Ch
MCASP_1__CFG	0234 221Ch
MCASP_2__CFG	0234 421Ch

Figure 11-682. MCASP_XBUF7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1499. MCASP_XBUF7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	R/W	0h	Transmit buffers for serializers.

Table 11-1500. Register Call Summary for MCASP_XBUF7

McASP Registers
- McASP Registers: [0]
- MCASP_XBUF7 Register (Offset = 21Ch) [reset = 0h]: [0]

11.9.5.85 MCASP_XBUF8 Register (Offset = 220h) [reset = 0h]

MCASP_XBUF8 is shown in Figure 11-683 and described in Table 11-1502.

The transmit buffers for the serializers (MCASP_XBUF0) hold data from the transmit format unit. For transmit operations, the MCASP_XBUF0 is an alias of the XRBUF in the serializer. Accessing MCASP_XBUF0 registers not implemented on a specific device may cause improper device operation.

Table 11-1501. MCASP_XBUF8 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0220h
MCASP_1__CFG	0234 2220h
MCASP_2__CFG	0234 4220h

Figure 11-683. MCASP_XBUF8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1502. MCASP_XBUF8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	R/W	0h	Transmit buffers for serializers.

Table 11-1503. Register Call Summary for MCASP_XBUF8

McASP Registers
- MCASP_XBUF8 Register (Offset = 220h) [reset = 0h]: [0]
- McASP Registers: [0]

11.9.5.86 MCASP_XBUF9 Register (Offset = 224h) [reset = 0h]

MCASP_XBUF9 is shown in Figure 11-684 and described in Table 11-1505.

The transmit buffers for the serializers (MCASP_XBUF0) hold data from the transmit format unit. For transmit operations, the MCASP_XBUF0 is an alias of the XRBUF in the serializer. Accessing MCASP_XBUF0 registers not implemented on a specific device may cause improper device operation.

Table 11-1504. MCASP_XBUF9 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0224h
MCASP_1__CFG	0234 2224h
MCASP_2__CFG	0234 4224h

Figure 11-684. MCASP_XBUF9 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1505. MCASP_XBUF9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	R/W	0h	Transmit buffers for serializers.

Table 11-1506. Register Call Summary for MCASP_XBUF9

McASP Registers
- McASP Registers: [0]
- MCASP_XBUF9 Register (Offset = 224h) [reset = 0h]: [0]

11.9.5.87 MCASP_XBUF10 Register (Offset = 228h) [reset = 0h]

MCASP_XBUF10 is shown in Figure 11-685 and described in Table 11-1508.

The transmit buffers for the serializers (MCASP_XBUF0) hold data from the transmit format unit. For transmit operations, the MCASP_XBUF0 is an alias of the XRBUF in the serializer. Accessing MCASP_XBUF0 registers not implemented on a specific device may cause improper device operation.

Table 11-1507. MCASP_XBUF10 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0228h
MCASP_1__CFG	0234 2228h
MCASP_2__CFG	0234 4228h

Figure 11-685. MCASP_XBUF10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1508. MCASP_XBUF10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	R/W	0h	Transmit buffers for serializers.

Table 11-1509. Register Call Summary for MCASP_XBUF10

McASP Registers
- McASP Registers: [0]
- MCASP_XBUF10 Register (Offset = 228h) [reset = 0h]: [0]

11.9.5.88 MCASP_XBUF11 Register (Offset = 22Ch) [reset = 0h]

MCASP_XBUF11 is shown in Figure 11-686 and described in Table 11-1511.

The transmit buffers for the serializers (MCASP_XBUF0) hold data from the transmit format unit. For transmit operations, the MCASP_XBUF0 is an alias of the XRBUF in the serializer. Accessing MCASP_XBUF0 registers not implemented on a specific device may cause improper device operation.

Table 11-1510. MCASP_XBUF11 Instances

Instance	Physical Address
MCASP_0__CFG	0234 022Ch
MCASP_1__CFG	0234 222Ch
MCASP_2__CFG	0234 422Ch

Figure 11-686. MCASP_XBUF11 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1511. MCASP_XBUF11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	R/W	0h	Transmit buffers for serializers.

Table 11-1512. Register Call Summary for MCASP_XBUF11

McASP Registers
- McASP Registers: [0]
- MCASP_XBUF11 Register (Offset = 22Ch) [reset = 0h]: [0]

11.9.5.89 MCASP_XBUF12 Register (Offset = 230h) [reset = 0h]

MCASP_XBUF12 is shown in Figure 11-687 and described in Table 11-1514.

The transmit buffers for the serializers (MCASP_XBUF0) hold data from the transmit format unit. For transmit operations, the MCASP_XBUF0 is an alias of the XRBUF in the serializer. Accessing MCASP_XBUF0 registers not implemented on a specific device may cause improper device operation.

Table 11-1513. MCASP_XBUF12 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0230h
MCASP_1__CFG	0234 2230h
MCASP_2__CFG	0234 4230h

Figure 11-687. MCASP_XBUF12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1514. MCASP_XBUF12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	R/W	0h	Transmit buffers for serializers.

Table 11-1515. Register Call Summary for MCASP_XBUF12

McASP Registers
- MCASP_XBUF12 Register (Offset = 230h) [reset = 0h]: [0]
- McASP Registers: [0]

11.9.5.90 MCASP_XBUF13 Register (Offset = 234h) [reset = 0h]

MCASP_XBUF13 is shown in Figure 11-688 and described in Table 11-1517.

The transmit buffers for the serializers (MCASP_XBUF0) hold data from the transmit format unit. For transmit operations, the MCASP_XBUF0 is an alias of the XRBUF in the serializer. Accessing MCASP_XBUF0 registers not implemented on a specific device may cause improper device operation.

Table 11-1516. MCASP_XBUF13 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0234h
MCASP_1__CFG	0234 2234h
MCASP_2__CFG	0234 4234h

Figure 11-688. MCASP_XBUF13 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1517. MCASP_XBUF13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	R/W	0h	Transmit buffers for serializers.

Table 11-1518. Register Call Summary for MCASP_XBUF13

McASP Registers
- McASP Registers: [0]
- MCASP_XBUF13 Register (Offset = 234h) [reset = 0h]: [0]

11.9.5.91 MCASP_XBUF14 Register (Offset = 238h) [reset = 0h]

MCASP_XBUF14 is shown in Figure 11-689 and described in Table 11-1520.

The transmit buffers for the serializers (MCASP_XBUF0) hold data from the transmit format unit. For transmit operations, the MCASP_XBUF0 is an alias of the XRBUF in the serializer. Accessing MCASP_XBUF0 registers not implemented on a specific device may cause improper device operation.

Table 11-1519. MCASP_XBUF14 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0238h
MCASP_1__CFG	0234 2238h
MCASP_2__CFG	0234 4238h

Figure 11-689. MCASP_XBUF14 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1520. MCASP_XBUF14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	R/W	0h	Transmit buffers for serializers.

Table 11-1521. Register Call Summary for MCASP_XBUF14

McASP Registers
- McASP Registers: [0]
- MCASP_XBUF14 Register (Offset = 238h) [reset = 0h]: [0]

11.9.5.92 MCASP_XBUF15 Register (Offset = 23Ch) [reset = 0h]

MCASP_XBUF15 is shown in Figure 11-690 and described in Table 11-1523.

The transmit buffers for the serializers (MCASP_XBUF0) hold data from the transmit format unit. For transmit operations, the MCASP_XBUF0 is an alias of the XRBUF in the serializer. Accessing MCASP_XBUF0 registers not implemented on a specific device may cause improper device operation.

Table 11-1522. MCASP_XBUF15 Instances

Instance	Physical Address
MCASP_0__CFG	0234 023Ch
MCASP_1__CFG	0234 223Ch
MCASP_2__CFG	0234 423Ch

Figure 11-690. MCASP_XBUF15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1523. MCASP_XBUF15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	R/W	0h	Transmit buffers for serializers.

Table 11-1524. Register Call Summary for MCASP_XBUF15

McASP Functional Description • Buffer Underrun Error-Transmitter: [0] • McASP Serializers: [0][1] • Transmit Data Ready: [0][1][2][3][4][5][6] • Transfers Through the Configuration Bus (CFG): [0][1][2] • McASP TDM Sequencers: [0] • McASP Data Transmission and Reception: [0]
McASP Registers • McASP Registers: [0] • MCASP_XBUF15 Register (Offset = 23Ch) [reset = 0h]: [0]

11.9.5.93 MCASP_RBUF0 Register (Offset = 280h) [reset = 0h]

MCASP_RBUF0 is shown in Figure 11-691 and described in Table 11-1526.

The receive buffers for the serializers (MCASP_RBUF0) hold data from the serializer before the data goes to the receive format unit. For receive operations, the MCASP_RBUF0 is an alias of the XRBUF0 in the serializer. Accessing MCASP_XBUF registers not implemented on a specific device may cause improper device operation.

Table 11-1525. MCASP_RBUF0 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0280h
MCASP_1__CFG	0234 2280h
MCASP_2__CFG	0234 4280h

Figure 11-691. MCASP_RBUF0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1526. MCASP_RBUF0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R/W	0h	Receive buffers for serializers.

Table 11-1527. Register Call Summary for MCASP_RBUF0

McASP Functional Description • Receive Data Ready: [0][1][2][3][4][5][6] • Buffer Overrun Error-Receiver: [0] • McASP Data Transmission and Reception: [0] • McASP Serializers: [0][1] • Transfers Through the Configuration Bus (CFG): [0][1][2]
McASP Registers • MCASP_RBUF2 Register (Offset = 288h) [reset = 0h]: [0][1] • MCASP_RBUF3 Register (Offset = 28Ch) [reset = 0h]: [0][1] • MCASP_RBUF0 Register (Offset = 280h) [reset = 0h]: [0][1][2] • MCASP_RBUF1 Register (Offset = 284h) [reset = 0h]: [0][1] • MCASP_RBUF6 Register (Offset = 298h) [reset = 0h]: [0][1] • MCASP_RBUF7 Register (Offset = 29Ch) [reset = 0h]: [0][1] • MCASP_RBUF4 Register (Offset = 290h) [reset = 0h]: [0][1] • MCASP_RBUF5 Register (Offset = 294h) [reset = 0h]: [0][1] • MCASP_RBUF15 Register (Offset = 2BCh) [reset = 0h]: [0][1] • McASP Registers: [0] • MCASP_RBUF8 Register (Offset = 2A0h) [reset = 0h]: [0][1] • MCASP_RBUF9 Register (Offset = 2A4h) [reset = 0h]: [0][1] • MCASP_RBUF14 Register (Offset = 2B8h) [reset = 0h]: [0][1] • MCASP_RBUF10 Register (Offset = 2A8h) [reset = 0h]: [0][1] • MCASP_RBUF13 Register (Offset = 2B4h) [reset = 0h]: [0][1] • MCASP_RBUF12 Register (Offset = 2B0h) [reset = 0h]: [0][1] • MCASP_RBUF11 Register (Offset = 2ACh) [reset = 0h]: [0][1]

11.9.5.94 MCASP_RBUF1 Register (Offset = 284h) [reset = 0h]

MCASP_RBUF1 is shown in Figure 11-692 and described in Table 11-1529.

The receive buffers for the serializers (MCASP_RBUF0) hold data from the serializer before the data goes to the receive format unit. For receive operations, the MCASP_RBUF0 is an alias of the XRBUF0 in the serializer. Accessing MCASP_XBUF registers not implemented on a specific device may cause improper device operation.

Table 11-1528. MCASP_RBUF1 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0284h
MCASP_1__CFG	0234 2284h
MCASP_2__CFG	0234 4284h

Figure 11-692. MCASP_RBUF1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1529. MCASP_RBUF1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R/W	0h	Receive buffers for serializers.

Table 11-1530. Register Call Summary for MCASP_RBUF1

McASP Registers
- McASP Registers: [0]
- MCASP_RBUF1 Register (Offset = 284h) [reset = 0h]: [0]

11.9.5.95 MCASP_RBUF2 Register (Offset = 288h) [reset = 0h]

MCASP_RBUF2 is shown in Figure 11-693 and described in Table 11-1532.

The receive buffers for the serializers (MCASP_RBUF0) hold data from the serializer before the data goes to the receive format unit. For receive operations, the MCASP_RBUF0 is an alias of the XRBUF0 in the serializer. Accessing MCASP_XBUF registers not implemented on a specific device may cause improper device operation.

Table 11-1531. MCASP_RBUF2 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0288h
MCASP_1__CFG	0234 2288h
MCASP_2__CFG	0234 4288h

Figure 11-693. MCASP_RBUF2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1532. MCASP_RBUF2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R/W	0h	Receive buffers for serializers.

Table 11-1533. Register Call Summary for MCASP_RBUF2

McASP Registers
- MCASP_RBUF2 Register (Offset = 288h) [reset = 0h]: [0]
- McASP Registers: [0]

11.9.5.96 MCASP_RBUF3 Register (Offset = 28Ch) [reset = 0h]

MCASP_RBUF3 is shown in Figure 11-694 and described in Table 11-1535.

The receive buffers for the serializers (MCASP_RBUF0) hold data from the serializer before the data goes to the receive format unit. For receive operations, the MCASP_RBUF0 is an alias of the XRBUF0 in the serializer. Accessing MCASP_XBUF registers not implemented on a specific device may cause improper device operation.

Table 11-1534. MCASP_RBUF3 Instances

Instance	Physical Address
MCASP_0__CFG	0234 028Ch
MCASP_1__CFG	0234 228Ch
MCASP_2__CFG	0234 428Ch

Figure 11-694. MCASP_RBUF3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1535. MCASP_RBUF3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R/W	0h	Receive buffers for serializers.

Table 11-1536. Register Call Summary for MCASP_RBUF3

McASP Functional Description • Transfers Through the Configuration Bus (CFG): [0]
McASP Registers • McASP Registers: [0] • MCASP_RBUF3 Register (Offset = 28Ch) [reset = 0h]: [0]

11.9.5.97 MCASP_RBUF4 Register (Offset = 290h) [reset = 0h]

MCASP_RBUF4 is shown in Figure 11-695 and described in Table 11-1538.

The receive buffers for the serializers (MCASP_RBUF0) hold data from the serializer before the data goes to the receive format unit. For receive operations, the MCASP_RBUF0 is an alias of the XRBUF0 in the serializer. Accessing MCASP_XBUF registers not implemented on a specific device may cause improper device operation.

Table 11-1537. MCASP_RBUF4 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0290h
MCASP_1__CFG	0234 2290h
MCASP_2__CFG	0234 4290h

Figure 11-695. MCASP_RBUF4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1538. MCASP_RBUF4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R/W	0h	Receive buffers for serializers.

Table 11-1539. Register Call Summary for MCASP_RBUF4

McASP Registers
- McASP Registers: [0]
- MCASP_RBUF4 Register (Offset = 290h) [reset = 0h]: [0]

11.9.5.98 MCASP_RBUF5 Register (Offset = 294h) [reset = 0h]

MCASP_RBUF5 is shown in Figure 11-696 and described in Table 11-1541.

The receive buffers for the serializers (MCASP_RBUF0) hold data from the serializer before the data goes to the receive format unit. For receive operations, the MCASP_RBUF0 is an alias of the XRBUF0 in the serializer. Accessing MCASP_XBUF registers not implemented on a specific device may cause improper device operation.

Table 11-1540. MCASP_RBUF5 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0294h
MCASP_1__CFG	0234 2294h
MCASP_2__CFG	0234 4294h

Figure 11-696. MCASP_RBUF5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1541. MCASP_RBUF5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R/W	0h	Receive buffers for serializers.

Table 11-1542. Register Call Summary for MCASP_RBUF5

McASP Registers
- McASP Registers: [0]
- MCASP_RBUF5 Register (Offset = 294h) [reset = 0h]: [0]

11.9.5.99 MCASP_RBUF6 Register (Offset = 298h) [reset = 0h]

MCASP_RBUF6 is shown in Figure 11-697 and described in Table 11-1544.

The receive buffers for the serializers (MCASP_RBUF0) hold data from the serializer before the data goes to the receive format unit. For receive operations, the MCASP_RBUF0 is an alias of the XRBUF0 in the serializer. Accessing MCASP_XBUF registers not implemented on a specific device may cause improper device operation.

Table 11-1543. MCASP_RBUF6 Instances

Instance	Physical Address
MCASP_0__CFG	0234 0298h
MCASP_1__CFG	0234 2298h
MCASP_2__CFG	0234 4298h

Figure 11-697. MCASP_RBUF6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1544. MCASP_RBUF6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R/W	0h	Receive buffers for serializers.

Table 11-1545. Register Call Summary for MCASP_RBUF6

McASP Registers
- McASP Registers: [0]
- MCASP_RBUF6 Register (Offset = 298h) [reset = 0h]: [0]

11.9.5.100 MCASP_RBUF7 Register (Offset = 29Ch) [reset = 0h]

MCASP_RBUF7 is shown in Figure 11-698 and described in Table 11-1547.

The receive buffers for the serializers (MCASP_RBUF0) hold data from the serializer before the data goes to the receive format unit. For receive operations, the MCASP_RBUF0 is an alias of the XRBUF0 in the serializer. Accessing MCASP_XBUF registers not implemented on a specific device may cause improper device operation.

Table 11-1546. MCASP_RBUF7 Instances

Instance	Physical Address
MCASP_0__CFG	0234 029Ch
MCASP_1__CFG	0234 229Ch
MCASP_2__CFG	0234 429Ch

Figure 11-698. MCASP_RBUF7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1547. MCASP_RBUF7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R/W	0h	Receive buffers for serializers.

Table 11-1548. Register Call Summary for MCASP_RBUF7

McASP Registers
- McASP Registers: [0]
- MCASP_RBUF7 Register (Offset = 29Ch) [reset = 0h]: [0]

11.9.5.101 MCASP_RBUF8 Register (Offset = 2A0h) [reset = 0h]

MCASP_RBUF8 is shown in Figure 11-699 and described in Table 11-1550.

The receive buffers for the serializers (MCASP_RBUF0) hold data from the serializer before the data goes to the receive format unit. For receive operations, the MCASP_RBUF0 is an alias of the XRBUF0 in the serializer. Accessing MCASP_XBUF registers not implemented on a specific device may cause improper device operation.

Table 11-1549. MCASP_RBUF8 Instances

Instance	Physical Address
MCASP_0__CFG	0234 02A0h
MCASP_1__CFG	0234 22A0h
MCASP_2__CFG	0234 42A0h

Figure 11-699. MCASP_RBUF8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1550. MCASP_RBUF8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R/W	0h	Receive buffers for serializers.

Table 11-1551. Register Call Summary for MCASP_RBUF8

McASP Registers
- McASP Registers: [0]
- MCASP_RBUF8 Register (Offset = 2A0h) [reset = 0h]: [0]

11.9.5.102 MCASP_RBUF9 Register (Offset = 2A4h) [reset = 0h]

MCASP_RBUF9 is shown in Figure 11-700 and described in Table 11-1553.

The receive buffers for the serializers (MCASP_RBUF0) hold data from the serializer before the data goes to the receive format unit. For receive operations, the MCASP_RBUF0 is an alias of the XRBUF0 in the serializer. Accessing MCASP_XBUF registers not implemented on a specific device may cause improper device operation.

Table 11-1552. MCASP_RBUF9 Instances

Instance	Physical Address
MCASP_0__CFG	0234 02A4h
MCASP_1__CFG	0234 22A4h
MCASP_2__CFG	0234 42A4h

Figure 11-700. MCASP_RBUF9 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1553. MCASP_RBUF9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R/W	0h	Receive buffers for serializers.

Table 11-1554. Register Call Summary for MCASP_RBUF9

McASP Registers
- McASP Registers: [0]
- MCASP_RBUF9 Register (Offset = 2A4h) [reset = 0h]: [0]

11.9.5.103 MCASP_RBUF10 Register (Offset = 2A8h) [reset = 0h]

MCASP_RBUF10 is shown in Figure 11-701 and described in Table 11-1556.

The receive buffers for the serializers (MCASP_RBUF0) hold data from the serializer before the data goes to the receive format unit. For receive operations, the MCASP_RBUF0 is an alias of the XRBUF0 in the serializer. Accessing MCASP_XBUF registers not implemented on a specific device may cause improper device operation.

Table 11-1555. MCASP_RBUF10 Instances

Instance	Physical Address
MCASP_0__CFG	0234 02A8h
MCASP_1__CFG	0234 22A8h
MCASP_2__CFG	0234 42A8h

Figure 11-701. MCASP_RBUF10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1556. MCASP_RBUF10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R/W	0h	Receive buffers for serializers.

Table 11-1557. Register Call Summary for MCASP_RBUF10

McASP Registers
- McASP Registers: [0]

11.9.5.104 MCASP_RBUF11 Register (Offset = 2ACh) [reset = 0h]

MCASP_RBUF11 is shown in Figure 11-702 and described in Table 11-1559.

The receive buffers for the serializers (MCASP_RBUF0) hold data from the serializer before the data goes to the receive format unit. For receive operations, the MCASP_RBUF0 is an alias of the XRBUF0 in the serializer. Accessing MCASP_XBUF registers not implemented on a specific device may cause improper device operation.

Table 11-1558. MCASP_RBUF11 Instances

Instance	Physical Address
MCASP_0__CFG	0234 02ACh
MCASP_1__CFG	0234 22ACh
MCASP_2__CFG	0234 42ACh

Figure 11-702. MCASP_RBUF11 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1559. MCASP_RBUF11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R/W	0h	Receive buffers for serializers.

Table 11-1560. Register Call Summary for MCASP_RBUF11

McASP Registers
- McASP Registers: [0]
- MCASP_RBUF11 Register (Offset = 2ACh) [reset = 0h]: [0]

11.9.5.105 MCASP_RBUF12 Register (Offset = 2B0h) [reset = 0h]

MCASP_RBUF12 is shown in Figure 11-703 and described in Table 11-1562.

The receive buffers for the serializers (MCASP_RBUF0) hold data from the serializer before the data goes to the receive format unit. For receive operations, the MCASP_RBUF0 is an alias of the XRBUF0 in the serializer. Accessing MCASP_XBUF registers not implemented on a specific device may cause improper device operation.

Table 11-1561. MCASP_RBUF12 Instances

Instance	Physical Address
MCASP_0__CFG	0234 02B0h
MCASP_1__CFG	0234 22B0h
MCASP_2__CFG	0234 42B0h

Figure 11-703. MCASP_RBUF12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1562. MCASP_RBUF12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R/W	0h	Receive buffers for serializers.

Table 11-1563. Register Call Summary for MCASP_RBUF12

McASP Registers
- McASP Registers: [0]
- MCASP_RBUF12 Register (Offset = 2B0h) [reset = 0h]: [0]

11.9.5.106 MCASP_RBUF13 Register (Offset = 2B4h) [reset = 0h]

MCASP_RBUF13 is shown in Figure 11-704 and described in Table 11-1565.

The receive buffers for the serializers (MCASP_RBUF0) hold data from the serializer before the data goes to the receive format unit. For receive operations, the MCASP_RBUF0 is an alias of the XRBUF0 in the serializer. Accessing MCASP_XBUF registers not implemented on a specific device may cause improper device operation.

Table 11-1564. MCASP_RBUF13 Instances

Instance	Physical Address
MCASP_0__CFG	0234 02B4h
MCASP_1__CFG	0234 22B4h
MCASP_2__CFG	0234 42B4h

Figure 11-704. MCASP_RBUF13 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1565. MCASP_RBUF13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R/W	0h	Receive buffers for serializers.

Table 11-1566. Register Call Summary for MCASP_RBUF13

McASP Registers
- McASP Registers: [0]
- MCASP_RBUF13 Register (Offset = 2B4h) [reset = 0h]: [0]

11.9.5.107 MCASP_RBUF14 Register (Offset = 2B8h) [reset = 0h]

MCASP_RBUF14 is shown in Figure 11-705 and described in Table 11-1568.

The receive buffers for the serializers (MCASP_RBUF0) hold data from the serializer before the data goes to the receive format unit. For receive operations, the MCASP_RBUF0 is an alias of the XRBUF0 in the serializer. Accessing MCASP_XBUF registers not implemented on a specific device may cause improper device operation.

Table 11-1567. MCASP_RBUF14 Instances

Instance	Physical Address
MCASP_0__CFG	0234 02B8h
MCASP_1__CFG	0234 22B8h
MCASP_2__CFG	0234 42B8h

Figure 11-705. MCASP_RBUF14 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1568. MCASP_RBUF14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R/W	0h	Receive buffers for serializers.

Table 11-1569. Register Call Summary for MCASP_RBUF14

McASP Registers
- MCASP_RBUF14 Register (Offset = 2B8h) [reset = 0h]: [0]
- McASP Registers: [0]

11.9.5.108 MCASP_RBUF15 Register (Offset = 2BCh) [reset = 0h]

MCASP_RBUF15 is shown in Figure 11-706 and described in Table 11-1571.

The receive buffers for the serializers (MCASP_RBUF0) hold data from the serializer before the data goes to the receive format unit. For receive operations, the MCASP_RBUF0 is an alias of the XRBUF0 in the serializer. Accessing MCASP_XBUF registers not implemented on a specific device may cause improper device operation.

Table 11-1570. MCASP_RBUF15 Instances

Instance	Physical Address
MCASP_0__CFG	0234 02BCh
MCASP_1__CFG	0234 22BCh
MCASP_2__CFG	0234 42BCh

Figure 11-706. MCASP_RBUF15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1571. MCASP_RBUF15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R/W	0h	Receive buffers for serializers.

Table 11-1572. Register Call Summary for MCASP_RBUF15

McASP Functional Description • Receive Data Ready: [0][1][2][3][4][5][6] • Buffer Overrun Error-Receiver: [0] • McASP Data Transmission and Reception: [0] • McASP Serializers: [0][1] • Transfers Through the Configuration Bus (CFG): [0][1][2]
McASP Registers • McASP Registers: [0] • MCASP_RBUF15 Register (Offset = 2BCh) [reset = 0h]: [0]

11.9.5.109 MCASP_WFIFOCTL Register (Offset = 1000h) [reset = 0h]

MCASP_WFIFOCTL is shown in Figure 11-707 and described in Table 11-1574.

The WNUMEVT and WNUMDMA values must be set prior to enabling the Write FIFO. If the Write FIFO is to be enabled, it must be enabled prior to taking the McASP out of reset

Table 11-1573. MCASP_WFIFOCTL Instances

Instance	Physical Address
MCASP_0__FIFO_CFG	0234 1400h
MCASP_1__FIFO_CFG	0234 3400h
MCASP_2__FIFO_CFG	0234 5400h

Figure 11-707. MCASP_WFIFOCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							WENA
R-0h							R/W-0h
15	14	13	12	11	10	9	8
WNUMEVT							
R/W-0h							
7	6	5	4	3	2	1	0
WNUMDMA							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1574. MCASP_WFIFOCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	WENA	R/W	0h	Write FIFO enable bit. 0h (R/W) = Write FIFO is disabled. The WLVL bit in the Write FIFO status register (MCASP_WFIFOSTS) is reset to 0 and pointers are initialized, that is, the Write FIFO is flushed. 1h (R/W) = Write FIFO is enabled. If Write FIFO is to be enabled, it must be enabled prior to taking McASP out of reset.
15-8	WNUMEVT	R/W	0h	Write word count per DMA event (32 bit). When the Write FIFO has space for at least WNUMEVT words of data, then an AXEVT (transmit DMA event) is generated to the host/DMA controller. This value should be set to a non-zero integer multiple of the number of serializers enabled as transmitters. This value must be set prior to enabling the Write FIFO. 40h = 3 to 64 words from 3h to 40h. FFh = Reserved from 41h to FFh. 0h (R/W) = 0 words 1h (R/W) = 1 word 2h (R/W) = 2 words 3h (R/W) = 3 to 64 words from 3h to 40h. 41h (R/W) = Reserved from 41h to FFh.

Table 11-1574. MCASP_WFIFOCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-0	WNUMDMA	R/W	0h	Write word count per transfer (32 bit words). Upon a transmit DMA event from the McASP, WNUMDMA words are transferred from the Write FIFO to the McASP. This value must equal the number of McASP serializers used as transmitters. This value must be set prior to enabling the Write FIFO. FFh = Reserved from 11h to FFh. 0h (R/W) = 0 words 1h (R/W) = 1 word 2h (R/W) = 2 words 3h (R/W) = 3-16 words from 3h to 10h. 11h (R/W) = Reserved from 11h to FFh.

Table 11-1575. Register Call Summary for MCASP_WFIFOCTL

McASP Functional Description • McASP Audio FIFO (AFIFO): [0] • AFIFO Data Transmission: [0] • Transmit DMA Event Pacer: [0][1][2][3]
McASP Registers • McASP Registers: [0] • MCASP_WFIFOCTL Register (Offset = 1000h) [reset = 0h]: [0]

11.9.5.110 MCASP_WFIFOSTS Register (Offset = 1004h) [reset = 0h]

MCASP_WFIFOSTS is shown in Figure 11-708 and described in Table 11-1577.

Table 11-1576. MCASP_WFIFOSTS Instances

Instance	Physical Address
MCASP_0__FIFO_CFG	0234 1404h
MCASP_1__FIFO_CFG	0234 3404h
MCASP_2__FIFO_CFG	0234 5404h

Figure 11-708. MCASP_WFIFOSTS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								WLVL							
R-0h																								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1577. MCASP_WFIFOSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	WLVL	R	0h	Write level (read-only). Number of 32 bit words currently in the Write FIFO. 40h = 3 to 64 words currently in Write FIFO from 3h to 40h. FFh = Reserved from 41h to FFh. 0h (R/W) = 0 words currently in Write FIFO. 1h (R/W) = 1 word currently in Write FIFO. 2h (R/W) = 2 words currently in Write FIFO. 3h (R/W) = 3 to 64 words currently in Write FIFO from 3h to 40h. 41h (R/W) = Reserved from 41h to FFh.

Table 11-1578. Register Call Summary for MCASP_WFIFOSTS

McASP Registers
- McASP Registers: [0]
- MCASP_WFIFOSTS Register (Offset = 1004h) [reset = 0h]: [0]
- MCASP_WFIFOCTL Register (Offset = 1000h) [reset = 0h]: [0]

11.9.5.111 MCASP_RFIFOCTL Register (Offset = 1008h) [reset = 0h]

MCASP_RFIFOCTL is shown in Figure 11-709 and described in Table 11-1580.

The RNUMEVT and RNUMDMA values must be set prior to enabling the Read FIFO. If the Read FIFO is to be enabled, it must be enabled prior to taking the McASP out of reset

Table 11-1579. MCASP_RFIFOCTL Instances

Instance	Physical Address
MCASP_0__FIFO_CFG	0234 1408h
MCASP_1__FIFO_CFG	0234 3408h
MCASP_2__FIFO_CFG	0234 5408h

Figure 11-709. MCASP_RFIFOCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							RENA
R-0h							R/W-0h
15	14	13	12	11	10	9	8
RNUMEVT							
R/W-0h							
7	6	5	4	3	2	1	0
RNUMDMA							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1580. MCASP_RFIFOCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	RENA	R/W	0h	Read FIFO enable bit. 0h (R/W) = Read FIFO is disabled. The RLVL bit in the Read FIFO status register (MCASP_RFIFOSTS) is reset to 0 and pointers are initialized, that is, the Read FIFO is flushed. 1h (R/W) = Read FIFO is enabled. If Read FIFO is to be enabled, it must be enabled prior to taking McASP out of reset.
15-8	RNUMEVT	R/W	0h	Read word count per DMA event (32 bit). When the Read FIFO contains at least RNUMEVT words of data, then an AREVT (receive DMA event) is generated to the host/DMA controller. This value should be set to a non-zero integer multiple of the number of serializers enabled as receivers. This value must be set prior to enabling the Read FIFO. 40h = 3 to 64 words from 3h to 40h. FFh = Reserved from 41h = FFh. 0h (R/W) = 0 words 1h (R/W) = 1 word 2h (R/W) = 2 words 3h (R/W) = 3 to 64 words from 3h to 40h. 41h (R/W) = Reserved from 41h to FFh.

Table 11-1580. MCASP_RFIFOCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-0	RNUMDMA	R/W	0h	Read word count per transfer (32 bit words). Upon a receive DMA event from the McASP, the Read FIFO reads RNUMDMA words from the McASP. This value must equal the number of McASP serializers used as receivers. This value must be set prior to enabling the Read FIFO. 10h = 3 to 16 words from 3h to 10h. FFh = Reserved from 11h to FFh. 0h (R/W) = 0 words 1h (R/W) = 1 word 2h (R/W) = 2 words 3h (R/W) = 3 to 16 words from 3h to 10h. 11h (R/W) = Reserved from 11h to FFh.

Table 11-1581. Register Call Summary for MCASP_RFIFOCTL

McASP Functional Description • McASP Audio FIFO (AFIFO): [0] • Receive DMA Event Pacer: [0][1][2][3] • AFIFO Data Reception: [0]
McASP Registers • McASP Registers: [0] • MCASP_RFIFOCTL Register (Offset = 1008h) [reset = 0h]: [0]

11.9.5.112 MCASP_RFIFOSTS Register (Offset = 100Ch) [reset = 0h]

MCASP_RFIFOSTS is shown in Figure 11-710 and described in Table 11-1583.

Table 11-1582. MCASP_RFIFOSTS Instances

Instance	Physical Address
MCASP_0__FIFO_CFG	0234 140Ch
MCASP_1__FIFO_CFG	0234 340Ch
MCASP_2__FIFO_CFG	0234 540Ch

Figure 11-710. MCASP_RFIFOSTS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								RLVL							
R-0h																								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1583. MCASP_RFIFOSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	RLVL	R	0h	Read level (read-only). Number of 32 bit words currently in the Read FIFO. 40h = 3 to 64 words currently in Read FIFO from 3h to 40h. FFh = Reserved from 41h to FFh. 0h (R/W) = 0 words currently in Read FIFO. 1h (R/W) = 1 word currently in Read FIFO. 2h (R/W) = 2 words currently in Read FIFO. 3h (R/W) = 3 to 64 words currently in Read FIFO from 3h to 40h. 41h (R/W) = Reserved from 41h to FFh.

Table 11-1584. Register Call Summary for MCASP_RFIFOSTS

McASP Registers
- McASP Registers: [0]
- MCASP_RFIFOCTL Register (Offset = 1008h) [reset = 0h]: [0]
- MCASP_RFIFOSTS Register (Offset = 100Ch) [reset = 0h]: [0]

11.9.5.113 MCASP_XBUF Register (Offset = 0h) [reset = 0h]

MCASP_XBUF is shown in Figure 11-711 and described in Table 11-1586.

Through the DATA port, the Host can service all serializers through a single address and the MCASP automatically cycles through the appropriate serializers. For transmit operations through the DATA port, the Host should write to the same DATA port address to service all of the active transmit serializers upon each transmit data ready event. To enable accesses from the Host to the MCASP XRBUFn registers through the DATA port, one must clear the XBUSEL bits to 0 in the respective MCASP_XFMT registers.

Table 11-1585. MCASP_XBUF Instances

Instance	Physical Address
MCASP_0_SLV	2180 4000h
MCASP_1_SLV	2180 4400h
MCASP_2_SLV	2180 4800h

Figure 11-711. MCASP_XBUF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XBUF																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 11-1586. MCASP_XBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	XBUF	W	0h	Tx buffer data.

Table 11-1587. Register Call Summary for MCASP_XBUF

McASP Registers

- MCASP_RBUF14 Register (Offset = 2B8h) [reset = 0h]: [0]
- MCASP_RBUF15 Register (Offset = 2BCh) [reset = 0h]: [0]
- MCASP_XSTAT Register (Offset = C0h) [reset = 0h]: [0][1][2][3][4]
- MCASP_RBUF10 Register (Offset = 2A8h) [reset = 0h]: [0]
- MCASP_RBUF11 Register (Offset = 2ACh) [reset = 0h]: [0]
- MCASP_RBUF12 Register (Offset = 2B0h) [reset = 0h]: [0]
- MCASP_RBUF13 Register (Offset = 2B4h) [reset = 0h]: [0]
- MCASP_SRCTL12 Register (Offset = 1B0h) [reset = 0h]: [0][1]
- MCASP_SRCTL13 Register (Offset = 1B4h) [reset = 0h]: [0][1]
- MCASP_SRCTL10 Register (Offset = 1A8h) [reset = 0h]: [0][1]
- MCASP_SRCTL11 Register (Offset = 1ACh) [reset = 0h]: [0][1]
- MCASP_SRCTL14 Register (Offset = 1B8h) [reset = 0h]: [0][1]
- MCASP_SRCTL15 Register (Offset = 1BCh) [reset = 0h]: [0][1]
- MCASP_XBUF Register (Offset = 0h) [reset = 0h]: [0]
- MCASP_RBUF2 Register (Offset = 288h) [reset = 0h]: [0]
- MCASP_RBUF3 Register (Offset = 28Ch) [reset = 0h]: [0]
- MCASP_RBUF0 Register (Offset = 280h) [reset = 0h]: [0]
- MCASP_RBUF1 Register (Offset = 284h) [reset = 0h]: [0]
- MCASP_RBUF6 Register (Offset = 298h) [reset = 0h]: [0]
- MCASP_RBUF7 Register (Offset = 29Ch) [reset = 0h]: [0]
- MCASP_RBUF4 Register (Offset = 290h) [reset = 0h]: [0]
- MCASP_RBUF5 Register (Offset = 294h) [reset = 0h]: [0]
- McASP Registers: [0][1]
- MCASP_RBUF8 Register (Offset = 2A0h) [reset = 0h]: [0]
- MCASP_RBUF9 Register (Offset = 2A4h) [reset = 0h]: [0]
- MCASP_SRCTL0 Register (Offset = 180h) [reset = 0h]: [0][1]
- MCASP_SRCTL1 Register (Offset = 184h) [reset = 0h]: [0][1]
- MCASP_SRCTL2 Register (Offset = 188h) [reset = 0h]: [0][1]
- MCASP_SRCTL3 Register (Offset = 18Ch) [reset = 0h]: [0][1]
- MCASP_SRCTL4 Register (Offset = 190h) [reset = 0h]: [0][1]
- MCASP_SRCTL5 Register (Offset = 194h) [reset = 0h]: [0][1]
- MCASP_SRCTL6 Register (Offset = 198h) [reset = 0h]: [0][1]
- MCASP_SRCTL7 Register (Offset = 19Ch) [reset = 0h]: [0][1]
- MCASP_SRCTL8 Register (Offset = 1A0h) [reset = 0h]: [0][1]
- MCASP_SRCTL9 Register (Offset = 1A4h) [reset = 0h]: [0][1]
- MCASP_XFMT Register (Offset = A8h) [reset = 0h]: [0]
- MCASP_GBLCTL Register (Offset = 44h) [reset = 0h]: [0][1]

11.9.5.114 MCASP_RBUF Register (Offset = 0h) [reset = 0h]

MCASP_RBUF is shown in Figure 11-712 and described in Table 11-1589.

Through the DATA port , the Host can service all serializers through a single address and the MCASP automatically cycles through the appropriate serializers. For receive operations through the DATA port, the Host should read from the same MCASP_RBUF DATA port address to service all of the active receive serializers upon each receive data ready event. To enable accesses from the Host to the MCASP XRBUFn registers through the DATA port, one must clear the RBUSEL bits to 0 in the respective MCASP_RFMT registers.

Table 11-1588. MCASP_RBUF Instances

Instance	Physical Address
MCASP_0_SLV	2180 4000h
MCASP_1_SLV	2180 4400h
MCASP_2_SLV	2180 4800h

Figure 11-712. MCASP_RBUF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBUF																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1589. MCASP_RBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RBUF	R	0h	Rx buffer data.

Table 11-1590. Register Call Summary for MCASP_RBUF

McASP Registers
- MCASP_SRCTL13 Register (Offset = 1B4h) [reset = 0h]: [0][1][2]
- MCASP_RBUF Register (Offset = 0h) [reset = 0h]: [0][1]
- MCASP_RSTAT Register (Offset = 80h) [reset = 0h]: [0][1][2][3]
- MCASP_SRCTL12 Register (Offset = 1B0h) [reset = 0h]: [0][1][2]
- McASP Registers: [0][1]
- MCASP_SRCTL10 Register (Offset = 1A8h) [reset = 0h]: [0][1][2]
- MCASP_SRCTL11 Register (Offset = 1ACh) [reset = 0h]: [0][1][2]
- MCASP_SRCTL14 Register (Offset = 1B8h) [reset = 0h]: [0][1][2]
- MCASP_RFMT Register (Offset = 68h) [reset = 0h]: [0]
- MCASP_SRCTL0 Register (Offset = 180h) [reset = 0h]: [0][1][2]
- MCASP_SRCTL1 Register (Offset = 184h) [reset = 0h]: [0][1][2]
- MCASP_SRCTL2 Register (Offset = 188h) [reset = 0h]: [0][1][2]
- MCASP_SRCTL3 Register (Offset = 18Ch) [reset = 0h]: [0][1][2]
- MCASP_SRCTL4 Register (Offset = 190h) [reset = 0h]: [0][1][2]
- MCASP_SRCTL5 Register (Offset = 194h) [reset = 0h]: [0][1][2]
- MCASP_SRCTL6 Register (Offset = 198h) [reset = 0h]: [0][1][2]
- MCASP_SRCTL7 Register (Offset = 19Ch) [reset = 0h]: [0][1][2]
- MCASP_SRCTL8 Register (Offset = 1A0h) [reset = 0h]: [0][1][2]
- MCASP_SRCTL9 Register (Offset = 1A4h) [reset = 0h]: [0][1][2]
- MCASP_SRCTL15 Register (Offset = 1BCh) [reset = 0h]: [0][1][2]

11.10 Multi-channel Buffered Serial Port (McBSP)

This section describes the Multi-channel Buffered Serial Port (McBSP) module in the device.

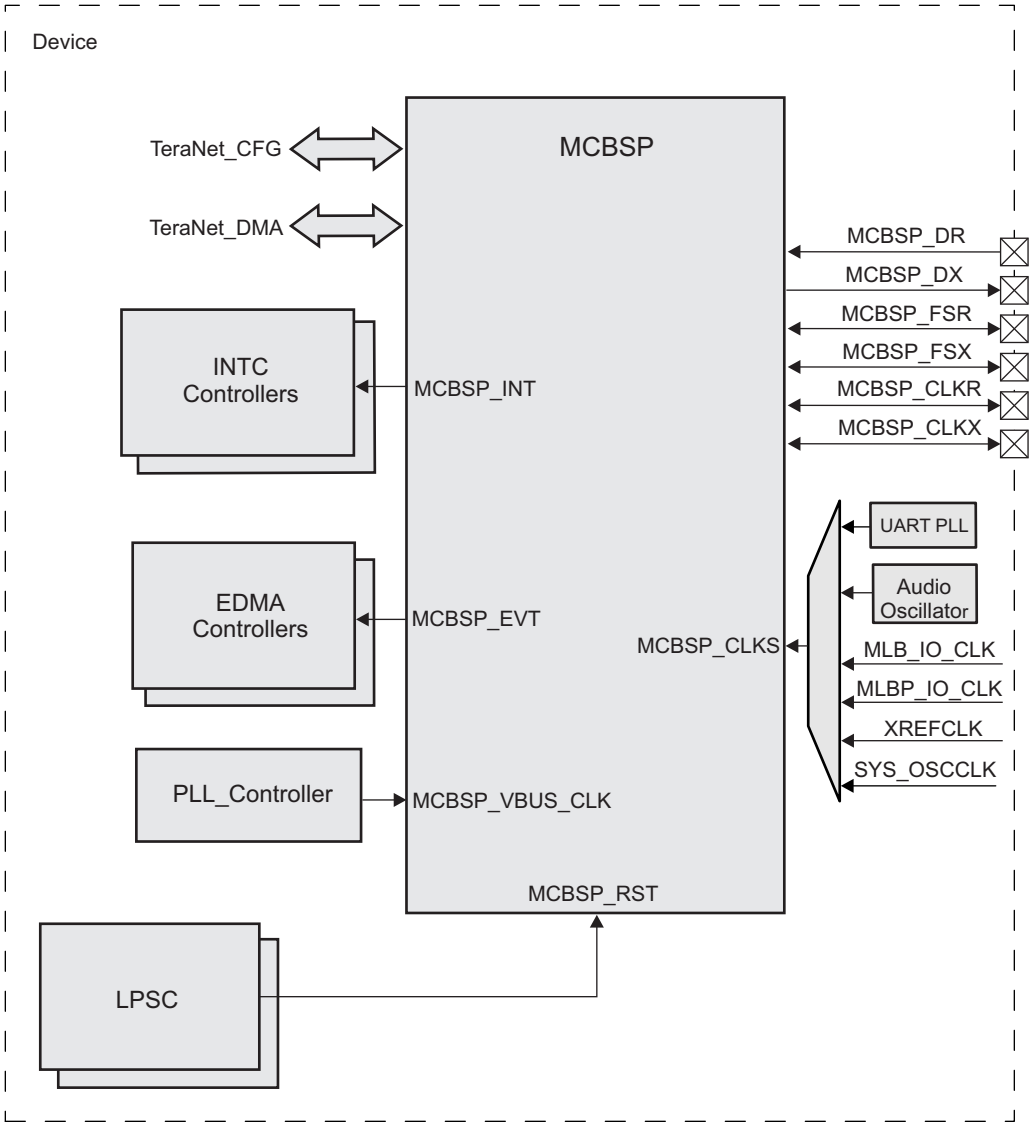
11.10.1 McBSP Overview

This section describes the Multi-channel Buffered Serial Port (McBSP) module in the device.

The Multi-channel Buffered Serial Port (McBSP) provides a full-duplex serial communication interface between the device and other devices in a system. The primary use for the McBSP is for audio interface purposes. The main audio modes that are supported are the AC97 and I^2S modes. In addition to the primary audio modes, the McBSP can be programmed to support other serial formats but is not intended to be used as a high-speed interface. The device communicates to the McBSP using 32-bit-wide control registers accessible via the internal peripheral bus.

Figure 11-713 shows an overview of the McBSP module in the device.

Figure 11-713. McBSP Module Overview



11.10.1.1 Features

The McBSP provides the following functions:

- Full-duplex communication
- Double-buffered data registers, which allow a continuous data stream
- Independent framing and clocking for receive and transmit
- Direct interface to industry-standard codecs, analog interface chips (AICs), and other serially connected analog-to-digital (A/D) and digital-to-analog (D/A) devices
- External shift clock or an internal, programmable frequency shift clock for data transfer

In addition, the McBSP has the following capabilities:

- Direct interface to:
  - T1/E1 framers
  - MVIP switching compatible and ST-BUS compliant devices including:
    - MVIP framers
    - H.100 framers
    - SCSA framers
  - IOM-2 compliant devices
  - AC97 compliant devices (the necessary multiphase frame synchronization capability is provided)
  - I^2S compliant devices
- Multi-channel transmit and receive of up to 128 channels
- A wide selection of data sizes, including 8, 12, 16, 20, 24, and 32 bits
- μ-Law and A-Law companding
- 8-bit data transfers with the option of LSB or MSB first
- Programmable polarity for both frame synchronization and data clocks
- Highly programmable internal clock and frame generation
- Additional McBSP Buffer FIFO (BFIFO):
  - Provides additional data buffering
  - Provides added tolerance to variations in host/DMA controller response times
  - May be used as a DMA event pacer
  - Independent Read FIFO and Write FIFO
  - 256 bytes of RAM for each FIFO (read and write)
  - Option to bypass Write FIFO and/or Read FIFO, independently

McBSP module unsupported features:

- The McBSP on this device does not support the SPI protocol.
- 512 Channel Mode
- Individual enable/disable channel control
- Timeslot buffering
- Super frame synchronization
- ABIS Mode

11.10.1.2 Industry Standard Compliance Statement

The McBSP supports the following industry standard interfaces:

AC97 — The AC97 standard specifies a 5-wire digital serial link between an audio codec device and its digital controller.

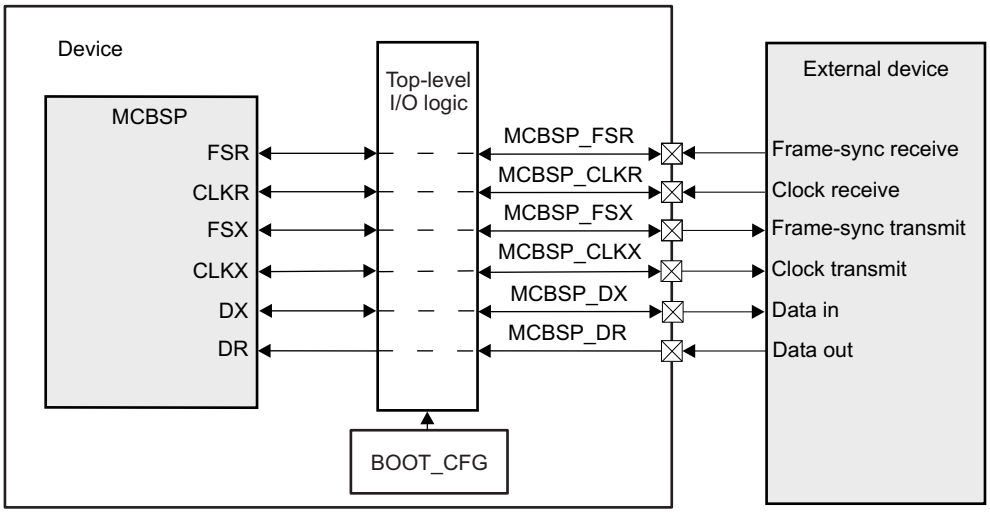
Inter-IC Sound — I^2S is a protocol specifically designed to transfer a stereo channel (left and right) over a single data pin. The I^2S bus is an industry standard interface for streaming stereo audio between devices, typically between a CPU/DSP and a DAC/ADC.

11.10.2 McBSP Environment

This section describes the McBSP application fields from an environment point of view (external connections).

Figure 11-714 shows the connection between the McBSP module and an external device.

Figure 11-714. McBSP Module Environment



11.10.2.1 Signal Descriptions

The McBSP consists of a data path and a control path that connect to external devices. Separate pins for transmission and reception communicate data to these external devices. The data is communicated from devices via the Data Transmit (DX) pin for transmit and the Data Receive (DR) pin for receive. Control information in the form of clocking and frame synchronization is communicated via CLKS, CLKX, CLKR, FSX and FSR.

Table 11-1591 lists the pins associated with the McBSP interface.

Table 11-1591. McBSP Input/Output Signals

Module Pin Name	Device Level Signal	I/O/Z[1]	Description
CLKR	MCBSP_CLKR	I/O/Z	Receive serial clock. Supplies or receives a reference clock for the receiver.
CLKX	MCBSP_CLKX	I/O/Z	Transmit serial clock. Supplies or receives a reference clock for the transmitter.
DR	MCBSP_DR	I	Received serial data
DX	MCBSP_DX	O/Z	Transmitted serial data
FSR	MCBSP_FSR	I/O/Z	Receive frame synchronization. Control signal to synchronize the start of received data.
FSX	MCBSP_FSX	I/O/Z	Transmit frame synchronization. Control signal to synchronize the start of transmitted data.

(1) I = Input; O = Output; I/O = Bidirectional;

11.10.2.2 Pad Multiplexing

Extensive pad multiplexing is used to accommodate the largest number of peripheral functions in the smallest possible package. Pad multiplexing is controlled using a combination of hardware configuration at device reset and software programmable register settings. See the device Data Manual to determine how pad multiplexing affects the McBSP.

11.10.3 McBSP Integration

This section describes module integration in the device, including information about clocks, resets, and hardware requests.

Figure 11-715 shows the integration of the McBSP module in the device.

Figure 11-715. McBSP Integration

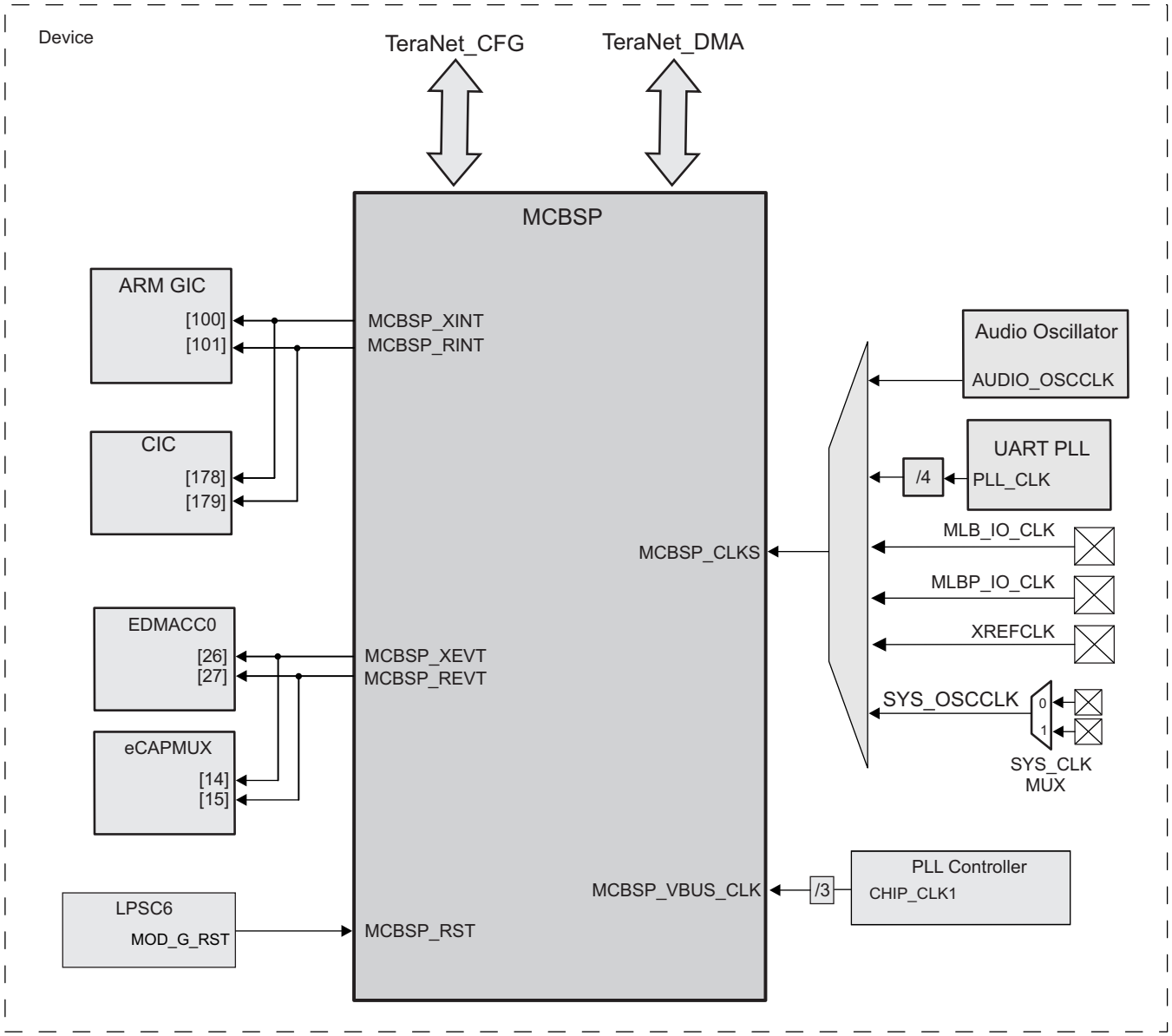


Table 11-1592 through Table 11-1594 summarize the integration of the module in the device.

Table 11-1592. McBSP Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
MCBSP	PD5	LPSC6	TeraNet_CFG (for control and configuration) TeraNet_DMA (for data traffic)

Table 11-1593. McBSP Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
MCBSP	MCBSP_VBUS_CLK	CHIP_CLK1 / 3	PLL Controller	McBSP Interface and Functional clock
	MCBSP_CLKS	AUDIO_OSCCLK	Audio Oscillator	The input clock to the sample rate generator, which can be either the internal clock source or a dedicated external clock source. For more details, see Figure 11-715 *McBSP Integration*.
		MLB_IO_CLK	MLB_CLK pin	
		MLBP_IO_CLK	MLBP_CLK_P and MLBP_CLK_N pins	
		SYS_OSCCLK	SYS_CLK mux	
		XREFCLK	XREFCLK pin	
		UART_PLL_CLK / 4	UART PLL	
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
MCBSP	MCBSP_RST	MOD_G_RST	LPSC6	Module Asynchronous Reset

Table 11-1594. McBSP Hardware Requests

Interrupt Requests				
Module Instance	**Event Name**	**Mapped To Input Event [Number]**		**Description**
		ARM GIC	**CIC**	
MCBSP	MCBSP_XINT	[100]	[178]	McBSP Transmit Interrupt event
	MCBSP_RINT	[101]	[179]	McBSP Receive Interrupt event
DMA Requests				
Module Instance	**Event Name**	**Mapped To Input Event [Number]**		**Description**
		EDMACC0	**eCAPMUX**	
MCBSP	MCBSP_XEVT	[26]	[14]	McBSP transmit event
	MCBSP_REVT	[27]	[15]	McBSP receive event

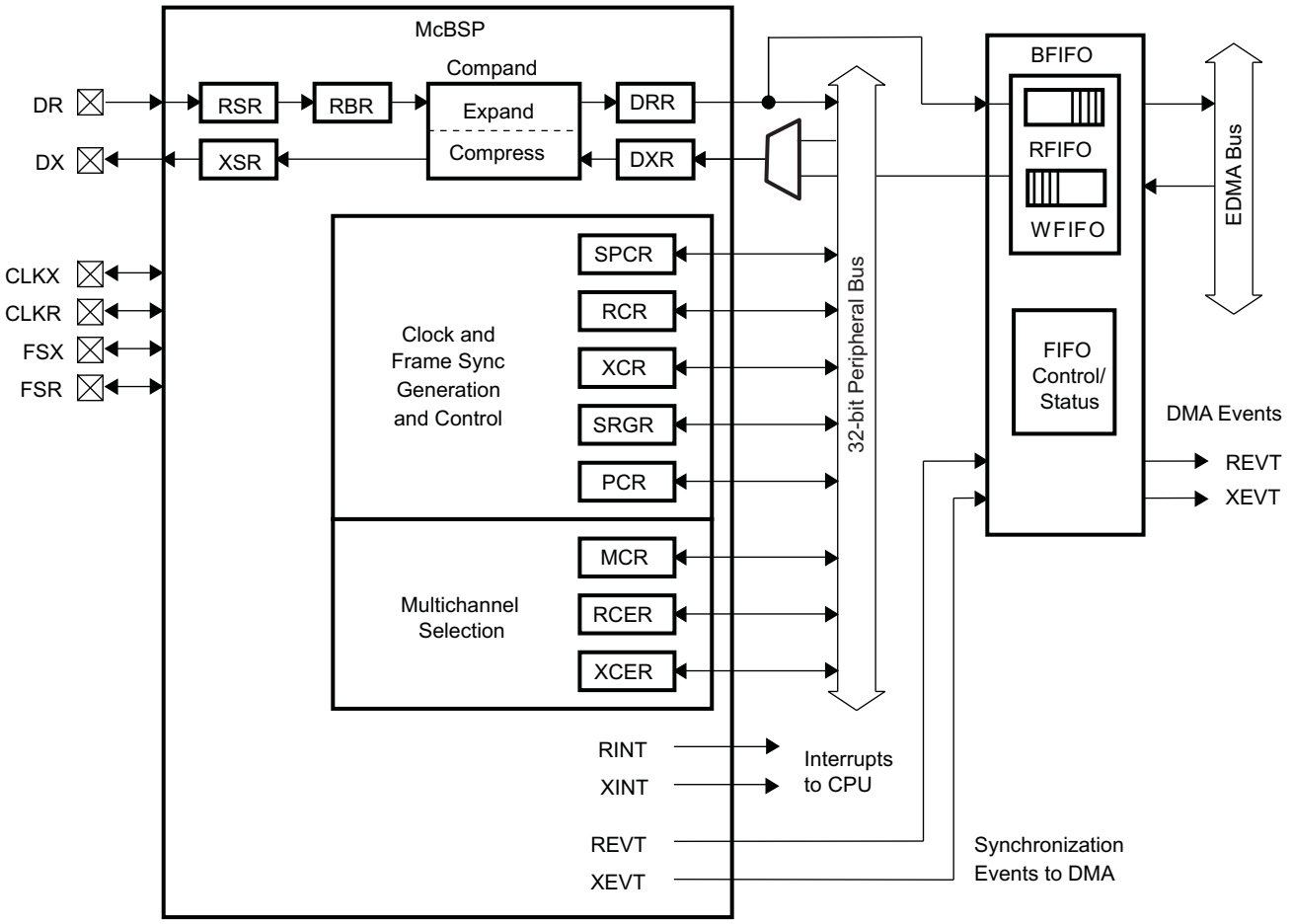
11.10.4 McBSP Functional Description

This section describes the functional description of the McBSP module.

11.10.4.1 McBSP Functional Block Diagram

Figure 11-716 shows the functional block diagram of the McBSP module.

Figure 11-716. McBSP Block Diagram

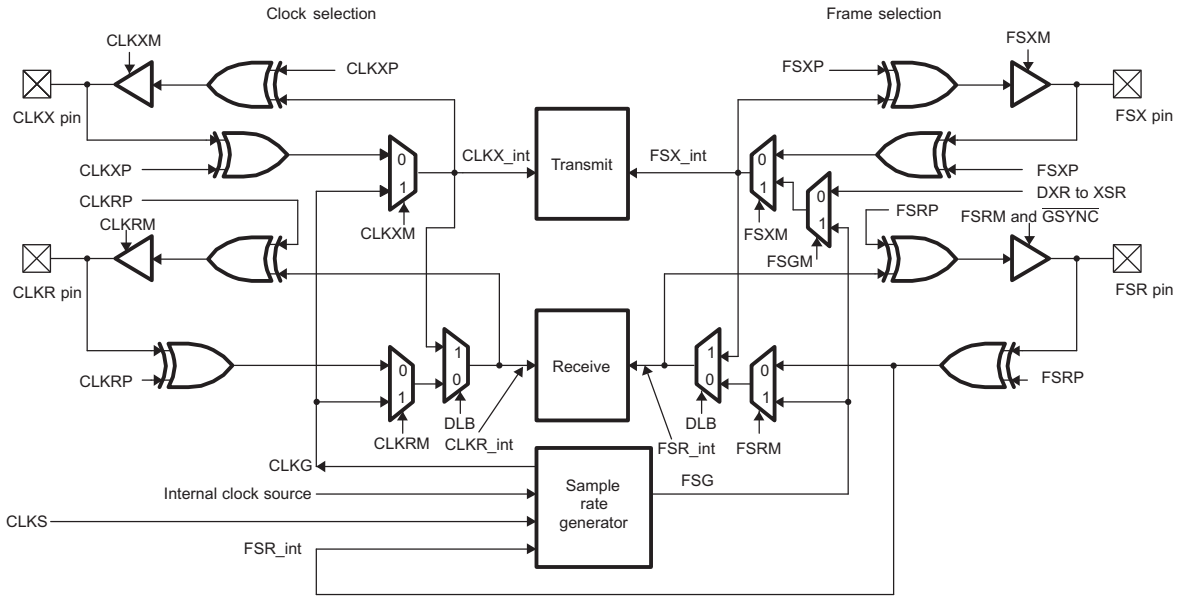


11.10.4.2 Clock, Frames, and Data

The McBSP can use an internal or external clock source. Either clock source can be divided-down inside the McBSP to generate the actual interface bit clock frequency.

The McBSP has several ways of selecting clocking and framing for both the receiver and transmitter. Clocking and framing can be sent to both portions by the sample rate generator. Each portion can select external clocking and/or framing independently. Figure 11-717 is a block diagram of the clock and frame selection circuitry.

Figure 11-717. Clock and Frame Generation



11.10.4.2.1 Frame and Clock Operation

Receive and transmit frame sync pulses (FSR/X), and clocks (CLKR/X), can either be generated internally by the sample rate generator (see Section 11.10.4.2.2) or be driven by an external source. The source of frame sync and clock is selected by programming the mode bits, FS(R/X)M and CLK(R/X)M respectively, in the pin control register (MCBSP_PCR). FSR is also affected by the GSYNC bit in the sample rate generator register (MCBSP_SRGR), see Section 11.10.4.2.4.2 for details.

When FSR and FSX are inputs (FSXM = FSRM = 0), the McBSP detects them on the internal falling edge of clock, CLKR_int and CLKX_int, respectively (see Figure 11-717). The receive data arriving at the DR pin is also sampled on the falling edge of CLKR_int. These internal clock signals are either derived from an external source via the CLK(R/X) pins or driven by the sample rate generator clock (CLKG) internal to the McBSP.

When FSR and FSX are outputs driven by the sample rate generator, they are generated (transition to their active state) on the rising edge of the internal clock, CLK(R/X)_int. Similarly, data on DX is output on the rising edge of CLKX_int.

The FSRP, FSXP, CLKRP, and CLKXP bits in MCBSP_PCR configure the polarities of FSR, FSX, CLKR, and CLKX. All frame sync signals (FSR_int and FSX_int) internal to the serial port are active high. If the serial port is configured for external frame synchronization (FSR/FSX are inputs to the McBSP) and FSRP = FSXP = 1, the external active (low) frame sync signals are inverted before being sent to the receiver signal (FSR_int) and transmitter signal (FSX_int). Similarly, if internal synchronization is selected (FSR/FSX are outputs and GSYNC = 0), the internal active (high) sync signals are inverted if the polarity bit FS(R/X)P = 1, before being sent to the FS(R/X) pin. Figure 11-717 shows this inversion using XOR gates.

On the transmit side, the transmit clock polarity bit, CLKXP, sets the edge used to shift and clock out transmit data. Data is always transmitted on the rising edge of CLKX_int (see Figure 11-718). If CLKXP = 1 and external clocking is selected (CLKXM = 0 and CLKX is an input), the external falling-edge-triggered input clock on CLKX is inverted to a rising-edge-triggered clock before being sent to the transmitter. If CLKXP = 1 and internal clocking is selected (CLKXM = 1 and CLKX is an output pin), the internal (rising-edge-triggered) clock, CLKX_int, is inverted before being sent out on the CLKX pin.

Similarly, the receiver can reliably sample data that is clocked (by the transmitter) with a rising-edge clock. The receive clock polarity bit, CLKRP, sets the edge used to sample received data. The receive data is always sampled on the falling edge of CLKR_int (see Figure 11-719). Therefore, if CLKRP = 1 and external clocking is selected (CLKRM = 0 and CLKR is an input pin), the external rising-edge triggered input clock on CLKR is inverted to a falling-edge clock before being sent to the receiver. If CLKRP = 1 and internal clocking is selected (CLKRM = 1), the internal falling-edge-triggered clock is inverted to a rising edge before being sent out on the CLKR pin.

In a system where the same clock (internal or external) is used to clock the receiver and transmitter, CLKRP = CLKXP. The receiver uses the opposite edge as the transmitter to ensure valid setup and hold times of data around this edge. Figure 11-719 shows how data clocked by an external serial device using a rising-edge clock can be sampled by the McBSP receiver with the falling edge of the same clock.

Figure 11-718. Transmit Data Clocking

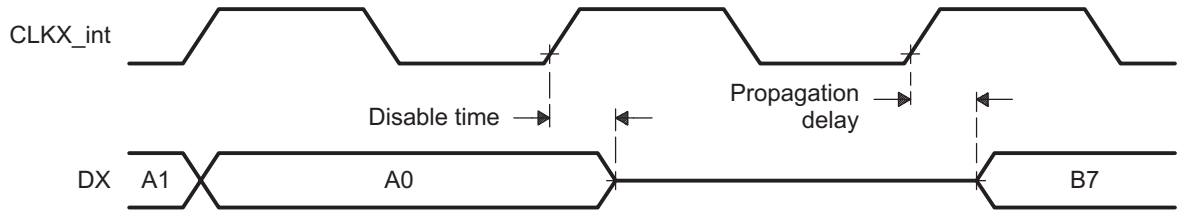
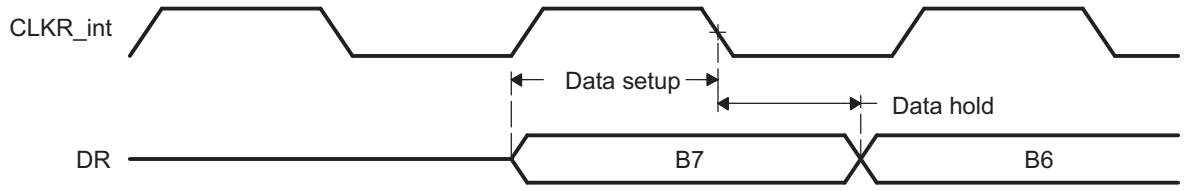


Figure 11-719. Receive Data Clocking



11.10.4.2.2 Sample Rate Generator Clocking and Framing

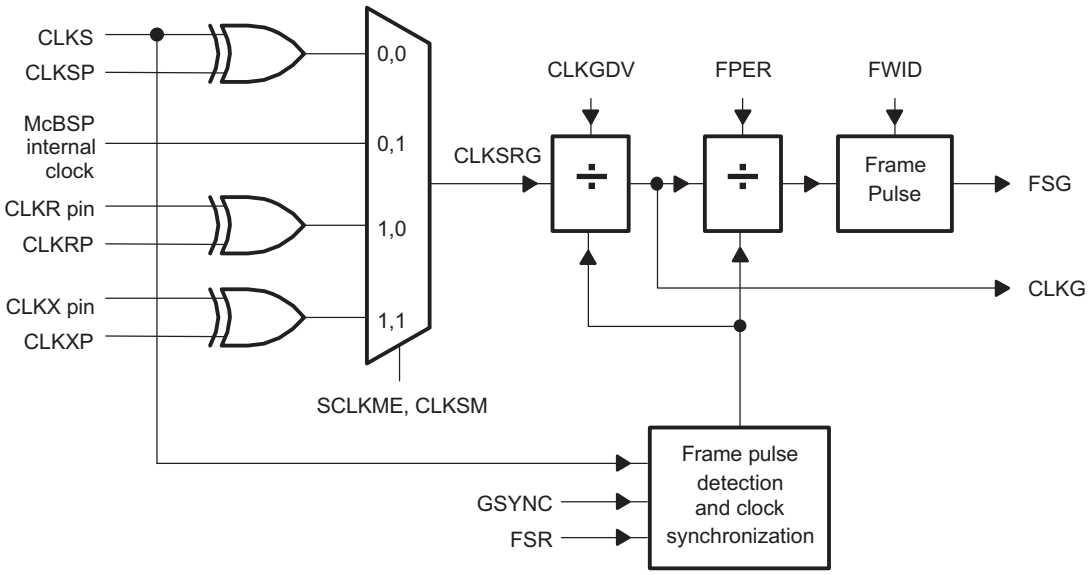
The sample rate generator is composed of a 3-stage clock divider that provides a programmable data clock (CLKG) and framing signal (FSG), as shown in Figure 11-720. CLKG and FSG are McBSP internal signals that can be programmed to drive receive and/or transmit clocking, CLK(R/X), and framing, FS(R/X). The sample rate generator can be programmed to be driven by an internal clock source or an internal clock derived from an external clock source.

The sample rate generator is not used when CLKX, FSX, CLKR, and FSR are driven by an external source. Therefore, the GRST bit in the serial port control register (MCBSP_SPCR) does not need to be enabled (GRST = 1) for this setup. The three stages of the sample rate generator circuit compute:

- Clock divide-down (CLKGDV): The number of input clocks per data bit clock
- Frame period (FPER): The frame period in data bit clocks
- Frame width (FWID): The width of an active frame pulse in data bit clocks

In addition, a frame pulse detection and clock synchronization module allows synchronization of the clock divide-down with an incoming frame pulse. The operation of the sample rate generator during device reset is described in Section 11.10.4.3.1.

Figure 11-720. Sample Rate Generator Block Diagram



11.10.4.2.3 Data Clock Generation

When the receive/transmit clock mode is set to 1 (CLK(R/X)M = 1 in the pin control register (MCBSP_PCR)), the data clocks (CLK(R/X)) are driven by the internal sample rate generator output clock, CLKG. A variety of data bit clocks can be select for the receiver and transmitter including:

- The input clock to the sample rate generator, which can be either the internal clock source or a dedicated external clock source via the CLKX, CLKR pins or CLKS. See Section 11.10.4.2.3.1 for details on the source of the McBSP internal clock.
- The input clock source (internal clock source or external clock CLKX/CLKR/CLKS) to the sample rate generator can be divided-down by a programmable value (CLKGDV bit in the sample rate generator register (MCBSP_SRGR)) to drive CLKG.

Regardless of the source to the sample rate generator, the rising edge of CLKSRG (see Figure 11-720) generates CLKG and FSG.

11.10.4.2.3.1 Input Clock Source Mode: CLKSM and SCLKME

The sample rate generator input clock signal can be driven from one of four sources selectable with the SCLKME bit in the pin control register (MCBSP_PCR) and the CLKSM bit in the sample rate generator register (MCBSP_SRGR), see Table 11-1595.

Table 11-1595. Choosing an Input Clock for the Sample Rate Generator with the SCLKME and CLKSM Bits

SCLKME Bit in MCBSP_PCR	CLKSM Bit in MCBSP_SRGR	Input Clock for Sample Rate Generator
0	0	Signal on CLKS
0	1	McBSP internal input clock
1	0	Signal on CLKR pin
1	1	Signal on CLKX pin

11.10.4.2.3.2 Sample Rate Generator Data Bit Clock Rate: CLKGDV

The first divider stage generates the serial data bit clock from the input clock. This divider stage uses a counter that is preloaded by the CLKGDV bit in the sample rate generator register (MCBSP_SRGR) and that contains the divide ratio value. The output of this stage is the data bit clock that is output on the sample rate generator output, CLKG, and that serves as the input for the second and third divider stages.

CLKG has a frequency equal to 1/(CLKGDV + 1) of the sample rate generator input clock. Thus, the sample rate generator input clock frequency is divided by a value between 1 to 256. The CLKGDV value chosen must result in a clock that meets the timing requirements/limitations specified in the device Data Manual.

When CLKGDV is an odd value or equal to 0, the CLKG duty cycle is 50%. Note that an odd CLKGDV value means an even divide down of the source clock and an even CLKGDV value means an odd divide down of the source clock. When CLKGDV is an even value (2p), the high state duration is p + 1 cycles and the low state duration is p cycles. This is illustrated in Example 11-2, Example 11-3, and Example 11-4.

In the following examples:

S_{IN} = sample generator input clock period

f_{IN} = sample generator input clock frequency

S_G = CLKG period

f_G = CLKG frequency

The following equation is given above:

$f_G = f_{IN} / dd(CLKGDV + 1))$; therefore, $S_G = (CLKGDV + 1) \times S_{IN}$.

Example 11-2. CLKGDV = 0

CLKGDV = 0

$S_G = (CLKGDV + 1) \times S_{IN} = (0 + 1) \times S_{IN} = S_{IN}$

Pulse width high = $S_{IN} \times (CLKGDV + 1) / 2 = S_{IN} \times (0 + 1) / 2 = 0.5 \times S_{IN}$

Pulse width low = $S_{IN} \times (CLKGDV + 1) / 2 = S_{IN} \times (0 + 1) / 2 = 0.5 \times S_{IN}$

Example 11-3. CLKGDV = 1

CLKGDV = 1

$S_G = (CLKGDV + 1) \times S_{IN} = (1 + 1) \times S_{IN} = 2 \times S_{IN}$

Pulse width high = $S_{IN} \times (CLKGDV + 1) / 2 = S_{IN} \times (1 + 1) / 2 = S_{IN}$

Pulse width low = $S_{IN} \times (CLKGDV + 1) / 2 = S_{IN} \times (1 + 1) / 2 = S_{IN}$

Example 11-4. CLKGDV = 2

CLKGDV = 2

$S_G = (CLKGDV + 1) \times S_{IN} = (2 + 1) \times S_{IN} = 3 \times S_{IN}$

Pulse width high = $S_{IN} \times (CLKGDV / 2 + 1) / 2 = S_{IN} \times (2 / 2 + 1) / 2 = 2 \times S_{IN}$

Pulse width low = $S_{IN} \times CLKGDV / 2 = S_{IN} \times 2 / 2 = 1 \times S_{IN}$

11.10.4.2.3.3 Bit Clock Polarity: CLKSP

CLKS is selected to drive the sample rate generator clock divider by selecting CLKSM = 0 in the sample rate generator register (MCBSP_SRGR) and SCLKME = 0 in the pin control register (MCBSP_PCR). In this case, the CLKSP bit in MCBSP_SRGR selects the edge of CLKS on which sample rate generator data bit clock (CLKG) and frame sync signal (FSG) are generated. Since the rising edge of CLKSRG generates CLKG and FSG, the rising edge of CLKS when CLKSP = 0 or the falling edge of CLKS when CLKSP = 1 causes the transition on CLKG and FSG.

11.10.4.2.3.4 Bit Clock and Frame Synchronization

CLKS is selected to drive the sample rate generator (CLKSM = 0 in MCBSP_SRGR and SCLKME = 0 in MCBSP_PCR), the GSYNC bit in MCBSP_SRGR can be used to configure the timing of CLKG relative to CLKS. GSYNC = 1 ensures that the McBSP and the external device to which it is communicating are dividing down the CLKS with the same phase relationship. If GSYNC = 0, this feature is disabled and CLKG runs freely and is not resynchronized. If GSYNC = 1, an inactive-to-active transition on FSR triggers a resynchronization of CLKG and the generation of FSG. CLKG always begins at a high state after synchronization. Also, FSR is always detected at the same edge of CLKS that generates CLKG, regardless of the length of the FSR pulse. Although an external FSR is provided, FSG can still drive internal receive frame synchronization when GSYNC = 1. When GSYNC = 1, FPER does not matter, because the frame period is determined by the arrival of the external frame sync pulse.

Figure 11-721 and Figure 11-722 show this operation with various polarities of CLKS and FSR. These figures assume that FWID is 0, for a FSG = 1 CLKG wide.

These figures show what happens to CLKG when it is initially in sync and GSYNC = 1, as well as when it is not in sync with the frame synchronization and GSYNC = 1.

Figure 11-721. CLKG Synchronization and FSG Generation When GSYNC = 1 and CLKGDV = 1

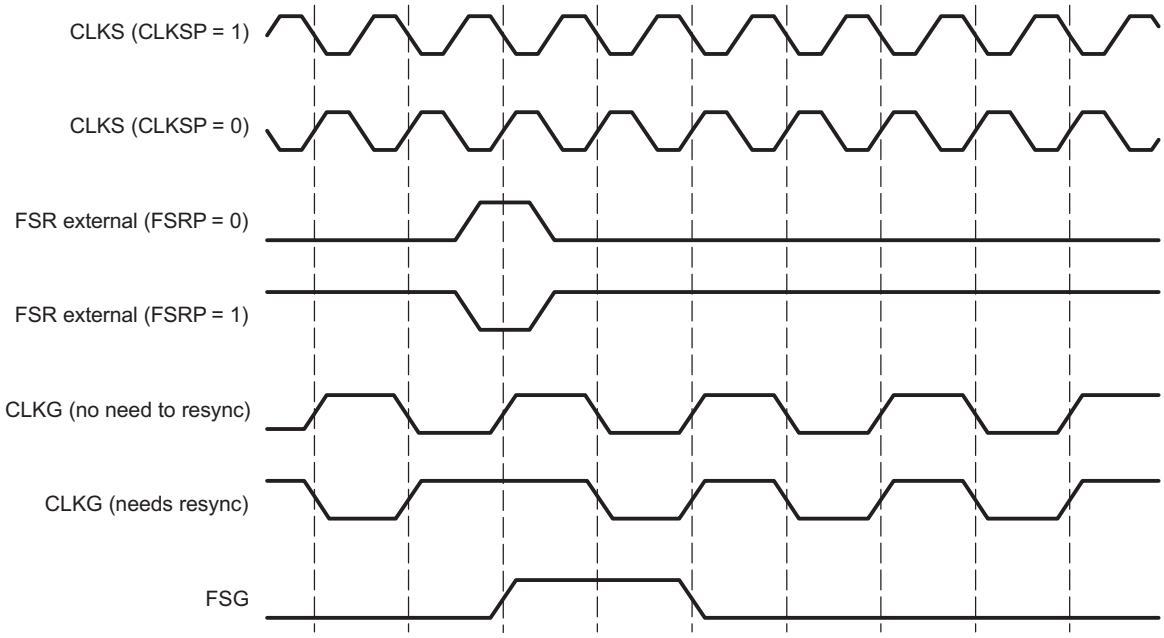
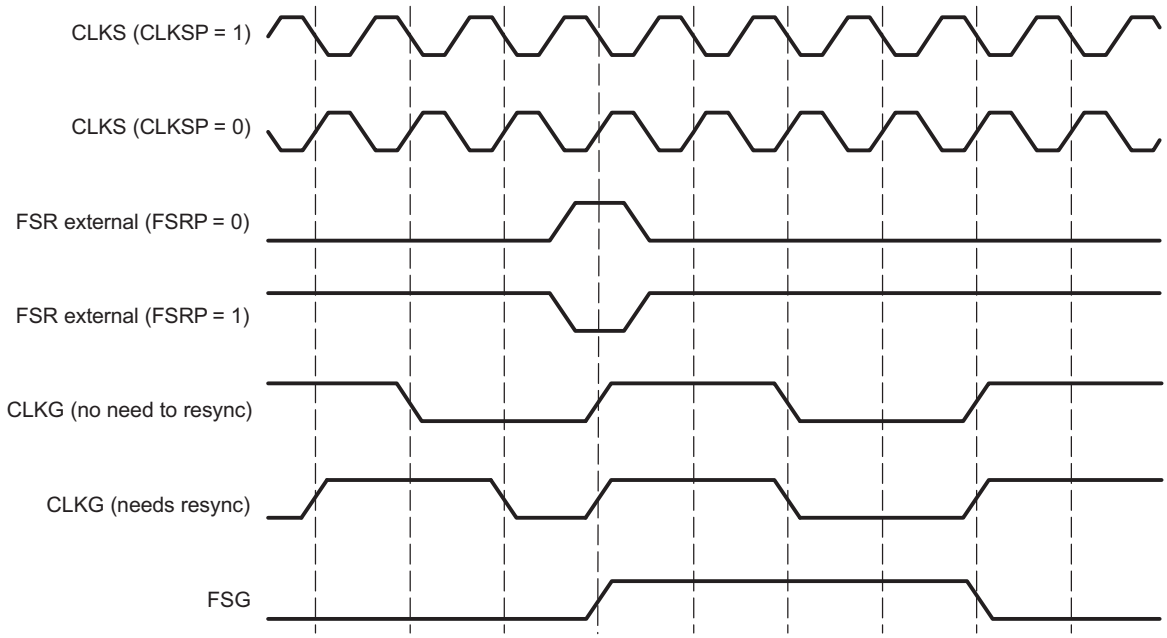


Figure 11-722. CLKG Synchronization and FSG Generation When GSYNC = 1 and CLKGDV = 3



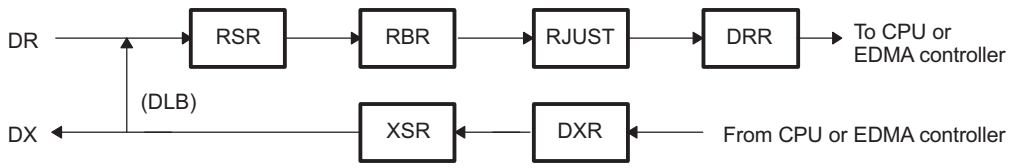
When GSYNC = 1, the transmitter can operate synchronously with the receiver, provided that the following conditions are met:

- FSX is programmed to be driven by the sample rate generator frame sync, FSG (FSGM = 1 in MCBSP_SRGR and FSXM = 1 in MCBSP_PCR).
- The sample-rate generator clock should drive the transmit and receive bit clock (CLK(R/X)M = 1 in MCBSP_SPCR). Therefore, the CLK(R/X) pin should not be driven by any other source.

11.10.4.2.3.5 Digital Loopback Mode: DLB

The DLB mode cannot be used when the McBSP is in clock stop mode (CLKSTP = 2h or 3h in the serial port control register (MCBSP_SPCR)). Setting DLB = 1 in MCBSP_SPCR enables digital loopback mode. In DLB mode, DR, FSR, and CLKR are internally connected through multiplexers to DX, FSX, and CLKX, respectively, as shown in Figure 11-717 and Figure 11-756. DLB mode allows testing of serial port code without using the external interface of the McBSP. CLKX and FSX must be enabled as outputs (CLKXM = FSXM = 1) in DLB mode.

Figure 11-723. Digital Loopback Mode



11.10.4.2.3.6 Receive Clock Selection: DLB, CLKRM

Table 11-1596 shows how the digital loopback bit (DLB) in the serial port control register (MCBSP_SPCR) and the CLKRM bit in the pin control register (MCBSP_PCR) select the receiver clock. In digital loopback mode (DLB = 1), the transmitter clock drives the receiver. CLKRM determines whether the CLKR pin is an input or an output.

Table 11-1596. Receive Clock Selection

DLB Bit in MCBSP_SPCR	CLKRM Bit in MCBSP_PCR	Source of Receive Clock	CLKR Function
0	0	CLKR acts as an input driven by the external clock and inverted as determined by CLKRP before being used.	Input.
0	1	The sample rate generator clock (CLKG) drives CLKR.	Output. CLKG inverted as determined by CLKRP before being driven out on CLKR.
1	0	CLKX_int drives the receive clock CLKR_int as selected and is inverted.	High impedance.
1	1	CLKX_int drives CLKR_int as selected and is inverted.	Output. CLKR (same as CLKX) is inverted as determined by CLKRP before being driven out.

11.10.4.2.3.7 Transmit Clock Selection: CLKXM

Table 11-1597 shows how the CLKXM bit in the pin control register (MCBSP_PCR) selects the transmit clock and whether the CLKX pin is an input or output.

Table 11-1597. Transmit Clock Selection

CLKXM Bit in MCBSP_PCR	Source of Transmit Clock	CLKX Function
0	The external clock drives the CLKX input pin. CLKX is inverted as determined by CLKXP before being used.	Input.
1	The sample rate generator clock (CLKG) drives the transmit clock.	Output.CLKG is inverted as determined by CLKXP before being driven out on CLKX.

11.10.4.2.3.8 Stopping Clocks

Two methods can be used to stop serial clocks between data transfers:

- The SPI CLKSTP mode where clocks are stopped between single-element transfers. This mode is not supported on this device.
- The clocks are inputs to the McBSP (CLKXM or CLKRM = 0 in the pin control register (MCBSP_PCR)) and the McBSP operates in non-SPI mode (the clocks can be stopped between data transfers). If the external device stops the serial clock between data transfers, the McBSP interprets it as a slowed-down serial clock. Ensure that there are no glitches on the CLK(R/X) lines as the McBSP may interpret them as clock-edge transitions. Restarting the serial clock is equivalent to a normal clock transition after a slow CLK(R/X) cycle. Note that just as in normal operations, transmit under flow (XEMPTY) may occur if the MCBSP_DXR is not properly serviced at least three CLKX cycles before the next frame sync. Therefore, if the serial clock is stopped before MCBSP_DXR is properly serviced, the external device needs to restart the clock at least three CLKX cycles before the next frame sync to allow the MCBSP_DXR write to be properly synchronized. See Figure 11-744 for a graphical explanation on when MCBSP_DXR needs to be written to avoid underflow.

11.10.4.2.4 Frame Sync Generation

Data frame synchronization is independently programmable for the receiver and the transmitter for all data delay values. When the FRST bit in the serial port control register (MCBSP_SPCR) is set to 1 the frame generation logic is activated to generate frame sync signals, provided that FSGM = 1 in the sample rate generator register (MCBSP_SRGR). The frame sync programming options are:

- A frame pulse with a programmable period between sync pulses and a programmable active width specified in MCBSP_SRGR.
- The transmitter can trigger its own frame sync signal that is generated by a DXR-to-XSR copy. This causes a frame sync to occur on every DXR-to-XSR copy. The data delays can be programmed as required. However, maximum packet frequency cannot be achieved in this method for data delays of 1 and 2.

- Both the receiver and transmitter can independently select an external frame synchronization on the FSR and FSX pins, respectively.

11.10.4.2.4.1 Frame Period (FPER) and Frame Width (FWID)

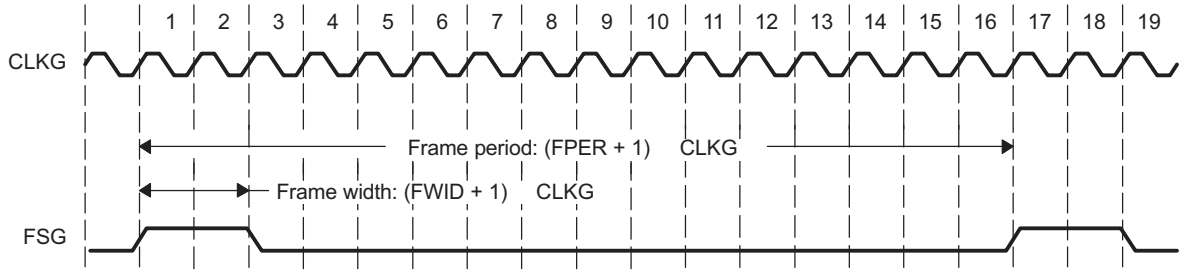
The FPER bits in the sample rate generator register (MCBSP_SRGR) are a 12-bit down-counter that can count down the generated data clocks from 4095 to 0. FPER controls the period of active frame sync pulses. The FWID bits in MCBSP_SRGR are an 8-bit down-counter. FWID controls the active width of the frame sync pulse.

When the sample rate generator comes out of reset, FSG is in an inactive (low) state. After this, when FRST = 1 in MCBSP_SPCR and FSGM = 1 in MCBSP_SRGR, frame sync signals are generated. The frame width value (FWID + 1) is counted down on every CLKG cycle until it reaches 0 when FSG goes low. Thus, the value of FWID + 1 determines an active frame pulse width ranging from 1 to 256 data bit clocks.

At the same time, the frame period value (FPER + 1) is also counting down, and when this value reaches 0, FSG goes high again indicating a new frame is beginning. Thus, the value of FPER + 1 determines a frame length from 1 to 4096 data bits. When GSYNC = 1 in MCBSP_SRGR, the value of FPER does not matter. Figure 11-724 shows a frame of 16 CLKG periods (FPER = 15 or 0000 1111b).

It is recommended that FWID be programmed to a value less than (R/X)WDLEN1/2.

Figure 11-724. Programmable Frame Period and Width



11.10.4.2.4.2 Receive Frame Synchronization Selection: DLB and FSRM

Table 11-1598 shows how to select various sources to provide the receive frame synchronization signal. Note that in digital loopback mode (DLB = 1 in the serial port control register (MCBSP_SPCR)), the transmit frame sync signal is used as the receive frame sync signal and that DR is internally connected to DX.

NOTE: FSR_int and FSX_int are shown in Figure 11-717.

Table 11-1598. Receive Frame Synchronization Selection

DLB Bit in MCBSP_SPCR	FSRM Bit in MCBSP_PCR	GSYNC Bit in MCBSP_SRGR	Source of Receive Frame Synchronization	FSR Pin Function
0	0	X	External frame sync signal drives the FSR input pin, whose signal is then inverted as determined by FSRP before being used as FSR_int.	Input.
0	1	0	Sample rate generator frame sync signal (FSG) drives FSR_int, FRST = 1.	Output. FSG is inverted as determined by FSRP before being driven out on the FSR pin.
0	1	1	Sample rate generator frame sync signal (FSG) drives FSR_int, FRST = 1.	Input. The external frame sync input on FSR is used to synchronize CLKG and generate FSG.
1	0	0	FSX_int drives FSR_int. FSX is selected as shown in Table 11-1599.	High impedance.

Table 11-1598. Receive Frame Synchronization Selection (continued)

DLB Bit in MCBSP_SPCR	FSRM Bit in MCBSP_PCR	GSYNC Bit in MCBSP_SRGR	Source of Receive Frame Synchronization	FSR Pin Function
1	X	1	FSX_int drives FSR_int and is selected as shown in Table 11-1599.	Input. External FSR is not used for frame synchronization but is used to synchronize CLKG and generate FSG since GSYNC = 1.
1	1	0	FSX_int drives FSR_int and is selected as shown in Table 11-1599.	Output. Receive (same as transmit) frame synchronization is inverted as determined by FSRP before being driven out.

11.10.4.2.4.3 Transmit Frame Synchronization Selection: FSXM and FSGM

Table 11-1599 shows how to select the source of the transmit frame synchronization signal. The three choices are:

- External frame sync input
- The sample rate generator frame sync signal (FSG)
- A signal that indicates a DXR-to-XSR copy has been made

NOTE: FSR_int and FSX_int are shown in Figure 11-717.

Table 11-1599. Transmit Frame Synchronization Selection

FSXM Bit in MCBSP_PCR	FSGM Bit in MCBSP_SRGR	Source of Transmit Frame Synchronization	FSX Pin Function
0	X	External frame sync input on the FSX pin. This is inverted by FSXP before being used as FSX_int.	Input
1	1	Sample rate generator frame sync signal (FSG) drives FSX_int. FRST = 1.	Output. FSG is inverted by FSXP before being driven out on FSX.
1	0	A DXR-to-XSR copy activates transmit frame sync signal.	Output. 1-bit-clock-wide signal inverted as determined by FSXP before being driven out on FSX.

11.10.4.2.4.4 Frame Detection

To facilitate detection of frame synchronization, the receive and transmit CPU interrupts (RINT and XINT) can be programmed to detect frame synchronization by setting the RINTM and XINTM bits in the serial port control register (MCBSP_SPCR) to 10b. The associated portion (receiver/transmitter) of the McBSP must be out of reset.

11.10.4.2.5 Data and Frames

All data transfers require a valid frame synchronization.

11.10.4.2.5.1 Frame Synchronization Phases

Frame synchronization indicates the beginning of a transfer on the McBSP. The data stream following frame synchronization can have up to two phases: phase 1 and phase 2. The number of phases can be selected by the phase bit, (R/X)PHASE, in MCBSP_RCR and MCBSP_XCR. The number of elements per frame and bits per element can be independently selected for each phase via (R/X)FRLEN1/2 and (R/X)WDLEN1/2, respectively.

Figure 11-725 shows a frame in which the first phase consists of two elements of 12 bits, each followed by a second phase of three elements of 8 bits each. The entire bit stream in the frame is contiguous, no gaps exist either between elements or phases. Table 11-1600 shows the fields in the receive/transmit control registers (MCBSP_RCR/MCBSP_XCR) that control the frame length and element length for each phase for both the receiver and the transmitter. The maximum number of elements per frame is 128 for a single-phase frame and 256 elements in a dual-phase frame. The number of bits per element can be 8, 12, 16, 20, 24, or 32.

NOTE: For a dual-phase frame with internally generated frame synchronization, the maximum number of elements per phase depends on the word length. This is because the frame period, FPER, is only 12-bits wide and, therefore, provides 4096 bits per frame. Hence, the maximum number of 256 elements per dual-phase frame applies only when the WDLEN is 16 bits. However, any combination of element numbers and element size (defined by the FRLEN and WDLEN bits, respectively) is valid as long as their product is less than or equal to 4096 bits. This limitation does not apply for dual-phase with external frame sync.

Figure 11-725. Dual-Phase Frame Example

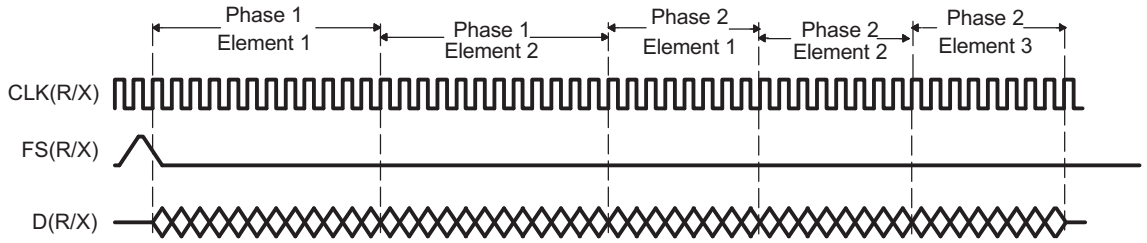


Table 11-1600. MCBSP_RCR/MCBSP_XCR Fields Controlling Elements per Frame and Bits per Element

Serial Port	Frame Phase	MCBSP_RCR/MCBSP_XCR Field Control	
		Elements per Frame	Bits per Element
Receive	1	RFRLEN1	RWDLEN1
Receive	2	RFRLEN2	RWDLEN2
Transmit	1	XFRLEN1	XWDLEN1
Transmit	2	XFRLEN2	XWDLEN2

11.10.4.2.5.2 Frame Length: RFRLEN1/2 and XFRLEN1/2

Frame length specifies the maximum number of serial elements or logical time slots or channels that are available for transfer per frame synchronization signal. In multi-channel selection mode, the frame length value is independent of, and possibly different from, the actual number of channels that the device is programmed to receive or transmit per frame via the MCBSP_MCR, RCEREn, and XCEREn registers. See Section 11.10.4.7 for details on multi-channel selection mode operation.

The 7-bit (R/X)FRLEN1/2 bits in (R/X)CR support up to 128 elements per phase in a frame, as shown in Table 11-1601. (R/X)PHASE = 0 selects a single-phase data frame, and (R/X)PHASE = 1 selects a dual-phase frame for the data stream. For a single-phase frame, the value of (R/X)FRLEN2 does not matter. Program the frame length fields with (w minus 1), where w represents the number of elements per frame. For Figure 11-725, (R/X)FRLEN1 = 1 or 000 0001b and (R/X)FRLEN2 = 2 or 000 0010b.

Table 11-1601. Receive/Transmit Frame Length Configuration

(R/X)PHASE	(R/X)FRLEN1	(R/X)FRLEN2	Frame Length
0	0 ≤ n ≤ 127	x	Single-phase frame; (n + 1) elements per frame
1	0 ≤ n ≤ 127	0 ≤ m ≤ 127	Dual-phase frame; (n + 1) plus (m + 1) elements per frame

11.10.4.2.5.3 Element Length: RWDLEN1/2 and XWDLEN1/2

The (R/X)WDLEN1/2 fields in the receive/transmit control register (MCBSP_RCR and MCBSP_XCR) determine the element length in bits per element for the receiver and the transmitter for each phase of the frame, as indicated in Table 11-1600. Table 11-1602 shows how the value of these fields selects particular element lengths in bits. For the example in Figure 11-725, (R/X)WDLEN1 = 001b and (R/X)WDLEN2 = 000b. If (R/X)PHASE = 0, indicating a single-phase frame, then (R/X)WDLEN2 is not used by the McBSP and its value does not matter.

Table 11-1602. Receive/Transmit Element Length Configuration

(R/X)WDLEN1/2	Element Length (Bits)
000	8
001	12
010	16
011	20
100	24
101	32
110	Reserved
111	Reserved

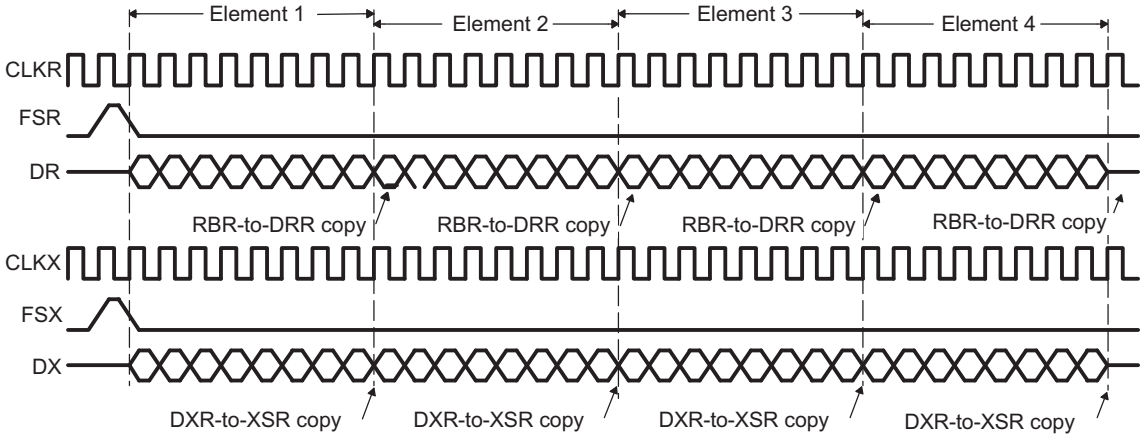
11.10.4.2.5.4 Data Packing Using Frame Length and Element Length

The frame length and element length can be manipulated to effectively pack data. For example, consider a situation in which four 8-bit elements are transferred in a single-phase frame, as shown in Figure 11-726. In this case:

- (R/X)PHASE = 0, indicating a single-phase frame
- (R/X)FRLEN1 = 000 0011b, indicating a 4-element frame
- (R/X)WDLEN1 = 000b, indicating 8-bit elements

In Figure 11-726 four 8-bit data elements are transferred to and from the McBSP by the CPU or the EDMA controller. Four reads of MCBSP_DRR and four writes of MCBSP_DXR are necessary for each frame.

Figure 11-726. Single-Phase Frame of Four 8-Bit Elements

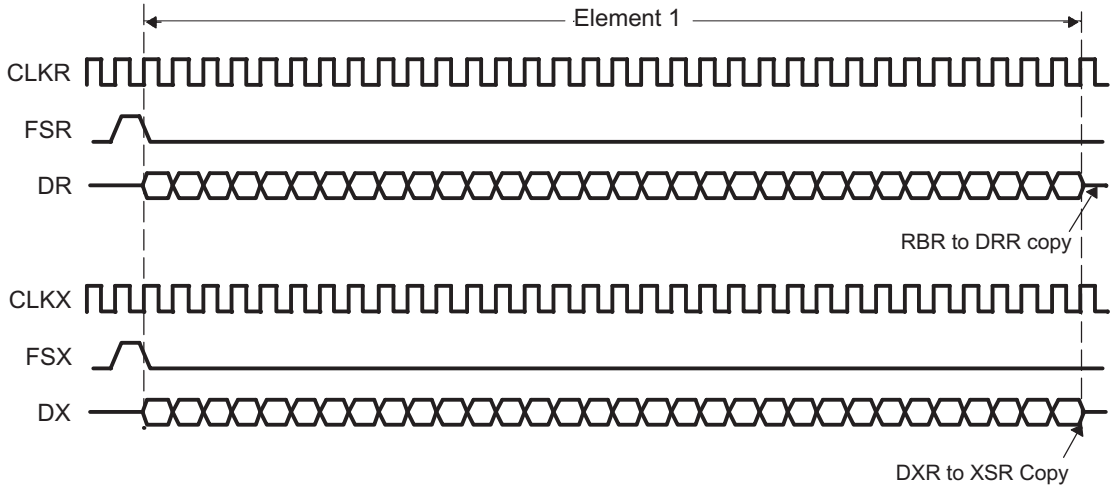


The example in Figure 11-726 can also be viewed as a data stream of a single-phase frame of one 32-bit data element, as shown in Figure 11-727. In this case:

- (R/X)PHASE = 0, indicating a single phase frame
- (R/X)FRLEN1 = 0, indicating a 1-element frame
- (R/X)WDLEN1 = 101b, indicating 32-bit elements

In Figure 11-727, one 32-bit data element is transferred to and from the McBSP by the CPU or the EDMA controller. Thus, one read of MCBSP_DRR and one write of MCBSP_DXR is necessary for each frame. As a result, the number of transfers is one-fourth that of the previous example (Figure 11-726). This manipulation reduces the percentage of bus time required for serial port data movement.

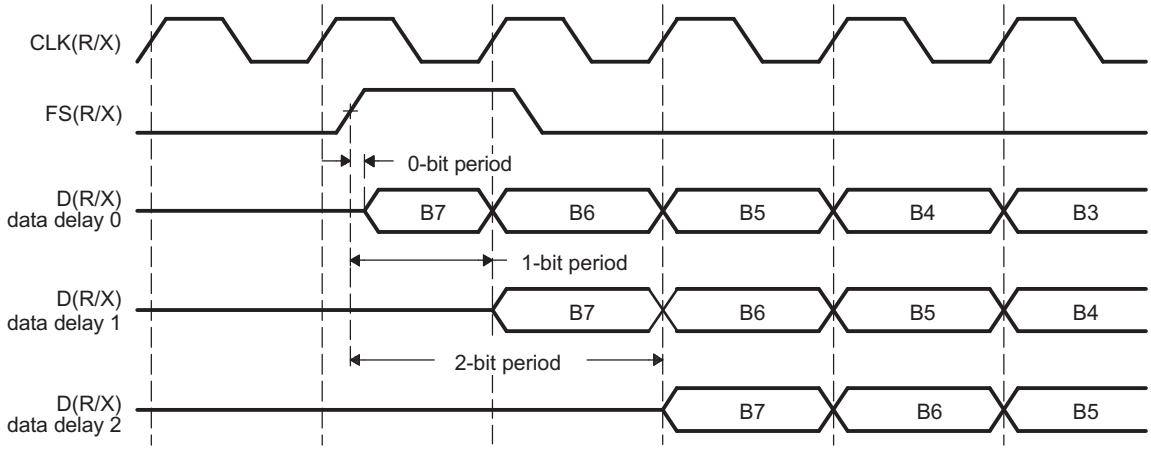
Figure 11-727. Single-Phase Frame of One 32-Bit Element



11.10.4.2.5.5 Data Delay: RDATDLY and XDATDLY

The start of a frame is defined by the first clock cycle in which frame synchronization is active. The beginning of actual data reception or transmission with respect to the start of the frame can be delayed if required. This delay is called data delay. RDATDLY (in MCBSP_RCR) and XDATDLY (in MCBSP_XCR) specify the data delay for reception and transmission, respectively. The range of programmable data delay is zero to two bit clocks ((R/X)DATDLY = 00b to10b), as shown in Figure 11-728. Typically, a 1-bit delay is selected because data often follows a 1-cycle active frame sync pulse.

Figure 11-728. Data Delay



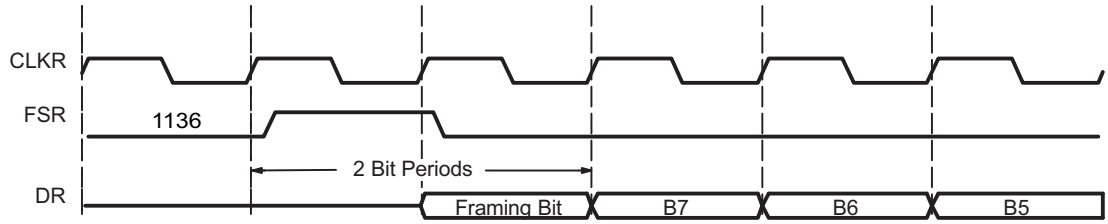
Normally, a frame sync pulse is detected or sampled with respect to an edge of serial clock CLK(R/X). Thus, on a subsequent cycle (depending on data delay value), data can be received or transmitted. However, in the case of a 0-bit data delay, the data must be ready for reception and/or transmission on the same serial clock cycle. For reception, this problem is solved by receive data being sampled on the first falling edge of CLKR when an active (high) FSR is detected.

However, data transmission must begin on the rising edge of CLKX that generated the frame synchronization. Therefore, the first data bit is assumed to be in the XSR and DX. The transmitter then asynchronously detects the frame synchronization, FSX goes active, and it immediately starts driving the first bit to be transmitted on the DX pin.

Another common operation uses a data delay of 2. This configuration allows the serial port to interface to different types of T1 framing devices in which the data stream is preceded by a framing bit. During the reception of such a stream with a data delay of two bits, the framing bit appears after a 1-bit delay and data appears after a 2-bit delay).

The serial port essentially discards the framing bit from the data stream, as shown in Figure 11-729. In transmission, by delaying the first transfer bit, the serial port essentially inserts a blank period (high-impedance period) in place of the framing bit. Here, it is expected that the framing device inserts its own framing bit or that the framing bit is generated by another device. Alternatively, the desired value may be achieved by pull up or pull down DX.

Figure 11-729. 2-Bit Data Delay Used to Discard Framing Bit



11.10.4.2.5.6 Receive Data Justification and Sign Extension: RJUST

The RJUST bit in the serial port control register (MCBSP_SPCR) selects whether data in RBR is right-or left-justified (with respect to the MSB) in the MCBSP_DRR. If right justification is selected, RJUST further selects whether the data is sign-extended or zero-filled. Table 11-1603 and Table 11-1604 summarize the effect that various values of RJUST have on example receive data.

Table 11-1603. Effect of RJUST Bit Values With 12-Bit Example Data ABCh

RJUST Bit in MCBSP_SPCR	Justification	Extension	Value in MCBSP_DRR
00	Right	Zero-fill MSBs	0000 0ABCh
01	Right	Sign-extend MSBs	FFFF FABCh
10	Left	Zero-fill LSBs	ABC0 0000h
11	Reserved	Reserved	Reserved

Table 11-1604. Effect of RJUST Bit Values With 20-Bit Example Data ABCDEh

RJUST Bit in MCBSP_SPCR	Justification	Extension	Value in MCBSP_DRR
00	Right	Zero-fill MSBs	000A BCDEh
01	Right	Sign-extend MSBs	FFFA BCDEh
10	Left	Zero-fill LSBs	ABCD E000h
11	Reserved	Reserved	Reserved

11.10.4.2.5.7 32-Bit Reversal: RWDREVRS, XWDREVRS

Normally all transfers are sent and received with the MSB first. However, the receive/transmit bit ordering of a 32-bit element (LSB first) using the 32-bit reversal feature of the McBSP can be reversed by setting all of the following:

- (R/X)WDREVRS = 1 in the receive/transmit control register (MCBSP_RCR/MCBSP_XCR).
- (R/X)COMPAND = 01b in MCBSP_RCR/MCBSP_XCR.
- (R/X)WDLEN(1/2) = 101b in MCBSP_RCR/MCBSP_XCR to indicate 32-bit elements.

When the register fields are set as above, the bit ordering of the 32-bit element is reversed before being received by or sent from the serial port. If the (R/W)WDREVRS and (R/X)COMPAND fields are set as above, but the element size is not set to 32-bit, operation is undefined.

11.10.4.3 McBSP Resets

11.10.4.3.1 Resetting the Serial Port: RRST, XRST, GRST, and RESET

Device reset or McBSP reset: When the McBSP is reset by device reset or McBSP reset, the state machine is reset to its initial state. All counters and status bits are reset. This includes the receive status bits RFULL, RRDY, and RSYNCERR, and the transmit status bits XEMPTY, XRDY, and XSYNCERR in the serial port control register (MCBSP_SPCR).

The serial port can be reset in the following two ways:

- Device reset (RESET pin is low) places the receiver, the transmitter, and the sample rate generator in reset. When the device reset is removed (RESET = 1), FRST = GRST = RRST = XRST = 0 in MCBSP_SPCR, keeping the entire serial port in the reset state.
- The serial port transmitter and receiver can be independently reset by the XRST and RRST bits in MCBSP_SPCR. The sample rate generator is reset by the GRST bit in MCBSP_SPCR.

Table 11-1605 shows the state of the McBSP pins when the serial port is reset by these methods.

Table 11-1605. Reset State of McBSP Pins

Pin	Direction	Device Reset (RESET = 0)	McBSP Reset
DR	I	Input	**Receiver Reset (RRST = 0 and GRST = 1)**
			Input
CLKR	I/O/Z	Input	Known state if input; CLKR if output
FSR	I/O/Z	Input	Known state if input; FSRP (inactive state) if output
DX	O/Z	High impedance	**Transmitter Reset (XRST = 0 and GRST = 1)**
			High impedance
CLKX	I/O/Z	Input	Known state if input; CLKX if output
FSX	I/O/Z	Input	Known state if input; FSXP (inactive state) if output

11.10.4.3.1.1 Software Reset Considerations

McBSP reset: When the receiver and transmitter reset bits, RRST and XRST in MCBSP_SPCR, are written with 0, the respective portions of the McBSP are reset and activity in the corresponding section stops. All input-only pins, such as DR, and all other pins that are configured as inputs are in a known state. FS(R/X) is driven to its inactive state (same as its polarity bit, FS(R/X)P) if it is an output. If CLK(R/X) are programmed as outputs, they are driven by CLKG, provided that GRST = 1. The DX pin is in the high-impedance state when the transmitter is reset.

During normal operation, the sample rate generator can be reset by writing a 0 to GRST. The sample rate generator should only be reset when not being used by the transmitter or the receiver. In this case, the internal sample rate generator clock CLKG, and its frame sync signal (FSG) is driven inactive (low). When the sample rate generator is not in the reset state (GRST = 1), FSR and FSX are in an inactive state when RRST = 0 and XRST = 0, respectively, even if they are outputs driven by FSG. This ensures that when only one portion of the McBSP is in reset, the other portion can continue operation when FRST = 1 and frame sync is driven by FSG.

Sample-rate generator reset: The sample rate generator is reset when the device is reset or when its reset bit, GRST in MCBSP_SPCR, is written with 0.

Emulator software reset: In the event of an emulator software reset initiated from the DSP, the McBSP register values are reset to their default values.

11.10.4.3.1.2 Hardware Reset Considerations

When the McBSP is reset due to device reset, the entire serial port (including the transmitter, receiver, and the sample rate generator) is reset. All input-only pins and 3-state pins should be in a known state. The output-only pin, DX, is in the high-impedance state. When the device is pulled out of reset, the serial port remains in the reset condition (RRST = XRST = FRST = GRST = 0 in MCBSP_SPCR).

11.10.4.4 Interrupt Support

The McBSP can send both receive and transmit interrupts to the device interrupt controller. For more details, see the Interrupt Controller (INTC) section.

11.10.4.4.1 Interrupt Events and Requests

The RRDY and XRDY bits in the serial port control register (MCBSP_SPCR) indicate the ready state of the McBSP receiver and transmitter, respectively. Writes and reads from the serial port can be synchronized by any of the following methods:

- Polling RRDY and XRDY bits in MCBSP_SPCR
- Using the events sent to the EDMA controller (REVT and XEVT)
- Using the interrupts to the CPU (RINT and XINT) that the events generate

Reading MCBSP_DRR and writing to MCBSP_DXR affects RRDY and XRDY, respectively.

11.10.4.4.1.1 Interrupt Events: RINT and XINT

The receive interrupt (RINT) and transmit interrupt (XINT) signals inform the DSP of changes to the serial port status. Three options exist for configuring these interrupts. These options are set by the receive/transmit interrupt mode bits (RINTM and XINTM) in MCBSP_SPCR. The possible values of the mode, and the configurations they represent, are:

- (R/X)INTM = 00b: Interrupt on every serial element by tracking the (R/X)RDY bits in MCBSP_SPCR.
- (R/X)INTM = 01b: Interrupt at the end of a subframe (16 elements or less) within a frame. See Section 11.10.4.7.9 for more details.
- (R/X)INTM = 10b: Interrupt on detection of frame synchronization pulses. The associated portion (receiver/transmitter) of the McBSP must be out of reset.
- (R/X)INTM = 11b: Interrupt on frame synchronization error. Note that if any of the other interrupt modes are selected, (R/X)SYNCERR may be read when servicing the interrupts to detect this condition. See Section 11.10.4.6.5.2 and Section 11.10.4.6.5.5 for more details on synchronization error.

11.10.4.4.1.2 Receive Ready Status: RINT and RRDY

RRDY = 1 indicates that the RBR contents have been copied to MCBSP_DRR and that the data can now be read by either the CPU or the EDMA controller. Once that data has been read by either the CPU, RRDY is cleared to 0. Also, at device reset or serial port receiver reset (RRST = 0), the RRDY bit is cleared to 0 to indicate that no data has been received and loaded into MCBSP_DRR. RRDY directly drives the McBSP receive interrupt (RINT) to the CPU if RINTM = 00b (default value) in MCBSP_SPCR.

11.10.4.4.1.3 Transmit Ready Status: XINT and XRDY

XRDY = 1 indicates that the MCBSP_DXR contents have been copied to XSR and that MCBSP_DXR is ready to be loaded with a new data word. When the transmitter transitions from reset to non-reset (XRST transitions from 0 to 1), XRDY also transitions from 0 to 1 indicating that MCBSP_DXR is ready for new data. Once new data is loaded by the CPU, the XRDY bit is cleared to 0. However, once this data is copied from MCBSP_DXR to XSR, the XRDY bit transitions again from 0 to 1. The CPU can write to MCBSP_DXR although XSR has not yet been shifted out on DX. XRDY directly drives the McBSP transmit interrupt (XINT) to the CPU if XINTM = 00b (default value) in MCBSP_SPCR.

NOTE: If the polling method is used to service the transmitter, the CPU should wait for one McBSP bit clock (CLKX) before polling again to write the next element in MCBSP_DXR. This is because XRDY transitions occur based on bit clock and not CPU clock. The CPU clock is much faster and can cause false XRDY status, leading to data errors due to overwrites.

11.10.4.5 EDMA Event Support

There are two types of events supported: transmit and receive.

11.10.4.5.1 Receive Ready Status: REVT and RRDY

RRDY = 1 in the serial port control register (MCBSP_SPCR) indicates that the RBR contents have been copied to MCBSP_DRR and that the data can now be read by the EDMA controller. Once that data has been read by the EDMA controller, RRDY is cleared to 0. Also, at device reset or serial port receiver reset (RRST = 0 in MCBSP_SPCR), the RRDY bit is cleared to 0 to indicate that no data has been received and loaded into MCBSP_DRR. RRDY directly drives the McBSP receive event to the EDMA controller (via REVT).

11.10.4.5.2 Transmit Ready Status: XEVT and XRDY

XRDY = 1 in the serial port control register (MCBSP_SPCR) indicates that the MCBSP_DXR contents have been copied to XSR and that MCBSP_DXR is ready to be loaded with a new data word. When the transmitter transitions from reset to non-reset (XRST transitions from 0 to 1 in MCBSP_SPCR), XRDY also transitions from 0 to 1 indicating that MCBSP_DXR is ready for new data. Once new data is loaded by the EDMA controller, the XRDY bit is cleared to 0.

However, once this data is copied from MCBSP_DXR to XSR, the XRDY bit transitions again from 0 to 1. The EDMA controller can write to MCBSP_DXR although XSR has not yet been shifted out on DX. XRDY directly drives the transmit synchronization event to the EDMA controller (via XEVT).

NOTE: If the polling method is used to service the transmitter, the CPU should wait for one McBSP bit clock (CLKX) before polling again to write the next element in MCBSP_DXR. This is because XRDY transitions occur based on bit clock and not CPU clock. The CPU clock is much faster and can cause false XRDY status, leading to data errors due to overwrites.

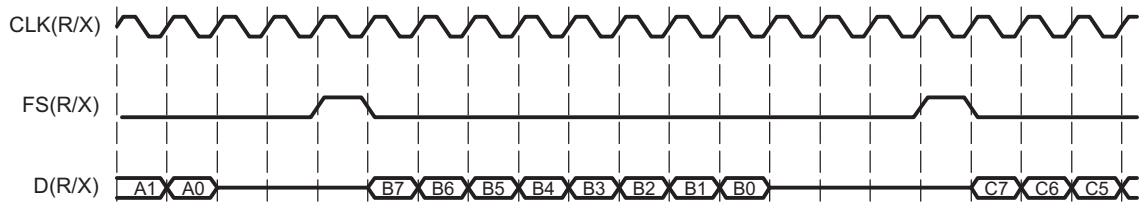
11.10.4.6 McBSP Standard Operation

During a serial transfer, there are typically periods of serial port inactivity between packets or transfers. The receive and transmit frame synchronization pulse occurs for every serial transfer. When the McBSP is not in the reset state and has been configured for the desired operation, a serial transfer can be initiated by programming (R/X)PHASE = 0 for a single-phase frame with the required number of elements programmed in (R/X)FRLEN1. The number of elements can range from 1 to 128 ((R/X)FRLEN1 = 00h to 7Fh). The required serial element length is set in the (R/X)WDLEN1 field in the (R/X)CR. If a dual-phase frame is required for the transfer, RPHASE = 1 and each (R/X)FRLEN1/2 can be set to any value between 0h and 7Fh.

Figure 11-730 shows a single-phase data frame of one 8-bit element. Since the transfer is configured for a 1-bit data delay, the data on the DX and DR pins are available one bit clock after FS(R/X) goes active. This figure, as well as all others in this section, use the following assumptions:

- (R/X)PHASE = 0, specifying a single-phase frame
- (R/X)FRLEN1 = 0b, specifying one element per frame
- (R/X)WDLEN1 = 000b, specifying eight bits per element
- (R/X)FRLEN2 = (R/X)WDLEN2 = Value is ignored
- CLK(R/X)P = 0, specifying that the receive data is clocked on the falling edge and that transmit data is clocked on the rising edge
- FS(R/X)P = 0, specifying that active (high) frame sync signals are used
- (R/X)DATDLY = 01b, specifying a 1-bit data delay

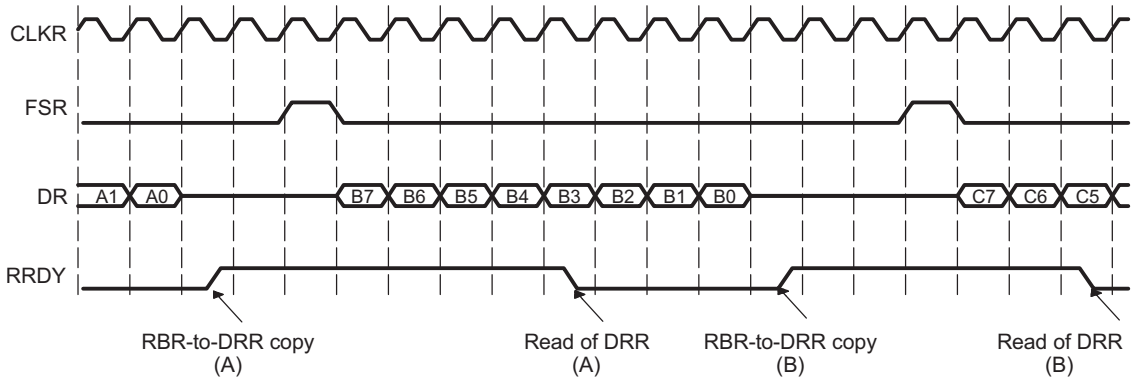
Figure 11-730. McBSP Standard Operation



11.10.4.6.1 Receive Operation

Figure 11-731 shows serial reception. Once the receive frame synchronization signal (FSR) transitions to its active state, it is detected on the first falling edge of the receivers CLKR. The data on the DR pin is then shifted into the receive shift register (RSR) after the appropriate data delay as set by the RDATDLY bit in the receive control register (MCBSP_RCR). The contents of RSR is copied to RBR at the end of every element on the rising edge of the clock, provided RBR is not full with the previous data. Then, an RBR-to-DRR copy activates the RRDY status bit in the serial port control register (MCBSP_SPCR) to 1 on the following falling edge of CLKR. This indicates that the receive data register (MCBSP_DRR) is ready with the data to be read by the CPU or the EDMA controller. RRDY is deactivated when the MCBSP_DRR is read by the CPU or the EDMA controller. See also Section 11.10.4.4.1.2 and Section 11.10.4.5.1.

Figure 11-731. Receive Operation

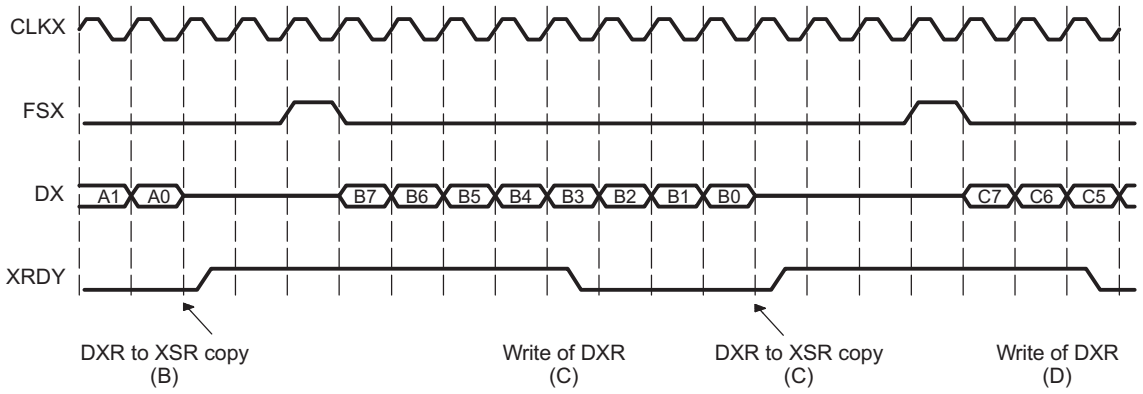


11.10.4.6.2 Transmit Operation

Once transmit frame synchronization occurs, the value in the transmit shift register (XSR) is shifted out and driven on the DX pin after the appropriate data delay as set by the XDATDLY bit in the transmit control register (MCBSP_XCR). The XRDY bit in the serial port control register (MCBSP_SPCR) is activated after every DXR-to-XSR copy on the following falling edge of CLKX, indicating that the data transmit register (MCBSP_DXR) can be written with the next data to be transmitted. XRDY is deactivated when the MCBSP_DXR is written by the CPU or the EDMA controller.

Figure 11-732 illustrates serial transmission. See Section 11.10.4.6.5.4 for information on transmit operation when the transmitter is pulled out of reset (XRST = 1). See also Section 11.10.4.4.1.3 and Section 11.10.4.5.2.

Figure 11-732. Transmit Operation



11.10.4.6.3 Maximum Frame Frequency

The frame frequency is determined by the following equation, which calculates the period between frame synchronization signals:

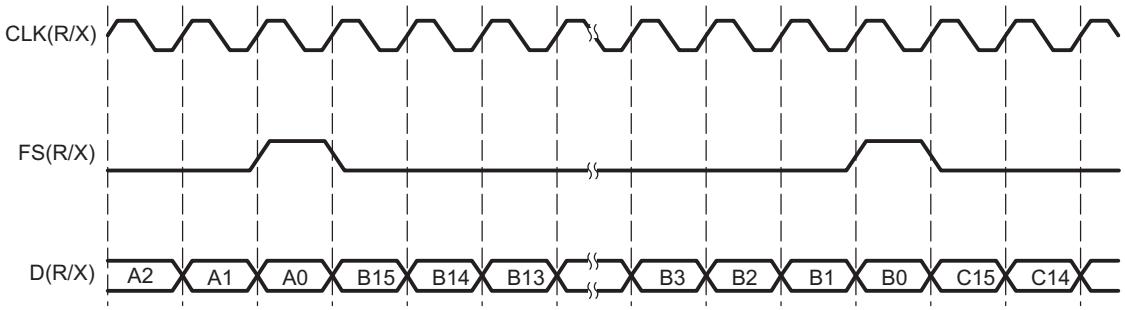
$$\text{Frame frequency} = \frac{\text{Bit-clock frequency}}{\text{Number of bit clocks between frame sync signals}}$$

The frame frequency may be increased by decreasing the time between frame synchronization signals in bit clocks (which is limited only by the number of bits per frame). As the frame transmit frequency is increased, the inactivity period between the data frames for adjacent transfers decreases to 0. The minimum time between frame synchronization pulses is the number of bits transferred per frame. This time also defines the maximum frame frequency, which is calculated by the following equation:

$$\text{Maximum frame frequency} = \frac{\text{Bit-clock frequency}}{\text{Number of bits per frame}}$$

Figure 11-733 shows the McBSP operating at maximum frame frequency. The data bits in consecutive frames are transmitted continuously with no inactivity between bits. If there is a 1-bit data delay, as shown, the frame synchronization pulse overlaps the last bit transmitted in the previous frame.

Figure 11-733. Maximum Frame Frequency for Transmit and Receive



NOTE: For (R/X)DATDLY = 0, the first bit of data transmitted is asynchronous to CLKX, as shown in Figure 11-728.

Maximum frame frequency may not be possible when the word length is 8-bits, depending on the clock divide value CLKGDV. The CPU or EDMA may not be able to service serial port requests that occur as frequently as every 8-bit clocks. This situation can be resolved by allowing additional space between words or choosing a slower bit clock (larger CLKGDV value).

11.10.4.6.4 Frame Synchronization Ignore

The McBSP can be configured to ignore transmit and receive frame synchronization pulses. The (R/X)FIG bit in (R/X)CR can be cleared to 0 to recognize frame sync pulses, or it can be set to 1 to ignore frame sync pulses. In this way,(R/X)FIG can be used either to pack data, if operating at maximum frame frequency, or to ignore unexpected frame sync pulses.

11.10.4.6.4.1 Frame Sync Ignore and Unexpected Frame Sync Pulses

RFIG and XFIG are used to ignore unexpected internal or external frame sync pulses. Any frame sync pulse is considered unexpected if it occurs one or more bit clocks earlier than the programmed data delay from the end of the previous frame specified by ((R/X)DATDLY). Setting the frame ignore bits to 1 causes the serial port to ignore these unexpected frame sync signals.

In reception, if not ignored (RFIG = 0), an unexpected FSR pulse discards the contents of RSR in favor of the incoming data. Therefore, if RFIG = 0, an unexpected frame synchronization pulse aborts the current data transfer, sets RSYNCERR in MCBSP_SPCR to 1, and begins the reception of a new data element. When RFIG = 1, the unexpected frame sync pulses are ignored.

In transmission, if not ignored (XFIG = 0), an unexpected FSX pulse aborts the ongoing transmission, sets the XSYNCERR bit in MCBSP_SPCR to 1, and reinitiates transmission of the current element that was aborted. When XFIG = 1, unexpected frame sync signals are ignored.

Figure 11-734 shows that element B is interrupted by an unexpected frame sync pulse when (R/X)FIG = 0. The reception of B is aborted (B is lost), and a new data element (C) is received after the appropriate data delay. This condition causes a receive synchronization error and thus sets the RSYNCERR bit. However, for transmission, the transmission of B is aborted and the same data (B) is retransmitted after the appropriate data delay. This condition is a transmit synchronization error and thus sets the XSYNCERR bit. Synchronization errors are discussed in Section 11.10.4.6.5.2 and Section 11.10.4.6.5.5.

Figure 11-734. Unexpected Frame Synchronization With (R/X)FIG = 0

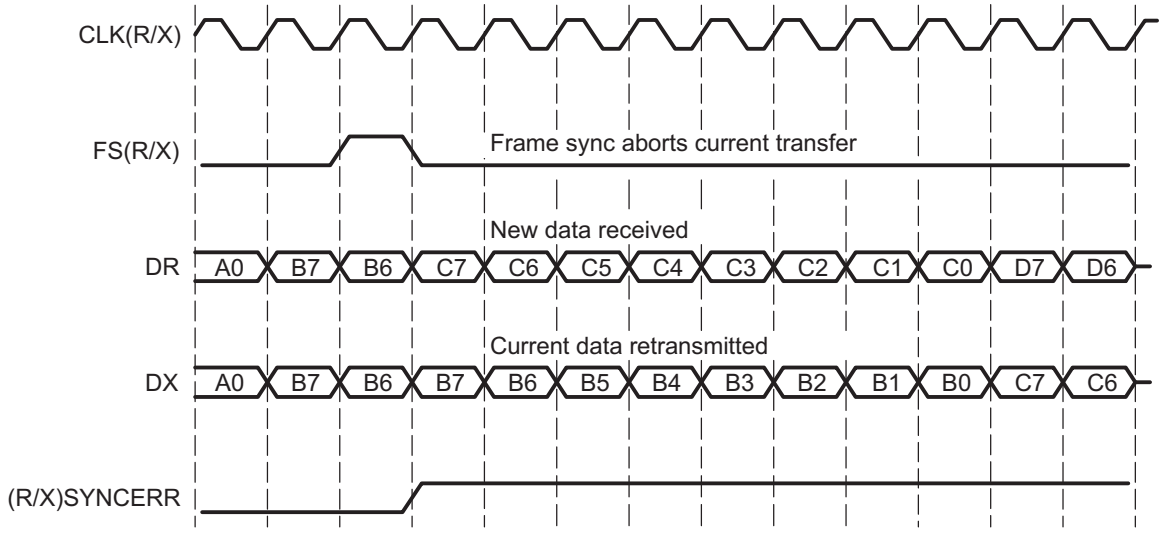
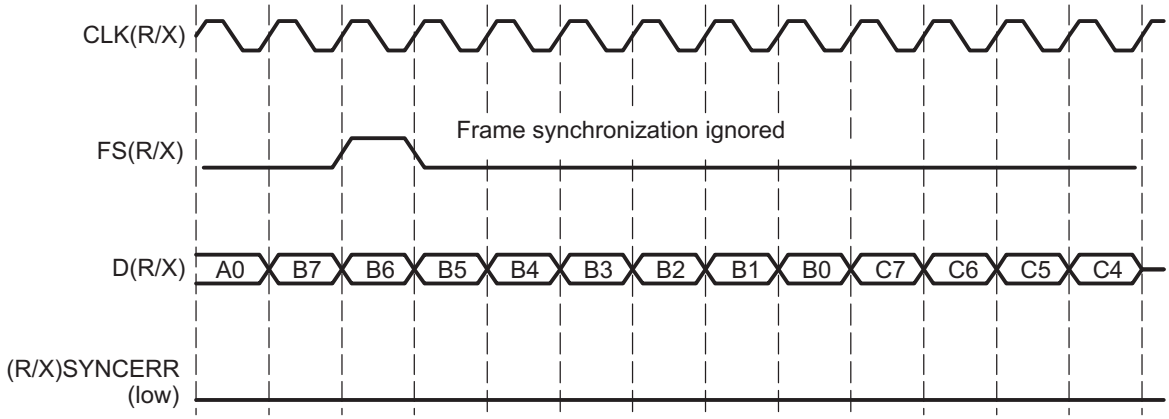


Figure 11-735 shows McBSP operation when unexpected internal or external frame synchronization signals are ignored by setting (R/X)FIG = 1. Here, the transfer of element B is not affected by an unexpected frame synchronization.

Figure 11-735. Unexpected Frame Synchronization With (R/X)FIG = 1



11.10.4.6.4.2 Data Packing using Frame Sync Ignore Bits

Section 11.10.4.2.5.4 describes one method of changing the element length and frame length to simulate 32-bit serial element transfers, thus requiring much less bus bandwidth than four 8-bit transfers require. This example works when there are multiple elements per frame.

Now consider the case of the McBSP operating at maximum packet frequency, as shown in Figure 11-736. Here, each frame has only a single 8-bit element. This stream takes one read transfer and one write transfer for each 8-bit element. Figure 11-737 shows the McBSP configured to treat this stream as a continuous stream of 32-bit elements.

In this example, (R/X)FIG is set to 1 to ignore unexpected subsequent frames. Only one read transfer and one write transfer is needed every 32 bits. This configuration effectively reduces the required bus bandwidth to one-fourth of the bandwidth needed to transfer four 8-bit blocks.

Figure 11-736. Maximum Frame Frequency Operation With 8-Bit Data

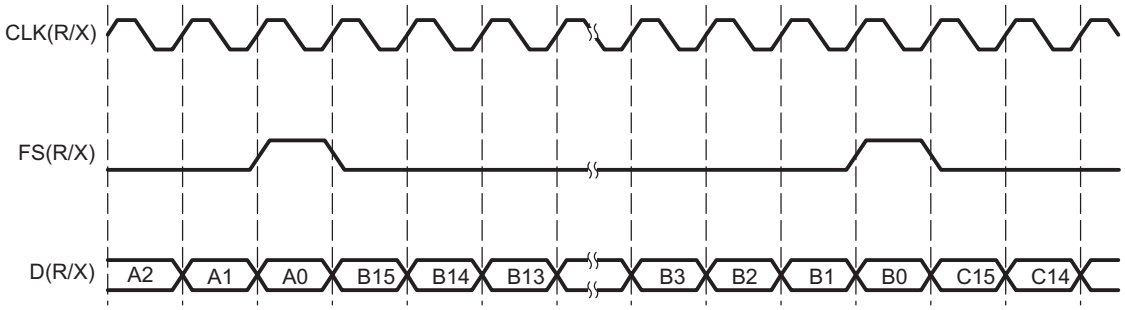
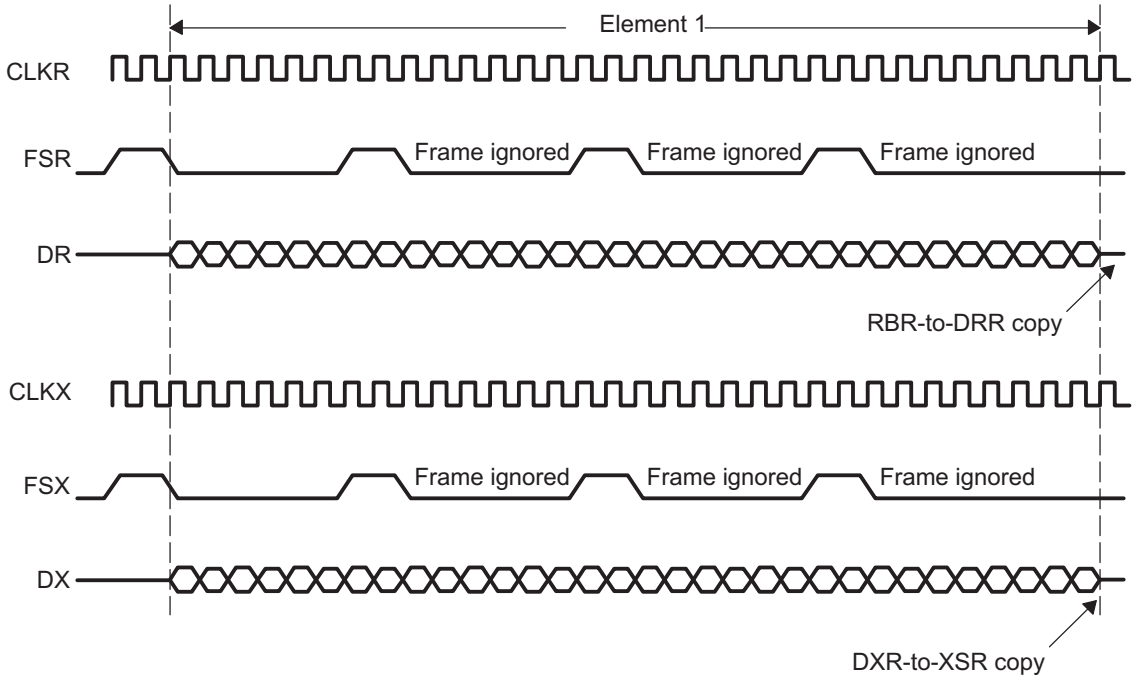


Figure 11-737. Data Packing at Maximum Frame Frequency With (R/X)FIG = 1



11.10.4.6.5 Serial Port Exception Conditions

There are five serial port events that can constitute a system error:

- Receive overrun (RFULL = 1 in MCBSP_SPCR)
- Unexpected receive frame synchronization (RSYNCERR = 1 in MCBSP_SPCR)
- Transmit data overwrite
- Transmit empty (XEMPTY = 0 in MCBSP_SPCR)
- Unexpected transmit frame synchronization (XSYNCERR = 1 in MCBSP_SPCR)

11.10.4.6.5.1 Receive Overrun: RFULL

RFULL = 1 in the serial port control register (MCBSP_SPCR) indicates that the receiver has experienced overrun and is in an error condition. RFULL is set when the following conditions are met:

- MCBSP_DRR has not been read since the last RBR-to-DRR transfer.
- RBR is full and an RBR-to-DRR copy has not occurred.
- RSR is full and an RSR-to-RBR transfer has not occurred.

The data arriving on DR is continuously shifted into RSR (Figure 11-738). Once a complete element is shifted into RSR, an RSR-to-RBR transfer can occur only if an RBR-to-DRR copy is complete. Therefore, if MCBSP_DRR has not been read by the CPU or the EDMA controller since the last RBR-to-DRR transfer (RRDY = 1), an RBR-to-DRR copy does not take place until RRDY = 0. This prevents an RSR-to-RBR copy.

New data arriving on the DR pin is shifted into RSR, and the previous contents of RSR are lost. After the receiver starts running from reset, a minimum of three elements must be received before RFULL can be set, because there was no last RBR-to-DRR transfer before the first element.

This data loss can be avoided if MCBSP_DRR is read no later than two and a half CLKR cycles before the end of the third element (data C) in RSR, as shown in Figure 11-739.

Either of the following events clears the RFULL bit to 0 and allows subsequent transfers to be read properly:

- Reading MCBSP_DRR

- Resetting the receiver (RRST = 0) or the device

Another frame synchronization is required to restart the receiver.

Figure 11-738 shows the receive overrun condition. Because element A is not read before the reception of element B is complete, B is not transferred to MCBSP_DRR yet. Another element, C, arrives and fills RSR. MCBSP_DRR is finally read, but not earlier than two and one half cycles before the end of element C. New data D overwrites the previous element C in RSR. If RFULL is still set after the MCBSP_DRR is read, the next element can overwrite D if MCBSP_DRR is not read in time.

Figure 11-739 shows the case in which RFULL is set but the overrun condition is averted by reading the contents of MCBSP_DRR at least two and a half cycles before the next element, C, is completely shifted into RSR. This ensures that a RBR-to-DRR copy of data B occurs before the next element is transferred from RSR to RBR.

Figure 11-738. Serial Port Receive Overrun

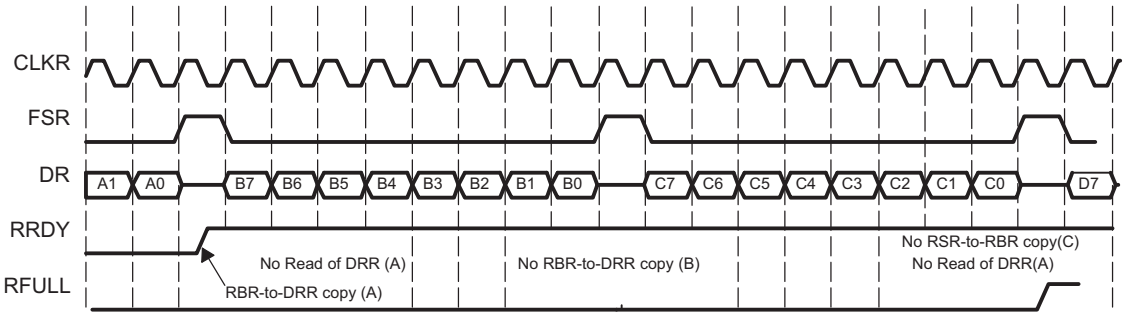
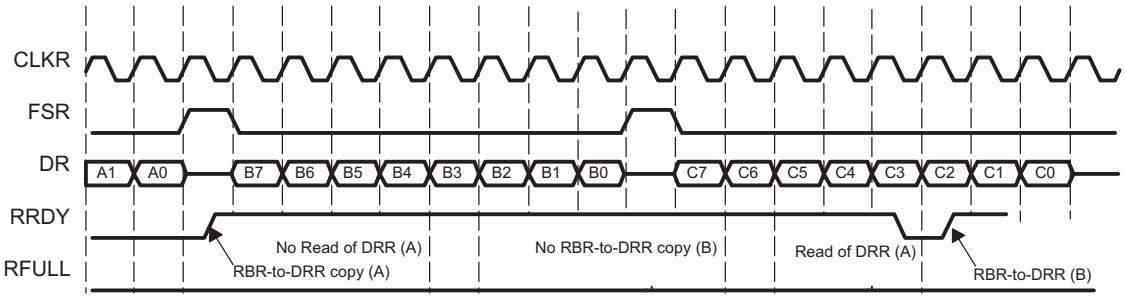


Figure 11-739. Serial Port Receive Overrun Avoided



11.10.4.6.5.2 Unexpected Receive Frame Synchronization: RSYNCERR

Figure 11-740 shows the decision tree that the receiver uses to handle all incoming frame synchronization pulses. The diagram assumes that the receiver has been activated (RRST = 1). Unexpected frame sync pulses can originate from an external source or from the internal sample rate generator. An unexpected frame sync pulse is defined as a sync pulse which occurs RDATDLY bit clocks earlier than the last transmitted bit of the previous frame. Any one of three cases can occur:

- Case 1: Unexpected FSR pulses with RFIG = 1. This case is discussed in Section 11.10.4.6.4.1 and shown in Figure 11-735. Here, receive frame sync pulses are ignored and the reception continues.
- Case 2: Normal serial port reception. There are three reasons for a receive not to be in progress:
  - This FSR is the first after RRST = 1.
  - This FSR is the first after MCBSP_DRR has been read clearing an RFULL condition.
  - The serial port is in the inter-packet intervals. The programmed data delay (RDATDLY) for reception may start during these inter-packet intervals for the first bit of the next element to be received. Thus, at maximum frame frequency, frame synchronization can still be received RDATDLY bit clocks before the first bit of the associated element.
  - For this case, reception continues normally, because these are not unexpected frame sync pulses.
- Case 3: Unexpected receive frame synchronization with RFIG = 0 (unexpected frame not ignored). This case was shown in Figure 11-734 for maximum packet frequency. Figure 11-741 shows this case during normal operation of the serial port with time intervals between packets. Unexpected frame sync pulses are detected when they occur the value in RDATDLY bit clocks before the last bit of the previous element is received on DR. In both cases, RSYNCERR in MCBSP_SPCR is set. RSYNCERR can be cleared only by receiver reset or by writing a 0 to this bit in MCBSP_SPCR. If RINTM = 11b in MCBSP_SPCR, RSYNCERR drives the receive interrupt (RINT) to the CPU.

NOTE: Note that the RSYNCERR bit in MCBSP_SPCR is a read/write bit, so writing a 1 to it sets the error condition. Typically, writing a 0 is expected.

Figure 11-740. Decision Tree Response to Receive Frame Synchronization Pulse

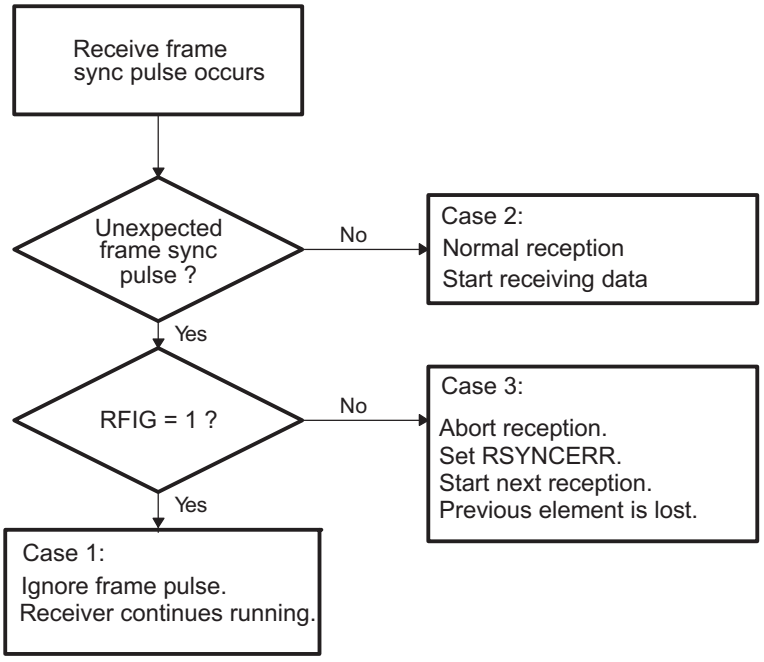
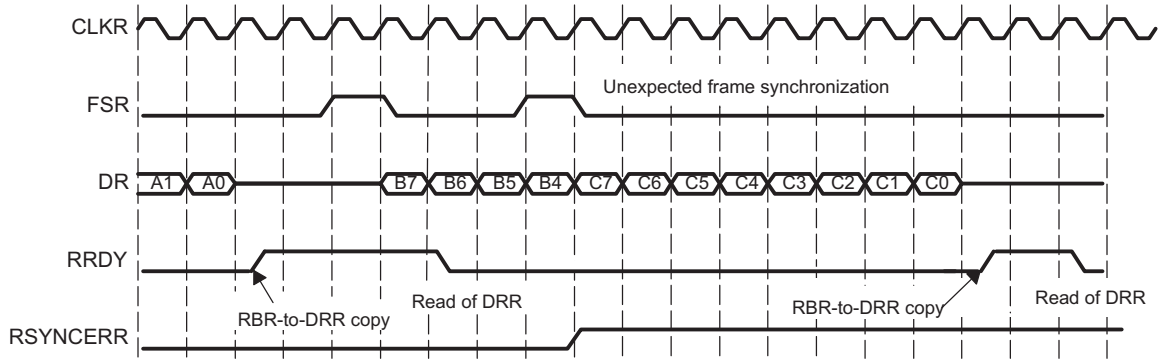


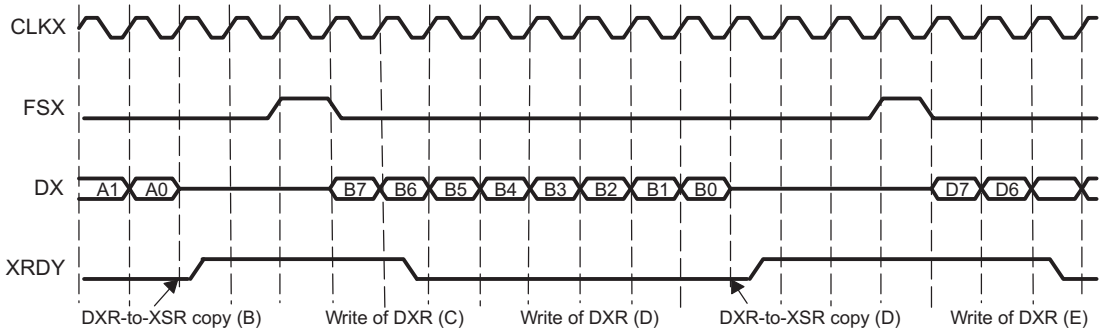
Figure 11-741. Unexpected Receive Frame Synchronization Pulse



11.10.4.6.5.3 Transmit With Data Overwrite

Figure 11-742 shows what happens if the data in MCBSP_DXR is overwritten before it is transmitted. For example, load the MCBSP_DXR with data C. A subsequent write to the MCBSP_DXR overwrites C with D before C is copied to the XSR. Thus, C is never transmitted on DX. The CPU can avoid overwriting data by polling XRDY before writing to MCBSP_DXR or by waiting for a programmed XINT to be triggered by XRDY (XINTM = 00b). The EDMA controller can avoid overwriting by synchronizing data writes with XEVT. See also Section 11.10.4.4.1.3.

Figure 11-742. Transmit with Data Overwrite



11.10.4.6.5.4 Transmit Empty: XEMPTY

XEMPTY indicates whether the transmitter has experienced underflow. Either of the following conditions causes XEMPTY to become active (XEMPTY = 0):

- During transmission, MCBSP_DXR has not been loaded since the last DXR-to-XSR copy, and all bits of the data element in XSR have been shifted out on DX.
- The transmitter is reset (XRST = 0 or the device is reset), and then restarted.

During an underflow condition, the transmitter continues to transmit the old data in MCBSP_DXR for every new frame sync signal FSX (generated by an external device, or by the internal sample rate generator) until a new element is loaded into MCBSP_DXR by the CPU or the EDMA controller. XEMPTY is deactivated
(XEMPTY = 1) when this new element in MCBSP_DXR is transferred to XSR. In the case when the FSX is generated by a DXR-to-XSR copy (FSXM = 1 in MCBSP_PCR and FSGM = 0 in MCBSP_SRGR), the McBSP does not generate any new frame sync until new data is written to the MCBSP_DXR and a DXR-to-XSR copy occurs.

When the transmitter is taken out of reset (XRST = 1), it is in a transmit ready (XRDY = 1) and transmit empty (XEMPTY = 0) condition. If MCBSP_DXR is loaded by the CPU or the EDMA controller before FSX goes active, a valid DXR-to-XSR transfer occurs. This allows for the first element of the first frame to be valid even before the transmit frame sync pulse is generated or detected. Alternatively, if a transmit frame sync is detected before MCBSP_DXR is loaded, 0s are output on DX.

Figure 11-743 shows a transmit underflow condition. After B is transmitted, B is retransmitted on DX if the MCBSP_DXR is failed to reload before the subsequent frame synchronization. Figure 11-744 shows the case of writing to MCBSP_DXR just before a transmit underflow condition that would otherwise occur. After B is transmitted, C is written to MCBSP_DXR before the next transmit frame sync pulse occurs.

Figure 11-743. Transmit Empty

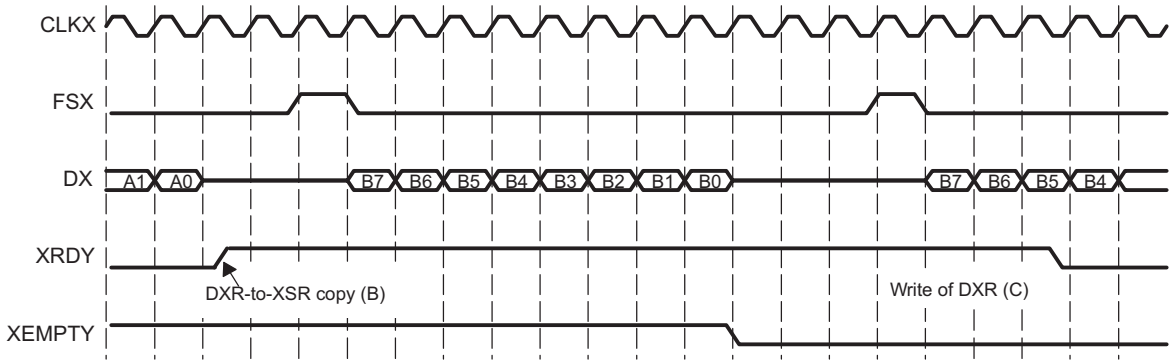
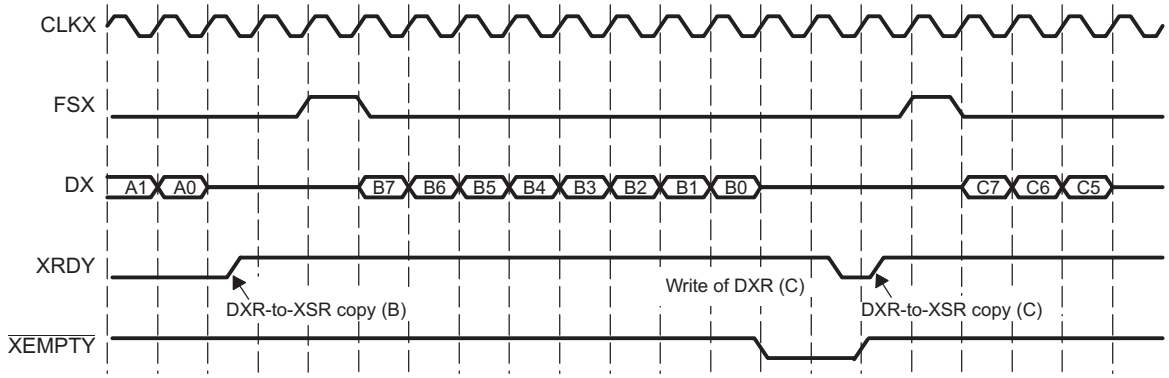


Figure 11-744. Transmit Empty Avoided



11.10.4.6.5.5 Unexpected Transmit Frame Synchronization: XSYNCERR

A transmit frame sync error (XSYNCERR) may occur the first time the transmitter is enabled (XRST = 1) after a device reset. To avoid this, after enabling the transmitter for the first time, the following procedure must be followed:

1. Wait for two CLKG cycles. The unexpected frame sync error (XSYNCERR), if any, occurs within this time period.
2. Disable the transmitter (XRST = 0). This clears any XSYNCERR.
3. Re-enable the transmitter (XRST = 1).

See also Section 11.10.5.1 for details on initialization procedure.

Figure 11-745 shows the decision tree that the transmitter uses to handle all incoming frame synchronization signals. The diagram assumes that the transmitter has been started (XRST = 1). An unexpected transmit frame sync pulse is defined as a sync pulse that occurs XDATDLY bit clocks earlier than the last transmitted bit of the previous frame. Any one of three cases can occur:

- Case 1: Unexpected FSX pulses with XFIG = 1. This case is discussed in Section 11.10.4.6.4.1 and shown in Figure 11-735. In this case, unexpected FSX pulses are ignored and the transmission continues.
- Case 2: FSX pulses with normal serial port transmission. This case is discussed in Section 11.10.4.6.2. There are two possible reasons for a transmit not to be in progress:
  - This FSX pulse is the first one to occur after XRST = 1.
  - The serial port is in the interpacket intervals. The programmed data delay (XDATDLY) may start during these interpacket intervals before the first bit of the next element is transmitted. Therefore, if operating at maximum packet frequency, frame synchronization can still be received XDATDLY bit clocks before the first bit of the associated element.
- Case 3: Unexpected transmit frame synchronization with XFIG = 0. The case was shown in Figure 11-734 for frame synchronization with XFIG = 0 at maximum packet frequency. Figure 11-746 shows the case for normal operation of the serial port with interpacket intervals. In both cases, XSYNCERR in MCBSP_SPCR is set. XSYNCERR can be cleared only by transmitter reset or by writing a 0 to this bit in MCBSP_SPCR.
  If XINTM = 11b in MCBSP_SPCR, XSYNCERR drives the receive interrupt (XINT) to the CPU.

NOTE: The XSYNCERR bit in MCBSP_SPCR is a read/write bit, so writing a 1 to it sets the error condition. Typically, writing a 0 is expected.

Figure 11-745. Decision Tree Response to Transmit Frame Synchronization Pulse

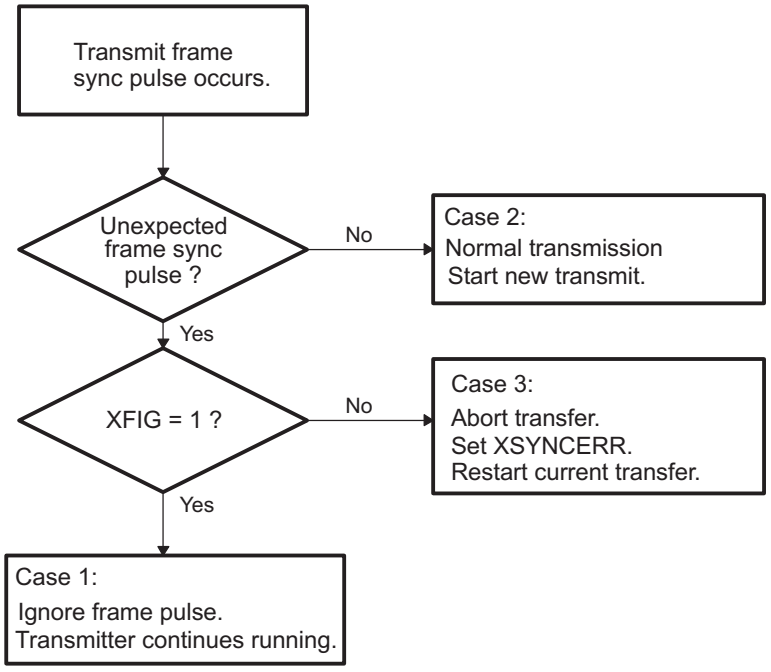
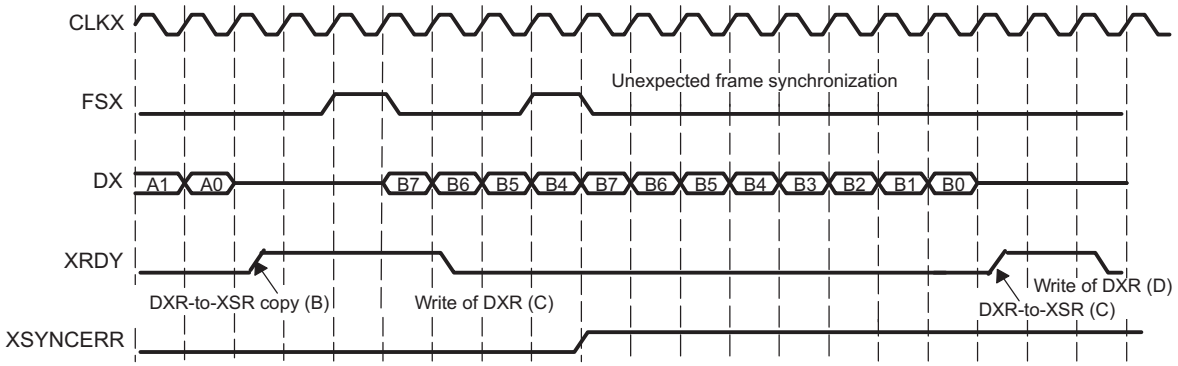


Figure 11-746. Unexpected Transmit Frame Synchronization Pulse



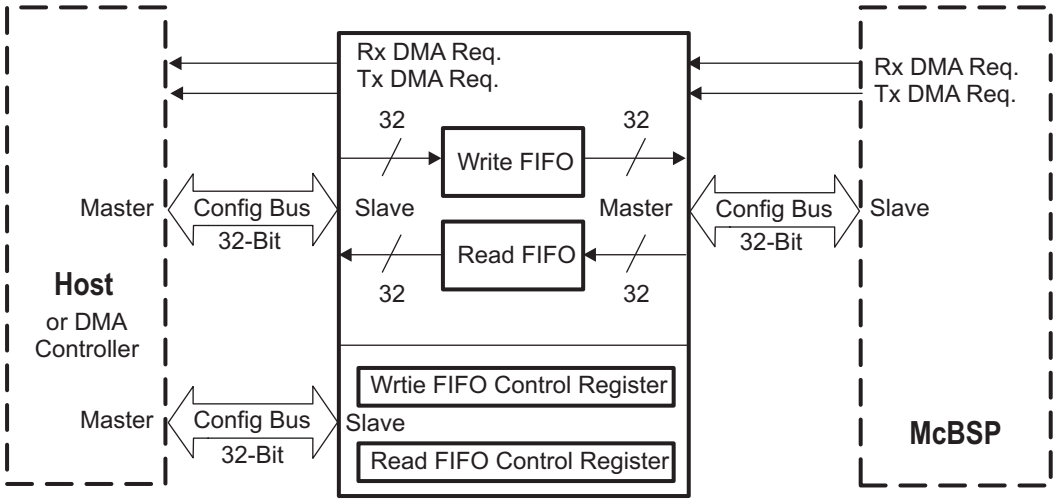
11.10.4.6.6 McBSP Buffer FIFO (BFIFO)

The McBSP Buffer FIFO (BFIFO) provides additional data buffering for the McBSP. The time it takes the host CPU or EDMA controller to respond to DMA requests from the McBSP may vary; the additional buffering provided by the BFIFO allows greater tolerance to such variations.

The BFIFO contains two FIFOs: one Read FIFO (RFIFO) and one Write FIFO (WFIFO). To ensure backward compatibility with existing software, both the Read and Write FIFOs are disabled by default. See Figure 11-747 for a high-level block diagram of the BFIFO.

The BFIFO can be enabled/disabled and configured using the Write FIFO control register (MCBSP_WFIFOCTL) and the Read FIFO control register (MCBSP_RFIFOCTL). Note that if the Read or Write FIFO is to be enabled, it must be enabled before initializing the receive/transmit section of the McBSP (see Section 11.10.5.1 for details).

Figure 11-747. McBSP Buffer FIFO (BFIFO) Block Diagram



NOTE: The McBSP Buffer FIFO (BFIFO) has a different memory-map than the McBSP memory-mapped registers (MMRs); therefore, the BFIFO is accessible by way of a different Configuration Bus.

11.10.4.6.6.1 BFIFO Data Transmission

When the Write FIFO is disabled, transmit DMA requests pass directly from the McBSP to the host/DMA controller. Whether the WFIFO is enabled or disabled, the McBSP generates transmit DMA requests as needed; the BFIFO is invisible to the McBSP.

When the Write FIFO is enabled, transmit DMA requests from the McBSP are sent to the BFIFO, which then generates transmit DMA requests to the host/DMA controller.

If the Write FIFO is enabled during a transmit DMA request from the McBSP, the WFIFO writes *WNUMDMA* 32-bit words to the McBSP, if and when there are at least two *WNUMDMA* words in the Write FIFO. If there are not, the WFIFO waits until this condition has been satisfied; at this point, it writes *WNUMDMA* words to the McBSP.

If the host CPU writes to the Write FIFO, independent of a transmit DMA request, the WFIFO accepts host writes until full. Beyond this point, excess data is discarded.

Note that when the WFIFO is first enabled, it immediately issues a transmit DMA request to the host. This is because it begins in an empty state and is therefore ready to accept data.

11.10.4.6.6.1.1 Transmit DMA Event Pacer

The BFIFO may be configured to delay making a transmit DMA request to the host until the Write FIFO has enough space for a specified number of words. In this situation, the number of transmit DMA requests to the host or DMA controller is reduced.

If the Write FIFO has space to accept *WNUMEVT* 32-bit words, it generates a transmit DMA request to the host and waits for a response. As soon as *WNUMEVT* words have been written to the WFIFO, the WFIFO checks again to see if there is space for *WNUMEVT* 32-bit words. If there is space, it generates another transmit DMA request to the host, and so on. In this fashion, the Write FIFO attempts to stay filled.

If transmit DMA event pacing is desired, the WNUMEVT bits in MCBSP_WFIFOCTL should be set to a non-zero integer multiple of the value in the WNUMDMA bits. If transmit DMA event pacing is not desired, the value in the WNUMEVT bits should be set equal to the value in the WNUMDMA bits.

11.10.4.6.6.2 BFIFO Data Reception

When the Read FIFO is disabled, receive DMA requests pass directly from McBSP to the host/DMA controller. Whether the RFIFO is enabled or disabled, the McBSP generates receive DMA requests as needed; the BFIFO is invisible to the McBSP.

When the Read FIFO is enabled, receive DMA requests from the McBSP are sent to the BFIFO, which then generates receive DMA requests to the host/DMA controller.

If the Read FIFO is enabled and the McBSP makes a receive DMA request, the RFIFO reads *RNUMDMA* 32-bit words from the McBSP, if and when the RFIFO has space for *RNUMDMA* words. If the RFIFO does not have space, the RFIFO waits until this condition has been satisfied; at this point, it reads *RNUMDMA* words from the McBSP.

If the host CPU reads the Read FIFO independent of a receive DMA request and the RFIFO currently contains no *RNUMEVT* words, those words will be read correctly, emptying the RFIFO.

11.10.4.6.6.2.1 Receive DMA Event Pacer

The BFIFO may be configured to delay making a receive DMA request to the host until the Read FIFO contains a specified number of words. In this situation, the number of receive DMA requests to the host or DMA controller is reduced.

If the Read FIFO contains at least *RNUMEVT* 32-bit words, it generates a receive DMA request to the host and then waits for a response. As soon as *RNUMEVT* 32-bit words have been read from the RFIFO, the RFIFO checks again to see if it contains at least another *RNUMEVT* words. If it does, it generates another receive DMA request to the host, and so on. In this fashion, the Read FIFO attempts to stay empty.

If receive DMA event pacing is desired, the RNUMEVT bits in MCBSP_RFIFOCTL should be set to a non-zero integer multiple of the value in RNUMDMA bits. If receive DMA event pacing is not desired, then the value in the RNUMEVT bits should be set equal to the value in the RNUMDMA bits.

11.10.4.6.6.3 Arbitration Between Transmit and Receive DMA Requests

If both the WFIFO and the RFIFO are enabled and a transmit DMA request and receive DMA request occur simultaneously, priority is given to the transmit DMA request. Once a transfer is in progress, it is allowed to complete.

If only the WFIFO is enabled and a transmit DMA request and receive DMA request occur simultaneously, priority is given to the transmit DMA request. Once a transfer is in progress, it is allowed to complete.

If only the RFIFO is enabled and a transmit DMA request and receive DMA request occur simultaneously, priority is given to the receive DMA request. Once a transfer is in progress, it is allowed to complete.

11.10.4.7 Multi-channel Selection Modes

This section defines the functions and related information concerning the multi-channel selection modes.

11.10.4.7.1 Channels, Blocks, and Partitions

A McBSP channel is a time slot for shifting in/out the bits of one serial word. Each McBSP supports up to 128 channels for reception and 128 channels for transmission. In the receiver and in the transmitter, the 128 available channels are divided into eight blocks that each contain 16 contiguous channels:

- Block 0: Channels 0 through 15
- Block 1: Channels 16 through 31
- Block 2: Channels 32 through 47
- Block 3: Channels 48 through 63
- Block 4: Channels 64 through 79
- Block 5: Channels 80 through 95
- Block 6: Channels 96 through 111
- Block 7: Channels 112 through 127

The blocks are assigned to partitions according to the selected partition mode. In the 2-partition mode, one even-numbered block (0, 2, 4, or 6) is assigned to partition A and one odd-numbered block (1, 3, 5, or 7) to partition B. In the 8-partition mode, blocks 0 through 7 are automatically assigned to partitions, A through H, respectively.

The number of partitions for reception and the number of partitions for transmission are independent. For example, it is possible to use 2 receive partitions (A and B) and 8 transmit partitions (A through H).

11.10.4.7.2 Multi-channel Selection

When McBSP uses a time-division multiplexed (TDM) data stream while communicating with other McBSPs or serial devices, the McBSP may need to receive and/or transmit on only a few channels. To save memory and bus bandwidth, use a multi-channel selection mode to prevent data flow in some of the channels. The McBSP has one receive multi-channel selection mode and three transmit multi-channel selection modes.

Each channel partition has a dedicated channel enable register. If the appropriate multi-channel selection mode is on, each bit in the register controls whether data flow is allowed or prevented in one of the channels that is assigned to that partition.

11.10.4.7.3 Configuring a Frame for Multi-channel Selection

Before enabling a multi-channel selection mode, make sure that the data frame is properly configured:

- Select a single-phase frame (RPHASE/XPHASE = 0). Each frame represents a TDM data stream.
- Set a frame length (RFRLEN1/XFRLEN1) that includes the highest-numbered channel that is to be used. For example, to use channels 0, 15, and 39 for reception, the receive frame length must be at least 40 (RFRLEN1 = 39). If XFRLEN1 = 39 in this case, the receiver creates 40 time slots per frame but only receives data during time slots 0, 15, and 39 of each frame.

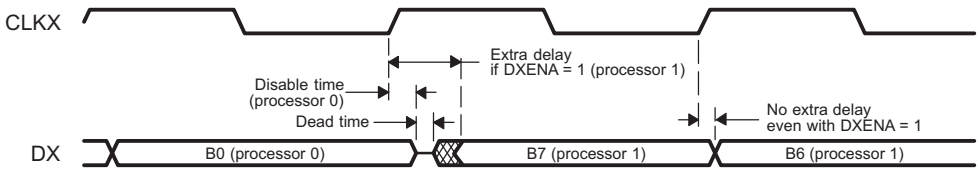
NOTE: The frame-sync pulse can be generated internally by the sample rate generator or it can be supplied externally by another source. In a multi-channel mode configuration with external frame-sync generation, the McBSP transmitter will ignore the first frame-sync pulse after it is taken out of reset. The transmitter will transmit only on the second frame-sync pulse. The receiver will shift in data on the first frame-sync pulse, regardless of whether it is generated internally or externally.

11.10.4.7.4 DX Enabler: DXENA

The DXENA bit in the serial port control register (MCBSP_SPCR) controls the high-impedance enable on the DX pin. When DXENA = 1, the McBSP enables extra delay for the DX pin turn-on time. This feature is useful for McBSP multi-channel operations, such as in a time-division multiplexed (TDM) system. The McBSP supports up to 128 channels in a multi-channel operation. These channels can be driven by different devices in a TDM data communication line, such as the T1/E1 line.

In any multi-channel operation where multiple devices transmit over the same DX line, to avoid bus contention two devices should not transmit data simultaneously. Enough dead time should exist between the transmission of the first data bit of the current device and the transmission of the last data bit of the previous device. In other words, the last data bit of the previous device needs to be disabled to a high-impedance state before the next device begins transmitting data to the same data line, as shown in Section 11.10.4.7.5.

Figure 11-748. DX Timing for Multi-channel Operation



When two McBSPs are used to transmit data over the same TDM line, bus contention occurs if DXENA = 0. The first McBSP turns off the transmission of the last data bit (changes DX from valid to a high-impedance state) after a disable time specified in the device Data Manual. As shown in Section 11.10.4.7.5, this disable time is measured from the CLKX active clock edge.

The next McBSP turns on its DX pin (changes from a high-impedance state to valid) after a delay time. Again, this delay time is measured from the CLKX active clock edge. Bus contention occurs because the dead time between the two devices is not enough. In this case an alternative software or hardware methods need to be applied to ensure proper multi-channel operation.

If DXENA = 1 is set in the second McBSP, the second McBSP turns on its DX pin after some extra delay time. This ensures that the previous McBSP on the same DX line is disabled before the second McBSP starts driving out data. The DX enabler controls only the high-impedance enable on the DX pin, not the data itself. Data is shifted out to the DX pin at the same time as in the case when DXENA = 0. The only difference is that with DXENA = 1, the DX pin is masked to a high-impedance state for some extra CPU cycles before the data is seen on the TDM data line. Therefore, only the first bit of data is delayed.

11.10.4.7.5 Using Two Partitions

For multi-channel selection operation in the receiver and/or the transmitter, two partitions or eight partitions can be used. In the 2-partition mode (RMCME = 0 for reception, XMCME = 0 for transmission), McBSP channels are activated using an alternating scheme. In response to a frame-sync pulse, the receiver or transmitter begins with the channels in partition A and then alternates between partitions B and A until the complete frame has been transferred. When the next frame-sync pulse occurs, the next frame is transferred, beginning with the channels in partition A.

11.10.4.7.5.1 Assigning Blocks to Partitions A and B

For reception, any two of the eight receive-channel blocks can be assigned to receive partitions A and B (see Table 11-1606), which means up to 32 receive channels can be enabled at any given point in time. Similarly, any two of the eight transmit-channel blocks (up to 32 enabled transmit channels) can be assigned to transmit partitions A and B (see Table 11-1607). To change which blocks of channels are assigned to the partitions, see Section 11.10.4.7.5.2.

For reception:

- Assign an even-numbered channel block (0, 2, 4, or 6) to receive partition A by writing to the RPABLK bit in the multi-channel control register (MCBSP_MCR). In the receive multi-channel selection mode, the channels in this partition are controlled by the enhanced receive channel enable register partition A/B (MCBSP_RCERE0).
- Assign an odd-numbered block (1, 3, 5, or 7) to receive partition B with the RPBBLK bit in MCBSP_MCR. In the receive multi-channel selection mode, the channels in this partition are controlled by the enhanced receive channel enable register partition A/B (MCBSP_RCERE0).

For transmission:

- Assign an even-numbered channel block (0, 2, 4, or 6) to transmit partition A by writing to the XPABLK bit in the multi-channel control register (MCBSP_MCR). In one of the transmit multi-channel selection modes, the channels in this partition are controlled by the enhanced transmit channel enable register partition A/B (MCBSP_XCERE0).
- Assign an odd-numbered block (1, 3, 5, or 7) to transmit partition B with the XPBBLK bit in MCBSP_MCR. In one of the transmit multi-channel selection modes, the channels in this partition are controlled by the enhanced transmit channel enable register partition A/B (MCBSP_XCERE0).

Table 11-1606. Receive Channel Assignment and Control When Two Receive Partitions are Used

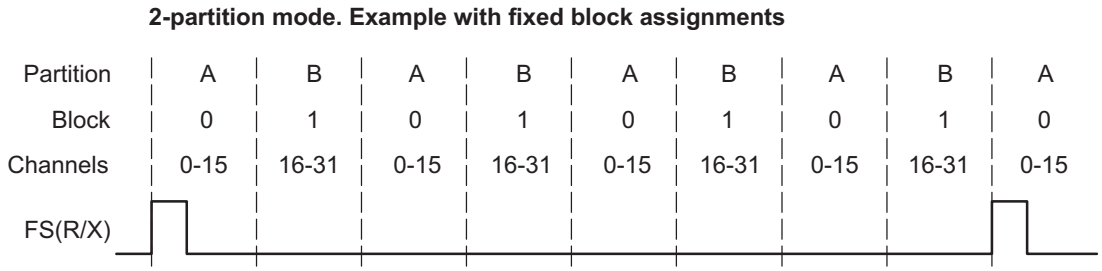
Receive Partition	Assigned Block of Receive Channels	RPABLK Bit in MCBSP_MCR	RPBBLK Bit in MCBSP_MCR	MCBSP_RCERE0 Bits
A	Block 0: channels 0 through 15	0	–	RCE0-RCE15
A	Block 2: channels 32 through 47	1h	–	RCE0-RCE15
A	Block 4: channels 64 through 79	2h	–	RCE0-RCE15
A	Block 6: channels 96 through 111	3h	–	RCE0-RCE15
B	Block 1: channels 16 through 31	–	0	RCE16-RCE31
B	Block 3: channels 48 through 63	–	1h	RCE16-RCE31
B	Block 5: channels 80 through 95	–	2h	RCE16-RCE31
B	Block 7: channels 112 through 127	–	3h	RCE16-RCE31

Table 11-1607. Transmit Channel Assignment and Control When Two Transmit Partitions are Used

Transmit Partition	Assigned Block of Transmit Channels	XPABLK Bit in MCBSP_MCR	XPBBLK Bit in MCBSP_MCR	MCBSP_XCERE0 Bits
A	Block 0: channels 0 through 15	0	–	XCE0-XCE15
A	Block 2: channels 32 through 47	1h	–	XCE0-XCE15
A	Block 4: channels 64 through 79	2h	–	XCE0-XCE15
A	Block 6: channels 96 through 111	3h	–	XCE0-XCE15
B	Block 1: channels 16 through 31	–	0	XCE16-XCE31
B	Block 3: channels 48 through 63	–	1h	XCE16-XCE31
B	Block 5: channels 80 through 95	–	2h	XCE16-XCE31
B	Block 7: channels 112 through 127	–	3h	XCE16-XCE31

Figure 11-749 shows an example of alternating between the channels of partition A and the channels of partition B. Channels 0-15 have been assigned to partition A, and channels 16-31 have been assigned to partition B. In response to a frame-sync pulse, the McBSP begins a frame transfer with partition A and then alternates between partitions B and A until the complete frame is transferred.

Figure 11-749. Alternating Between the Channels of Partition A and the Channels of Partition B



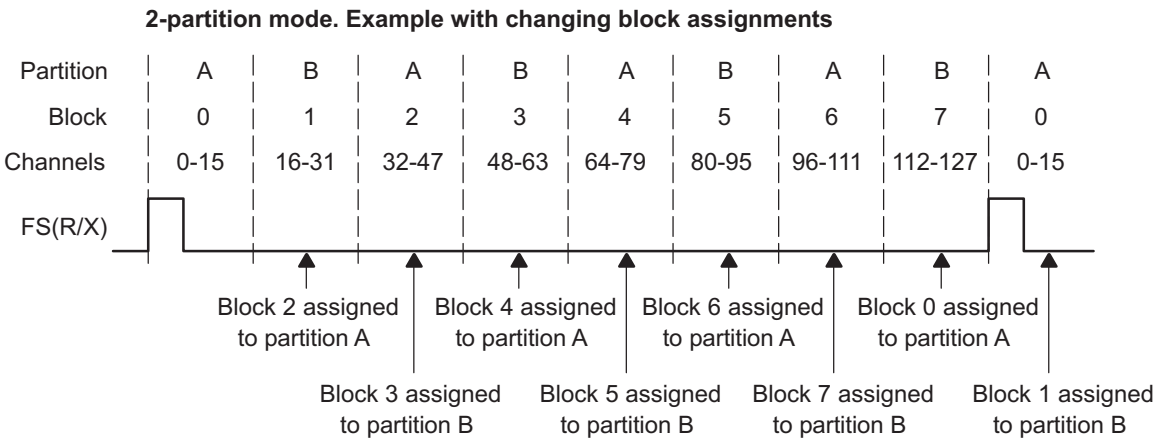
11.10.4.7.5.2 Reassigning Blocks During Reception/Transmission

To use more than 32 channels, change which channel blocks are assigned to partitions A and B during the course of a data transfer. However, these changes must be carefully timed. While a partition is being transferred, the associated block assignment bits cannot be modified, and its associated channel enable register cannot be modified. For example, if block 3 is being transferred and block 3 is assigned to partition A, (R/X)PABLK cannot be modified to assign different channels to partition A, and (R/X)CERE0 cannot be modified to change the channel configuration for partition A. Several features of the McBSP help to time the reassignment:

- The block of channels currently involved in reception/transmission (the current block) is reflected in the RCBLK/XCBLK bits. The program can poll these bits to determine which partition is active. When a partition is not active, it is safe to change its block assignment and channel configuration.
- At the end of every block (at the boundary of two partitions), an interrupt can be sent to the CPU. In response to the interrupt, the CPU can then check the RCBLK/XCBLK bits and update the inactive partition.

Figure 11-750 shows an example of reassigning channels throughout a data transfer. In response to a frame-sync pulse, the McBSP alternates between partitions A and B. Whenever partition B is active, the CPU changes the block assignment for partition A. Whenever partition A is active, the CPU changes the block assignment for partition B.

Figure 11-750. Reassigning Channel Blocks Throughout a McBSP Data Transfer



11.10.4.7.6 Using Eight Partitions

For multi-channel selection operation in the receiver and/or the transmitter, eight partitions or two partitions can be used. To choose the 8-partition mode (RMCME = 1 for reception, XMCME = 1 for transmission), McBSP partitions are activated in the following order: A, B, C, D, E, F, G, H. In response to a frame-sync pulse, the receiver or transmitter begins with the channels in partition A and then continues with the other partitions in order until the complete frame has been transferred. When the next frame-sync pulse occurs, the next frame is transferred, beginning with the channels in partition A. In the 8-partition mode, the (R/X)PABLK and (R/X)PBBLK bits are ignored and the 16-channel blocks are assigned to the partitions as shown in Table 11-1608 and Table 11-1609. These assignments cannot be changed. Table 11-1608 and Table 11-1609 also show the registers used to control the channels in the partitions.

Table 11-1608. Receive Channel Assignment and Control When Eight Receive Partitions are Used

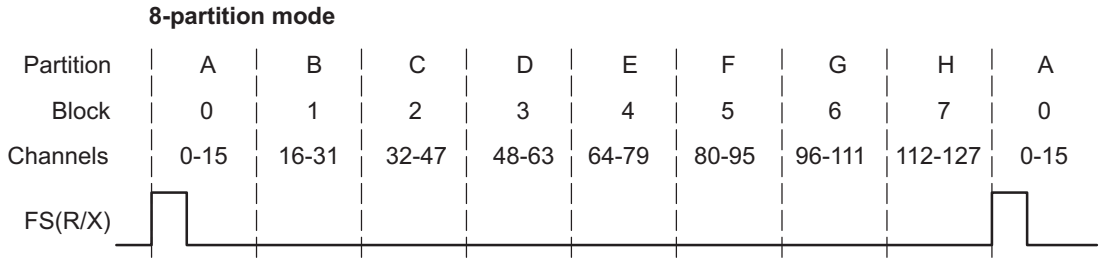
Receive Partition	Assigned Block of Receive Channels	Register Used For Channel Control
A	Block 0: channels 0 through 15	MCBSP_RCERE0
B	Block 1: channels 16 through 31	MCBSP_RCERE0
C	Block 2: channels 32 through 47	MCBSP_RCERE1
D	Block 3: channels 48 through 63	MCBSP_RCERE1
E	Block 4: channels 64 through 79	MCBSP_RCERE2
F	Block 5: channels 80 through 95	MCBSP_RCERE2
G	Block 6: channels 96 through 111	MCBSP_RCERE3
H	Block 7: channels 112 through 127	MCBSP_RCERE3

Table 11-1609. Transmit Channel Assignment and Control When Eight Transmit Partitions are Used

Transmit Partition	Assigned Block of Transmit Channels	Register Used For Channel Control
A	Block 0: channels 0 through 15	MCBSP_XCERE0
B	Block 1: channels 16 through 31	MCBSP_XCERE0
C	Block 2: channels 32 through 47	MCBSP_XCERE1
D	Block 3: channels 48 through 63	MCBSP_XCERE1
E	Block 4: channels 64 through 79	MCBSP_XCERE2
F	Block 5: channels 80 through 95	MCBSP_XCERE2
G	Block 6: channels 96 through 111	MCBSP_XCERE3
H	Block 7: channels 112 through 127	MCBSP_XCERE3

Figure 11-751 shows an example of the McBSP using the 8-partition mode. In response to a frame-sync pulse, the McBSP begins a frame transfer with partition A and then activates B, C, D, E, F, G, and H to complete a 128-word frame.

Figure 11-751. McBSP Data Transfer in the 8-Partition Mode



11.10.4.7.7 Receive Multi-channel Selection Mode

The RMCM bit in the multi-channel control register (MCBSP_MCR) determines whether all channels or only selected channels are enabled for reception. When RMCM = 0, all 128 receive channels are enabled and cannot be disabled. When RMCM = 1, the receive multi-channel selection mode is enabled. In this mode:

- Channels can be individually enabled or disabled. The only channels enabled are those selected in the appropriate enhanced receive channel enable register (RCEREn). The way channels are assigned to the RCEREn depends on the number of receive channel partitions (2 or 8), as defined by the RMCME bit in MCBSP_MCR.
- If a receive channel is disabled, any bits received in that channel are passed only as far as the receive buffer register (RBR). The receiver does not copy the content of the RBR to the MCBSP_DRR, and as a result, does not set the receiver ready bit (RRDY). Therefore, no DMA synchronization event (REVT) is generated, and if the receiver interrupt mode depends on RRDY (RINTM = 0), no interrupt is generated.

As an example of how the McBSP behaves in the receive multi-channel selection mode, enable only channels 0, 15, and 39 and that the frame length is 40. The McBSP:

1. Accepts bits shifted in from the DR pin in channel 0.
2. Ignores bits received in channels 1-14.
3. Accepts bits shifted in from the DR pin in channel 15.
4. Ignores bits received in channels 16-38.
5. Accepts bits shifted in from the DR pin in channel 39.

11.10.4.7.7.1 RCEREn Registers Used in Receive Multi-channel Selection Mode

For multi-channel selection operation, the assignment of channels to the enhanced receive channel enable register (RCERE *n*) depends on whether 32 or 128 channels are individually selectable, as defined by the RMCME bit in the multi-channel control register (MCR). For each of these two cases, shows which block of channels is assigned to each RCERE *n* used. For each RCERE *n*, shows which channel is assigned to each of the bits.

Table 11-1610. Use of the Receive Channel Enable Registers

	Block Assignments		Channel Assignments	
Number of selectable channels	**RCERE n**	**Block assigned[(1)]**	**Bit in RCERE n**	**Channel assigned[(1)]**
32 (RMCME = 0)	RCERE0	Channels *n* to (*n* + 15) The block of channels (0, 2, 4, or 6) is selected with the RPABLK bit in MCR	RCE0 RCE1 ... RCE15	Channel *n* Channel (*n* + 1) ... Channel (*n* + 15)
		Channels m to (*m* + 15) The block of channels (1, 3, 5, or 7) is selected with the RPBBLK bit in MCR	RCE16 RCE17 ... RCE31	Channel *m* Channel (*m* + 1) ... Channel (*m* + 15)

(1) *n* is any even-numbered block 0, 2, 4, or 6. *m* is any odd-numbered block 1, 3, 5, or 7.

Table 11-1610. Use of the Receive Channel Enable Registers (continued)

	Block Assignments		Channel Assignments	
Number of selectable channels	**RCERE n**	**Block assigned[1]**	**Bit in RCERE n**	**Channel assigned[1]**
128 (RMCME = 1)	RCERE0	Block 0	RCE0 ... RCE15	Channel 0 ... Channel 15
		Block 1	RCE16 ... RCE31	Channel 16 ... Channel 31
	RCERE1	Block 2	RCE0 ... RCE15	Channel 32 ... Channel 47
		Block 3	RCE16 ... RCE31	Channel 48 ... Channel 63
	RCERE2	Block 4	RCE0 ... RCE15	Channel 64 ... Channel 79
		Block 5	RCE16 ... RCE31	Channel 80 ... Channel 95
	RCERE3	Block 6	RCE0 ... RCE15	Channel 96 ... Channel 111
		Block 7	RCE16 ... RCE31	Channel 112 ... Channel 127

11.10.4.7.8 Transmit Multi-channel Selection Mode

The XMCM bit in the multi-channel control register (MCBSP_MCR) determines whether all channels or only selected channels are enabled and unmasked for transmission. The McBSP has three transmit multi-channel selection modes (XMCM = 1, XMCM = 2h, and XMCM = 3h), which are described in Table 11-1611.

Table 11-1611. Selecting a Transmit Multi-channel Selection Mode With the XMCM Bits

XMCM Bit in MCBSP_MCR	Transmit Multi-channel Selection Mode
0	No transmit multi-channel selection mode is on. All channels are enabled and unmasked. No channels can be disabled or masked.
1h	All channels are disabled unless they are selected in the appropriate enhanced transmit channel enable register (XCERE *n*). If enabled, a channel in this mode is also unmasked.
	The XMCME bit in MCBSP_MCR determines whether 32 channels or 128 channels are selectable in XCERE *n*.
2h	All channels are enabled, but they are masked unless they are selected in the appropriate enhanced transmit channel enable register (XCERE *n*).
	The XMCME bit in MCBSP_MCR determines whether 32 channels or 128 channels are selectable in XCERE *n*.
3h	This mode is used for symmetric transmission and reception. All channels are disabled for transmission unless they are enabled for reception in the appropriate enhanced receive channel enable register (RCERE *n*). Once enabled, they are masked unless they are also selected in the appropriate enhanced transmit channel enable register (XCERE *n*).

Table 11-1611. Selecting a Transmit Multi-channel Selection Mode With the XMCM Bits (continued)

XMCM Bit in MCBSP_MCR	Transmit Multi-channel Selection Mode
	The XMCME bit in MCBSP_MCR determines whether 32 channels or 128 channels are selectable in RCERE *n* and XCERE *n*.

As an example of how the McBSP behaves in a transmit multi-channel selection mode, suppose that XMCM = 1 (all channels disabled unless individually enabled) and only channels 0,15, and 39 have been enabled. Suppose also that the frame length is 40. The McBSP:

1. Shifts data to the DX pin in channel 0.
2. Places the DX pin in the high-impedance state in channels 1–14.
3. Shifts data to the DX pin in channel 15.
4. Places the DX pin in the high-impedance state in channels 16–38.
5. Shifts data to the DX pin in channel 39.

11.10.4.7.8.1 Disabling/Enabling Versus Masking/Unmasking

For transmission, a channel can be:

- Enabled and unmasked (transmission can begin and can be completed)
- Enabled but masked (transmission can begin but cannot be completed)
- Disabled (transmission cannot occur)

The following definitions explain the channel control options:

Enabled channel — A channel that can begin transmission by passing data from the data transmit register (DXR) to the transmit shift register (XSR).

Masked channel — A channel that cannot complete transmission. The DX pin is held in the high-impedance state; data cannot be shifted out on the DX pin.

In systems where symmetric transmit and receive provides software benefits, this feature allows transmit channels to be disabled on a shared serial bus. A similar feature is not needed for reception because multiple receptions cannot cause serial bus contention.

Disabled channel — A channel that is not enabled. A disabled channel is also masked.

Because no DXR-to-XSR copy occurs, the XRDY bit in MCBSP_SPCR is not set. Therefore, no DMA synchronization event (XEVT) is generated, and if the transmit interrupt mode depends on XRDY (XINTM = 0 in MCBSP_SPCR), no interrupt is generated.

The XEMPTY bit in MCBSP_SPCR is not affected.

Unmasked channel — A channel that is not masked. Data in the XSR is shifted out on the DX pin.

11.10.4.7.8.2 Activity on McBSP Pins for Different Values of XMCM

Figure 11-752 shows the activity on the McBSP pins for the various XMCM values. In all cases, the transmit frame is configured as follows:

- In transmit control register (MCBSP_XCR):
  - XPHASE = 0: Single-phase frame (required for multi-channel selection modes)
  - XFRLEN1 = 3h: 4 words per frame
  - XWDLEN1 = 0: 8 bits per word
- In multi-channel control register (MCBSP_MCR):
  - XMCME = 0: 2-partition mode (only partitions A and B used)

In the case where XMCM = 3h, transmission and reception are symmetric, which means the corresponding bits for the receiver (RPHASE, RFRLEN1, RWDLEN1, and RMCME) must have the same values as XPHASE, XFRLEN1, and XWDLEN1, respectively.

In Figure 11-752, the arrows showing where the various events occur are only sample indications. Wherever possible, there is a time window in which these events can occur.

11.10.4.7.8.3 XCEREn Registers Used in Transmit Multi-channel Selection Mode

For multi-channel selection operation, the assignment of channels to the enhanced transmit channel enable register (XCERE *n*) depends on whether 32 or 128 channels are individually selectable, as defined by the XMCME bit in the multi-channel control register (MCR). For each of these two cases, Table 11-1612 shows which block of channels is assigned to each XCERE *n* used. For each XCERE *n*, Table 11-1612 shows which channel is assigned to each of the bits.

When XMCM = 3h (symmetric transmission and reception), the transmitter uses the enhanced receive channel enable register (RCERE *n*) to enable channels and uses XCERE *n* to unmask channels for transmission.

Table 11-1612. Use of the Transmit Channel Enable Registers

Number of selectable channels	Block Assignments		Channel Assignments	
	XCERE n	Block assigned[1]	Bit in XCERE n	Channel assigned[1]
32 (XMCME = 0)	XCERE *0*	Channels *n* to (*n* + 15)	XCE0	Channel *n*
		When XMCM = 1h or 2h, the block of channels (0, 2, 4, or 6) is selected with the XPABLK bit in MCR.	XCE1	Channel (*n* + 1)
		When XMCM = 3h, the block of channels (0, 2, 4, or 6) is selected with the RPABLK bit in MCR.	...	...
			XCE15	Channel (*n* + 15)
		Channels *m* to (*m* + 15)	XCE16	Channel *m*
		When XMCM = 1h or 2h, the block of channels (1, 3, 5, or 7) is selected with the XPBBLK bit in MCR.	XCE17	Channel (*m* + 1)
		When XMCM = 3h, the block of channels (1, 3, 5, or 7) is selected with the RPBBLK bit in MCR.	...	...
			XCE31	Channel (m + 15)

(1) *n* is any even-numbered block 0, 2, 4, or 6. *m* is any odd-numbered block 1, 3, 5, or 7.

Table 11-1612. Use of the Transmit Channel Enable Registers (continued)

Number of selectable channels	Block Assignments		Channel Assignments	
	XCERE n	Block assigned[1]	Bit in XCERE n	Channel assigned[1]
128 (XMCME = 1)	XCERE0	Block 0	XCE0 ... XCE15	Channel 0 ... Channel 15
		Block 1	XCE16 ... XCE31	Channel 16 ... Channel 31
	XCERE1	Block 2	XCE0 ... XCE15	Channel 32 ... Channel 47
		Block 3	XCE16 ... XCE31	Channel 48 ... Channel 63
	XCERE2	Block 4	XCE0 ... XCE15	Channel 64 ... Channel 79
		Block 5	XCE16 ... XCE31	Channel 80 ... Channel 95
	XCERE3	Block 6	XCE0 ... XCE15	Channel 96 ... Channel 111
		Block 7	XCE16 ... XCE31	Channel 112 ... Channel 127

11.10.4.7.9 Using Interrupts Between Block Transfers

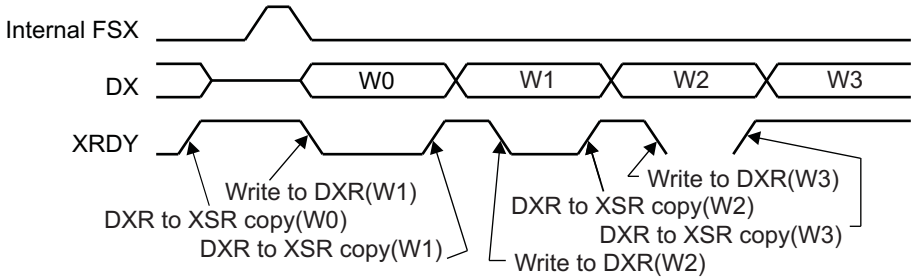
When a multi-channel selection mode is used, an interrupt request can be sent to the CPU at the end of every 16-channel block (at the boundary between partitions and at the end of the frame). In the receive multi-channel selection mode, a receive interrupt (RINT) request is generated at the end of each block transfer if the RINTM bit in the serial port control register (MCBSP_SPCR) is set to 1. In any of the transmit multi-channel selection modes, a transmit interrupt (XINT) request is generated at the end of each block transfer if the XINTM bit in MCBSP_SPCR is set to 1. When RINTM/XINTM = 1, no interrupt is generated unless a multi-channel selection mode is on.

These interrupt pulses are active high and last for two McBSP internal input clock cycles.

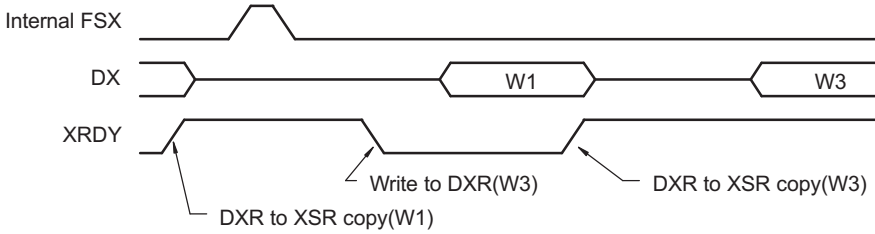
This type of interrupt is especially helpful if the two-partition mode is used and want to know when to assign a different block of channels to partition A or B.

Figure 11-752. Activity on McBSP Pins for the Possible Values of XMCM

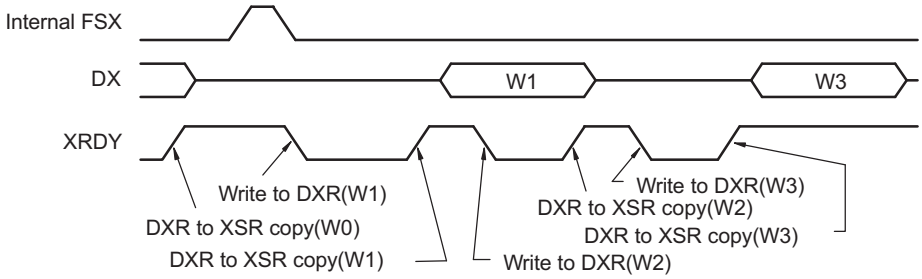
(a) XMCM = 0: All channels enabled and unmasked



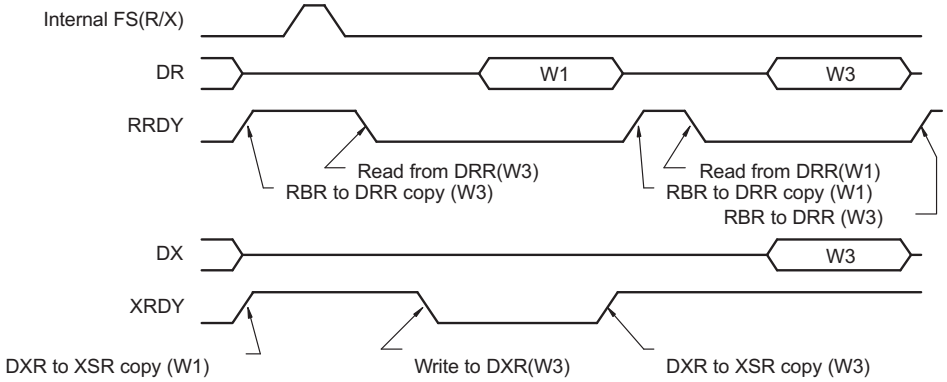
(b) XMCM = 1h, XPABLK = 0, XCERE0 = 0000 000Ah: Only channels 1 and 3 enabled and unmasked



(c) XMCM = 2h, XPABLK = 0, XCERE0 = 0000 000Ah: All channels enabled, only 1 and 3 unmasked



(d) XMCM = 3h, RPABLK = 0, XPABLK = x, RCERE0 = 0000 0008h, XCERE0 = 0000 000Ah: Receive channels: 1 and 3 enabled; transmit channels: 1 and 3 enabled, but only 3 unmasked



11.10.4.8 μ-Law/A-Law Companding Hardware Operation

Companding (compressing and expanding) hardware allows compression and expansion of data in either μ-law or A-law format. The specification for μ-law and A-law log PCM is part of the CCITT G.711 recommendation. The companding standard employed in the United States and Japan is μ-law and allows 14 bits of dynamic range. The European companding standard is A-law and allows 13 bits of dynamic range. Any values outside these ranges are set to the most positive or most negative value. Thus, for companding to work best, the data transferred to and from the McBSP via the CPU or the EDMA controller must be at least 16 bits wide.

The μ-law and A-law formats encode data into 8-bit code elements. Companded data is always 8 bits wide, so the appropriate (R/X)WDLEN1/2 must be cleared to 0, indicating an 8-bit serial data stream. If companding is enabled and either phase of the frame does not have an 8-bit element length, companding continues as if the element length is eight bits.

When companding is used, transmit data is encoded according to the specified companding law, and receive data is decoded to 2s-complement format. Companding is enabled and the desired format is selected by appropriately setting (R/X)COMPAND in the (R/X)CR. Compression occurs during the process of copying data from MCBSP_DXR to XSR and expansion occurs from RBR to MCBSP_DRR, as shown in Figure 11-753.

For transmit data to be compressed, it should be 16-bit, left-justified data, such as LAW16, as shown in Figure 11-754. The value can be either 13 or 14 bits wide, depending on the companding law. This 16-bit data is aligned in MCBSP_DXR, as shown in Figure 11-755.

For reception, the 8-bit compressed data in RBR is expanded to a left-justified 16-bit data, LAW16. This can be further justified to 32-bit data by programming the RJUST bits in the serial port control register (MCBSP_SPCR), as shown in Table 11-1613.

Figure 11-753. Companding Flow

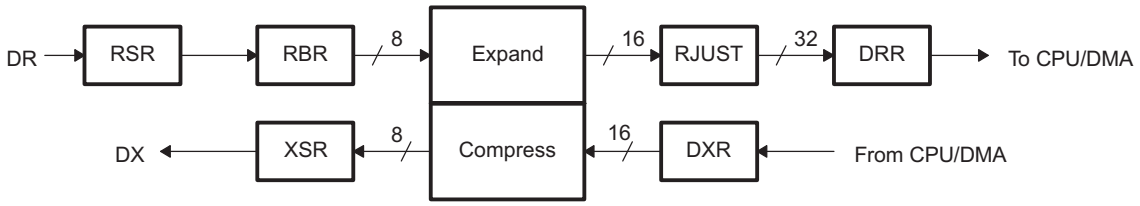


Figure 11-754. Companding Flow

LAW16	15	2	1	0
μ-Law	Value		0	0

LAW16	15	3	2	1	0
A-Law	Value		0	0	0

Figure 11-755. Transmit Data Companding Format in MCBSP_DXR

31	16	15	0
Reserved		LAW16	

Table 11-1613. Justification of Expanded Data in MCBSP_DRR

RJUST Bit in MCBSP_SPCR	MCBSP_DRR Bits			
	31	16	15	0
00	0		LAW16	
01	sign		LAW16	
10	LAW16		0	
11	Reserved			

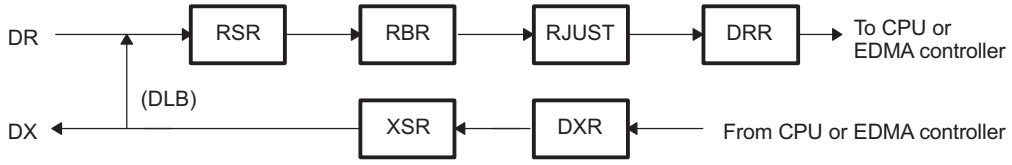
11.10.4.8.1 Companding Internal Data

If the McBSP is unused, the companding hardware can compand internal data. This hardware can be used to:

- Convert linear data to the appropriate μ-law or A-law format.
- Convert μ-law or A-law data to the linear format.
- Observe the quantization effects in companding by transmitting linear data and compressing and re-expanding this data. This is useful only if both XCOMPAND and RCOMPAND enable the same companding format.

Figure 11-756 shows the method by which the McBSP can compand internal data. The data path is indicated by the (DLB) arrow. The McBSP is enabled in digital loopback (DLB) mode with companding appropriately enabled by the RCOMPAND and XCOMPAND bits. Receive and transmit interrupts (RINT when RINTM = 0 and XINT when XINTM = 0) or synchronization events (REVT and XEVT) allow synchronization of the CPU or the EDMA controller to these conversions, respectively. The time for this companding depends on the serial bit rate selected.

Figure 11-756. Companding of Internal Data



11.10.4.8.2 Bit Ordering

Normally, all transfers on the McBSP are sent and received with the MSB first. However, certain 8-bit data protocols (that do not use companded data) require the LSB to be transferred first. By setting the (R/X)COMPAND = 01b in (R/X)CR, the bit ordering of 8-bit elements is reversed (LSB first) before being sent to the serial port. Like the companding feature, this feature is enabled only if the appropriate (R/X)WDLEN1/2 bit is cleared to 0, indicating that 8-bit elements are to be serially transferred. A 32-bit reversal feature is also available, as shown in Section 11.10.4.2.5.7.

11.10.4.9 Power Management

The McBSP can be placed in reduced power modes to conserve power during periods of low activity. The power management of the peripheral is controlled by the processor Power and Sleep Controller (PSC). The PSC acts as a master controller for power management for all of the peripherals on the device.

In order for the McBSP to be placed in power-down mode by the PSC, ensure that the XRDY and RRDY flags in the serial port control register (MCBSP_SPCR) are cleared by performing the following steps:

11.10.4.9.1 Clearing the Serial Port Control Register (MCBSP_SPCR)

1. Place the McBSP in reset by clearing the XRST, RRST, FRST, and GRST bits to 0 in MCBSP_SPCR.
   If EDMA is being used to service the transmitter and/or the receiver, disable the associated EDMA channels.
2. Switch the McBSP clocks and frames to internal clock source:
   a. Set the CLKSM and FSGM bits to 1 in the sample rate generator register (MCBSP_SRGR).
   b. Set the CLKXM, CLKRM, FSXM, and FSRM bits to 1 in the pin control register (MCBSP_PCR).
   c. Clear the SCLKME bit to 0 in MCBSP_PCR.
3. Bring the McBSP out of reset by setting the XRST, RRST, and GRST bits to 1 in MCBSP_SPCR.
4. Wait for two CLKSRG cycles for proper internal synchronization.
5. Write a dummy data value to the data transmit register (MCBSP_DXR) in order to clear the first XRDY flag.
6. Wait for at least one McBSP bit clock, since once the first dummy data value is internally copied from MCBSP_DXR to XSR, the XRDY flag transitions again from 0 to 1.
7. Write a second dummy data value to MCBSP_DXR in order to clear the second XRDY flag.
8. Check the RRDY flag in MCBSP_SPCR and if set to 1, read the data receive register (MCBSP_DRR) and discard the data to clear the RRDY flag.
9. If required, place the McBSP in power-down mode by issuing the proper PSC commands.

NOTE: After waking up the McBSP from a power-down mode using the proper PSC commands, remember to reconfigure the MCBSP_SPCR, MCBSP_SRGR, and MCBSP_PCR registers to the clock and frame combination that they were in before entering the power-down sequence and discard the two dummy data values that were used to clear the XRDY flags. If EDMA is used, re-enable the corresponding EDMA channels.

11.10.4.10 Emulation Considerations

The FREE and SOFT bits are special emulation bits in the serial port control register (MCBSP_SPCR) that determine the state of the McBSP when an emulation suspend event occurs in the emulator. An emulation suspend event corresponds to any type of emulator access to the DSP, such as a hardware or software breakpoint, a probe point, or a printf instruction.

Table 11-1614 shows the effects of the FREE and SOFT bits on the response of the McBSP to emulation suspend events.

Table 11-1614. McBSP Emulation Modes Selectable With the FREE and SOFT Bits of MCBSP_SPCR

FREE Bit in MCBSP_SPCR	SOFT Bit in MCBSP_SPCR	McBSP Emulation Mode
0	0	Immediate stop mode (reset condition). The transmitter and receiver stop immediately in response to an emulation suspend event.
0	1	Soft stop mode. When an emulation suspend event occurs, the transmitter stops after completion of the current word. The receiver is not affected.
1	0 or 1	Free run mode. The transmitter and receiver continue to run when an emulation suspend event occurs.

11.10.5 McBSP Programming Guide

This section describes the programming model of a typical McBSP module.

11.10.5.1 McBSP Initialization Procedure

The McBSP initialization procedure varies depending on the specific system setup. Section 11.10.5.1.1 provides a general initialization sequence.

The transmitter and the receiver of the McBSP can operate independently from each other. Therefore, they can be placed in or taken out of reset individually by modifying only the desired bit in the registers without disrupting the other portion. The steps in the following sections discuss the initialization procedure for taking both the transmitter and the receiver out of reset. To initialize only one portion, configure only the portion desired.

The McBSP internal sample rate generator and internal frame sync generator are shared between the transmitter and the receiver. Table 11-1615 and Table 11-1616 describe their usage based upon the clock and frame sync configurations of the receiver and transmitter, respectively.

Table 11-1615. Receiver Clock and Frame Configurations

CLKR Source	FSR Source	Comment on Configuration
Internal	Internal	The McBSP internal sample rate generator and internal frame sync generator are used by the receiver.
External	Internal	The McBSP internal sample rate generator and internal frame sync generator are used by the receiver.
Internal	External	The McBSP internal sample rate generator is used but the internal frame sync generator is not used by the receiver.
External	External	The McBSP internal sample rate generator and internal frame sync generator are not used by the receiver.

Table 11-1616. Transmitter Clock and Frame Configurations

CLKX Source	FSX Source	Comment on Configuration
Internal	Internal	The McBSP internal sample rate generator is used by the transmitter. The transmitter can generate frame sync FSX in one of two ways. First, it can generate FSX by using the internal frame sync generator (FSGM = 1). Alternatively, it can generate FSX upon each DXR-to-XSR copy (FSGM = 0). In this case, the internal frame sync generator can be kept in reset (FRST = 0) if it is not used by the receiver. The general initialization sequence can be followed in Section 11.10.5.1.1.
External	Internal	The McBSP internal sample rate generator and internal frame sync generator are not used by the transmitter. This configuration is only valid with FSGM = 0 where the McBSP transmitter generates FSX upon each DXR-to-XSR copy. The general initialization sequence can be followed in Section 11.10.5.1.1.
Internal	External	The McBSP internal sample rate generator is used by the transmitter but the internal frame sync generator is not.
External	External	The McBSP internal sample rate generator and internal frame sync generator are not used by the transmitter.

11.10.5.1.1 General Initialization Procedure

This section provides the general initialization procedure.

11.10.5.1.1.1 General Initialization

1. With the McBSP still in reset and the Power and Sleep Controller (PSC) in the default state:
   a. Program the PSC registers to put the McBSP in the enable state if required.
   b. Perform the necessary device pad multiplexing setup (see the device Data Manual).
2. Ensure that no portion of the McBSP is using the internal sample rate generator signal CLKG and the internal frame sync generator signal FSG (GRST = FRST = 0 in MCBSP_SPCR). The respective portion of the McBSP must be in reset (XRST = 0 and/or RRST = 0 in MCBSP_SPCR).
3. Program the control registers as required. Ensure the internal sample rate generator and the internal frame sync generator are still in reset (GRST = FRST = 0). Also ensure the respective portion of the McBSP is still in reset in this step (XRST = 0 and/or RRST = 0).
4. Wait for internal synchronization. If the external device provides the bit clock, wait for two CLKR or CLKX cycles. If the McBSP generates the bit clock as a clock master, wait for two CLKSRG cycles. In this case, the clock source to the sample rate generator (CLKSRG) is selected by the CLKSM bit in MCBSP_SRGR and the SCLKME bit in MCBSP_PCR.
5. Skip this step if the bit clock is provided by the external device. This step applies only if the McBSP is the bit clock master and the internal sample rate generator is used.
   a. Start the sample rate generator by setting the GRST bit to 1. Wait two CLKG bit cycles for synchronization. CLKG is the output of the sample rate generator.
   b. On the next rising edge of CLKSRG, CLKG transitions to 1 and starts clocking with a frequency equal to 1/(CLKGDV + 1) of the sample rate generator source clock CLKSRG.
6. Skip this step if the transmitter is not used. If the transmitter is used, a transmit sync error (XSYNCERR) may occur when it is enabled for the first time after device reset. The purpose of this step is to clear any potential XSYNCERR that occurs on the transmitter at this time:
   a. Set the XRST bit to 1 to enable the transmitter.
   b. Wait for any unexpected frame sync error to occur. If the external device provides the bit clock, wait for two CLKR or CLKX cycles. If the McBSP generates the bit clock as a clock master, wait for two CLKG cycles. The unexpected frame sync error (XSYNCERR), if any, occurs within this time period.
   c. Disable the transmitter (XRST = 0). This clears any outstanding XSYNCERR.
7. Setup data acquisition as required:
   a. If the EDMA is used to service the McBSP, setup data acquisition as desired and start the EDMA in this step, before the McBSP is taken out of reset.
   b. If CPU interrupt is used to service the McBSP, enable the transmit and/or receive interrupt as required.
   c. If CPU polling is used to service the McBSP, no action is required in this step.
8. Set the XRST bit and/or the RRST bit to 1 to enable the corresponding section of the McBSP. The McBSP is now ready to transmit and/or receive.
   a. If the EDMA is used to service the McBSP, it services the McBSP automatically upon receiving the XEVT and/or REVT.
   b. If CPU interrupt is used to service the McBSP, the interrupt service routine is automatically entered upon receiving the XINT and/or RINT.
   c. If CPU polling is used to service the McBSP, it can do so now by polling the XRDY and/or RRDY bit.

9. If the internal frame sync generator is used (FSGM = 1), proceed to the additional steps to turn on the internal frame sync generator. Initialization is complete if any one of the following is true:
   a. The external device generates frame sync FSX and/or FSR. The McBSP is now ready to transmit and/or receive upon receiving external frame sync.
   b. The McBSP generates transmit frame sync FSX upon each DXR-to-XSR copy. The internal frame sync generator is not used (FSGM = 0).

   The following additional steps to turn on the internal frame sync generator apply only if FSGM = 1:
10. Skip this step if the transmitter is not used. If the transmitter is used, ensure that MCBSP_DXR is serviced before the start of the internal frame sync generator. That can be done by checking XEMPTY = 1 (XSR is not empty) in MCBSP_SPCR.
11. Set the FRST bit to 1 to start the internal frame sync generator. The internal frame sync signal FSG is generated on a CLKG active edge after 7 to 8 CLKG clocks have elapsed.

11.10.5.1.2 Special Case: External Device is the Transmit Frame Master

Care must be taken if the transmitter expects a frame sync from an external device. After the transmitter comes out of reset (XRST = 1), it waits for a frame sync from the external device. If the first frame sync arrives very shortly after the transmitter is enabled, the CPU or EDMA controller may not have a chance to service the data transmit register (MCBSP_DXR). In this case, the transmitter shifts out the default data in the transmit shift register (XSR) instead of the desired value, which has not yet arrived in MCBSP_DXR. This causes problems in some applications, as the first data element in the frame is invalid. The data stream appears element-shifted (the first data word may appear in the second channel instead of the first).

To ensure proper operation when the external device is the frame master, MCBSP_DXR must be already serviced with the first word when a frame sync occurs. To do so, keep the transmitter in reset until the first frame sync is detected. Software is set up such that it will only take the transmitter out of reset (XRST = 1) promptly after detecting the first frame sync. This assures that the transmitter does not begin data transfers at the data pin during the first frame sync period. This also provides almost an entire frame period for the DSP to service MCBSP_DXR with the first word before the second frame sync occurs. The transmitter only begins data transfers upon receiving the second frame sync. At this point, MCBSP_DXR is already serviced with the first word.

11.10.5.1.2.1 How to Detect First Frame Sync

Although the McBSP is capable of generating an interrupt to the CPU upon the detection of frame synchronization (XINTM = 2h and/or RINTM = 2h in the serial port control register (MCBSP_SPCR)), the McBSP requires the associated portion (receiver/transmitter) of the McBSP to be out of reset in order for the interrupt to be generated. Therefore, instead of directly using the McBSP interrupt to detect the first frame sync, use the GPIO peripheral. This can be achieved by connecting the frame sync signal to a GPIO pin. Software can either poll the GPIO pin to detect the first frame sync or program the GPIO peripheral to generate an interrupt to the CPU upon detecting the first frame sync edge.

The following are some recommended GPIO pin(s) on the device that can be used to detect the first McBSP external frame sync:

- **GPIO pin located near the McBSP pins**. Connect the external frame sync to both the McBSP FSX/FSR pin(s) and the dedicated GPIO pin.
- **GPIO pin multiplexed with the McBSP FSX signal**. Note that on the device, the GPIO pins (of the GPIO peripheral) are multiplexed with the McBSP pins. Software can program the device's pad multiplexing register (PADCONFIG) to default these pins to the GPIO function, and only switch them to the McBSP function upon detecting the first frame sync. This method is only recommended if the external device is both the frame sync and clock master; that is, the external device drives both the FSX and CLKX signals. This method is not recommended if the McBSP is the clock master (driving CLKX and/or CLKR), as the “on-the-fly” pin multiplexed switching can cause a glitch on the CLKX/CLKR pin.

11.10.5.1.2.2 Initialization Procedure When External Device is Frame Sync Master

The initialization procedure assumes the following:

- Using a GPIO pin multiplexed with the McBSP FSX signal. If a dedicated GPIO pin is used instead, skip step 1 and step 8b.
- Software polls the GPIO pin to detect the first frame sync. If the GPIO interrupt is used instead to detect the first frame sync, step 8 can be performed within an interrupt service routine (ISR).

11.10.5.1.2.2.1 Initialization When External Device is Frame Sync Master

1. The GPIO and McBSP signals are multiplexed together on the device. Start by programming the pad multiplexing register (PADCONFIG) to select the GPIO function on the GPIO/McBSP multliplexed pins. Program the GPIO peripheral so that these pins function as GPIO inputs.
2. Ensure that no portion of the McBSP is using the internal sample rate generator signal CLKG and the internal frame sync generator signal FSG (GRST = FRST = 0 in MCBSP_SPCR). The respective portion of the McBSP needs to be in reset (XRST = 0 and/or RRST = 0 in MCBSP_SPCR).
3. Program the sample rate generator register (MCBSP_SRGR) and other control registers as required. Ensure the internal sample rate generator and the internal frame sync generator are still in reset (GRST = FRST = 0 in MCBSP_SPCR). Also ensure the respective portion of the McBSP is still in reset in this step (XRST = 0 and/or RRST = 0 in MCBSP_SPCR).
4. Wait for proper McBSP internal synchronization:
   a. If the external device provides the bit clock, wait for two CLKR or CLKX cycles. Skip step 5.
   b. If the McBSP generates the bit clock as a clock master, wait for two CLKSRG cycles. In this case, the clock source to the sample rate generator (CLKSRG) is selected by the CLKSM bit in MCBSP_SRGR.
5. Skip this step if the bit clock is provided by the external device. This step only applies if the McBSP is the bit clock master and the internal sample rate generator is used.
   a. Start the sample rate generator by setting the GRST bit in MCBSP_SPCR to 1. Wait two CLKG bit clocks for synchronization. CLKG is the output of the sample rate generator.
   b. On the next rising edge of CLKSRG, CLKG transitions to 1 and starts clocking with a frequency equal to 1/(CLKGDV + 1) of the sample rate generator source clock CLKSRG.
6. A transmit sync error (XSYNCERR) may occur when it is enabled for the first time after device reset. The purpose of this step is to clear any potential XSYNCERR that occurs on the transmitter at this time:
   a. Set the XRST bit in MCBSP_SPCR to 1 to enable the transmitter.
   b. Wait for any unexpected frame sync error to occur. If the external device provides the bit clock, wait for two CLKR or CLKX cycles. If the McBSP generates the bit clock as a clock master, wait for two CLKG cycles. The unexpected frame sync error (XSYNCERR), if any, occurs within this time period.
   c. Disable the transmitter (XRST = 0). This clears any outstanding XSYNCERR.
7. Setup data acquisition as required:
   a. If the EDMA controller is used to service the McBSP, setup data acquisition as desired and start the EDMA controller in this step, before the McBSP is taken out of reset.
   b. If the CPU interrupt is used to service the McBSP, no action is required in this step.
   c. If CPU polling is used to service the McBSP, no action is required in this step.
8. Poll the GPIO pin (through reading the appropriate registers in the GPIO peripheral) to detect the first transmit frame sync from the external device. Upon detection of the first frame sync, perform the following in this order:
   a. Set the XRST bit and/or the RRST bit to 1 to enable the respective portion of the McBSP. The McBSP is now ready to transmit and/or receive.
   b. Program PADCONFIG to switch the GPIO/McBSP multiplexed pins to the McBSP function.
9. Service the McBSP:
   a. If CPU polling is used to service the McBSP in normal operations, it can do so upon exit from the

ISR.

b. If the CPU interrupt is used to service the McBSP in normal operations, upon XRDY interrupt service routine is entered. The ISR should be setup to verify that XRDY = 1 and service the McBSP accordingly.

c. If the EDMA controller is used to service the McBSP in normal operations, it services the McBSP automatically upon receiving the XEVT and/or REVT.

10. Upon detection of the second frame sync, MCBSP_DXR is already serviced and the transmitter is ready to transmit the valid data. The receiver is also serviced properly by the DSP.

11.10.5.2 Reset and Initialization Procedure for the SRG

To reset and initialize the SRG:

1. Place the MCBSP SRG in reset.
2. Program the registers that affect the SRG.
3. Enable the SRG (take it out of reset).
4. If necessary, and/or the transmitter.
5. If necessary, remove the receiver and/or transmitter from reset.

11.10.5.3 Receiver Configuration

To configure the MCBSP receiver, perform the following steps:

Step 1. Place the MCBSP receiver in reset.

Step 2. Program the MCBSP registers for the desired receiver operation.

Step 3. Take the receiver out of reset.

11.10.5.3.1 Place the Receiver in Reset (Step 1)

Table 11-1617. Receiver Reset

Step	Register/Bit Field/Programming Model
Place the receiver in reset.	MCBSP_SPCR[0] RRST

11.10.5.3.2 Programming the MCBSP Registers for the Desired Receiver Configuration (Step 2)

This section describes the steps to be performed when software configures the MCBSP receiver.

Global Configuration

Table 11-1618 describes the steps to perform the global configuration.

Table 11-1618. Global Configuration

Step	Register/Bit Field/Programming Model
Enable/disable DLB mode(1).	MCBSP_SPCR[15] DLB
Enable/disable the receive multi-channel selection Mode(1).	MCBSP_MCR[0] RMCM

(1) Software decision

NOTE: In DLB mode the SRG and frame-sync generator must be enabled to generate the CLKX and FSX signals.

Data Configuration

Table 11-1619 describes the steps to perform the data configuration.

Table 11-1619. Data Configuration

Step	Register/Bit Field/Programming Model
Select single/dual-phase frame (1).	MCBSP_RCR[31] RPHASE
Set the receive word length(s) for phase 1(1).	MCBSP_RCR[7-5] RWDLEN1
Set the receive word length(s) for phase 2(1).	MCBSP_XCR[23-21] XWDLEN2
Set the receive frame length*.	MCBSP_RCR[14-8] RFRLEN1 MCBSP_RCR[30-24] RFRLEN2
Set the receive sign-extension and justification mode(1).	MCBSP_SPCR[14-13] RJUST
Enable the serial receiver port.	MCBSP_SPCR[0] RRST

(1) Software decision

NOTE:

- When dual-phase frame is selected, the number of words per phase must be set to 1.
- If a single-phase frame is selected, RWDLEN1 selects the length for every serial word received in the frame. If a dual-phase frame is selected, RWDLEN1 and RWDLEN2 must be set to select both lengths. These bits can have different values.
- If a dual-phase frame is selected, the frame length must be two words.

Frame-Sync Configuration

Table 11-1620 describes the steps to perform the frame-sync configuration.

Table 11-1620. Frame-Sync Configuration

Step	Register/Bit Field/Programming Model
Set the receive frame-sync mode(1).	MCBSP_PCR[10] FSRM
Select SRG synchronization(1).	MCBSP_SRGR[31] GSYNC
Set the receive frame-sync polarity (1).	MCBSP_PCR[2] FSRP
Set the SRG frame-sync period(1).	MCBSP_SRGR[27-16] FPER
Set the SRG frame-sync pulse width(1).	MCBSP_SRGR[15-8] FWID

(1) Software decision

Clock Configuration

Table 11-1621 describes the steps to perform clock configuration.

Table 11-1621. Clock Configuration

Step	Register/Bit Field/Programming Model
Set the receive clock mode[1].	MCBSP_PCR[8] CLKRM
Set the receive clock polarity[1].	MCBSP_PCR[0] CLKRP
Set the SRG clock divide-down value[1].	MCBSP_SRGR[7-0] CLKGDV
Set the SRG clock synchronization mode[1].	MCBSP_SRGR[31] GSYNC
Set the SRG input clock mode[1].	MCBSP_SRGR[13] CLKSM
Set the SRG input clock polarity[1].	MCBSP_SRGR[30] CLKSP MCBSP_PCR[1] CLKXP MCBSP_PCR[0] CLKRP

(1) Software decision

NOTE: CLKRP = CLKXP in a system in which the same clock (internal or external) is used to clock the receiver and transmitter. The receiver uses the opposite edge as the transmitter to ensure valid setup and hold of data around this edge.

11.10.5.3.3 Take the Receiver Out of Reset (Step 3)

Table 11-1622. Take the Receiver Out of Reset

Step	Register/Bit Field/Programming Model
Place the receiver in reset.	MCBSP_SPCR[0] RRST

11.10.5.4 Transmitter Configuration

To configure the MCBSP transmitter, perform the following steps:

Step 1. Place the MCBSP transmitter in reset.

Step 2. Program the MCBSP registers for the desired transmitter operation.

Step 3. Take the transmitter out of reset.

11.10.5.4.1 Place the Transmitter in Reset (Step 1)

Table 11-1623. Transmitter Reset

Step	Register/Bit Field/Programming Model
Place the transmitter in reset.	MCBSP_SPCR[16] XRST

11.10.5.4.2 Programming the MCBSP Registers for the Desired Transmitter Operation (Step 2)

This section describes the steps to be performed when software configures the MCBSP transmitter.

Global Configuration

Table 11-1624 describes the steps to perform the global configuration.

Table 11-1624. Global Configuration

Step	Register/Bit Field/Programming Model
Enable/disable DLB mode[1].	MCBSP_SPCR[15] DLB
Enable/disable the transmit multi-channel selection mode[1].	MCBSP_MCR[9] RMCM

(1) Software decision

Data Configuration

Table 11-1625 describes the steps to perform the data configuration.

Table 11-1625. Data Configuration

Step	Register/Bit Field/Programming Model
Select single/dual-phase frame [(1)].	MCBSP_XCR[31] XPHASE
Set the transmit word length(s) for phase 1[(1)].	MCBSP_XCR[7-5] XWDLEN1
Set the transmit word length(s) for phase 2[(1)].	MCBSP_XCR[23-21] XWDLEN2
Set the transmit frame length[(1)].	MCBSP_XCR[14-8] XFRLEN1 MCBSP_XCR[30-24] XFRLEN2
Set the extra delay (DX delay) mode[(1)].	MCBSP_SPCR[7] DXENA

(1) Software decision

NOTE:

- When dual-phase frame is selected, the number of words per phase must be set to 1.
- If a single-phase frame is selected, XWDLEN1 selects the length for every serial word transmit in the frame. If a dual–phase frame is selected, XWDLEN1 and XWDLEN2 must be set to select both lengths. These bits can have different values.
- If a dual-phase frame is selected, the frame length must be two words.

Frame-Sync Configuration

Table 11-1626 describes the steps to perform the frame-sync configuration.

Table 11-1626. Frame-Sync Configuration

Step	Register / Bit Field / Programming Model
Set the transmit frame-sync mode[(1)].	MCBSP_PCR[11] FSXM MCBSP_SRGR[28] FSGM
Set the transmit frame-sync polarity[(1)].	MCBSP_PCR[3] FSXP
Set the SRG frame-sync period[(1)].	MCBSP_SRGR[27-16] FPER
Set the SRG frame-sync pulse width[(1)].	MCBSP_SRGR[15-8] FWID

(1) Software decision

Clock Configuration

Table 11-1627 describes the steps to perform the clock configuration.

Table 11-1627. Clock Configuration

Step	Register / Bit Field / Programming Model
Set the transmit clock mode [(1)].	MCBSP_PCR[9] CLKXM
Set the transmit clock polarity[(1)].	MCBSP_PCR[1] CLKXP
Set the SRG clock divide-down value[(1)].	MCBSP_SRGR[7-0] CLKGDV
Set the SRG clock synchronization mode[(1)].	MCBSP_SRGR[31] GSYNC
Set the SRG input clock mode[(1)].	MCBSP_SRGR[29] CLKSM
Set the SRG input clock polarity[(1)] .	MCBSP_SRGR[30] CLKSP MCBSP_PCR[1] CLKXP MCBSP_PCR[0] CLKRP

(1) Software decision

NOTE: CLKRP = CLKXP in a system in which the same clock (internal or external) is used to clock the receiver and transmitter. The receiver uses the opposite edge as the transmitter to ensure valid setup and hold of data around this edge.

11.10.5.4.3 Take the Receiver Out of Reset (Step 3)

Table 11-1628. Take the Receiver Out of Reset

Step	Register/Bit Field/Programming Model
Enable the transmitter.	MCBSP_SPCR[0] XRST

11.10.6 McBSP Registers

Table 11-1630 lists the memory-mapped registers for the McBSP. All register offset addresses not listed in Table 11-1630 should be considered as reserved locations and the register contents should not be modified.

Table 11-1629. MCBSP Instances

Instance	Base Address
MCBSP	0234 6000h

Table 11-1630. McBSP Registers

Offset	Acronym	Register Name	MCBSP Physical Address	Section
0h	MCBSP_DRR[1][2]	Data receive register	0234 6000h	Section 11.10.6.1
4h	MCBSP_DXR[1]	Data transmit register	0234 6004h	Section 11.10.6.2
8h	MCBSP_SPCR	Serial port control register	0234 6008h	Section 11.10.6.3
Ch	MCBSP_RCR	Receive control register	0234 600Ch	Section 11.10.6.4
10h	MCBSP_XCR	Transmit control register	0234 6010h	Section 11.10.6.5
14h	MCBSP_SRGR	Sample rate generator register	0234 6014h	Section 11.10.6.6
18h	MCBSP_MCR	Multichannel control register	0234 6018h	Section 11.10.6.7
1Ch	MCBSP_RCERE0	Enhanced receive channel enable register partition A/B	0234 601Ch	Section 11.10.6.8
20h	MCBSP_XCERE0	Enhanced transmit channel enable register partition A/B	0234 6020h	Section 11.10.6.12
24h	MCBSP_PCR	Pin control register	0234 6024h	Section 11.10.6.16
28h	MCBSP_RCERE1	Enhanced receive channel enable register partition C/D	0234 6028h	Section 11.10.6.9
2Ch	MCBSP_XCERE1	Enhanced transmit channel enable register partition C/D	0234 602Ch	Section 11.10.6.13
30h	MCBSP_RCERE2	Enhanced receive channel enable register partition E/F	0234 6030h	Section 11.10.6.10
34h	MCBSP_XCERE2	Enhanced transmit channel enable register partition E/F	0234 6034h	Section 11.10.6.14
38h	MCBSP_RCERE3	Enhanced receive channel enable register partition G/H	0234 6038h	Section 11.10.6.11
3Ch	MCBSP_XCERE3	Enhanced transmit channel enable register partition G/H	0234 603Ch	Section 11.10.6.15
0h	MCBSP_BFIFOREV[3]	BFIFO Revision Identification Register	0234 6080h	Section 11.10.6.17
10h	MCBSP_WFIFOCTL[3]	Write FIFO Control Register	0234 6090h	Section 11.10.6.18
14h	MCBSP_WFIFOSTS[3]	Write FIFO Status Register	0234 6094h	Section 11.10.6.19
18h	MCBSP_RFIFOCTL[3]	Read FIFO Control Register	0234 6098h	Section 11.10.6.20

(1) The MCBSP_DRR and MCBSP_DXR are accessible via the CPUs or the EDMA controller.
(2) The CPU and EDMA controller can only read this register; they cannot write to it.
(3) The McBSP Buffer FIFO (BFIFO) has a different memory-map than the McBSP memory-mapped registers (MMRs); therefore, the BFIFO is accessible by way of a different Configuration Bus.

Table 11-1630. McBSP Registers (continued)

Offset	Acronym	Register Name	MCBSP Physical Address	Section
1Ch	MCBSP_RFIFOSTS[3]	Read FIFO Status Register	0234 609Ch	Section 11.10.6.21

11.10.6.1 MCBSP_DRR Register (Offset = 0h) [reset = 0h]

The data receive register (MCBSP_DRR) contains the value to be written to the data bus. The MCBSP_DRR is shown in Figure 11-757 and described in Table 11-1632.

Both the CPUs and the EDMA can access MCBSP_DRR in all the memory-mapped locations. An access to any EDMA bus location is equivalent to an access to MCBSP_DRR of the corresponding McBSP.

Table 11-1631. MCBSP_DRR Instances

Instance	Physical Address
MCBSP	0234 6000h

Figure 11-757. MCBSP_DRR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1632. MCBSP_DRR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DR	R	0h	Data receive register value to be written to the data bus. Value is 0-FFFF FFFFh

Table 11-1633. Register Call Summary for MCBSP_DRR

McBSP Functional Description • Receive Operation: [0][1] • Receive Multi-channel Selection Mode: [0] • Receive Data Justification and Sign Extension: RJUST: [0][1][2] • Data Packing Using Frame Length and Element Length: [0][1] • Receive Ready Status: REVT and RRDY: [0][1] • μ-Law/A-Law Companding Hardware Operation: [0][1] • Receive Ready Status: RINT and RRDY: [0][1] • Interrupt Events and Requests: [0] • Clearing the Serial Port Control Register (MCBSP_SPCR): [0] • Unexpected Receive Frame Synchronization: RSYNCERR: [0] • Receive Overrun: RFULL: [0][1][2][3][4][5][6][7][8]
McBSP Registers • MCBSP_SPCR Register (Offset = 8h) [reset = 0h]: [0][1][2][3][4] • MCBSP_DRR Register (Offset = 0h) [reset = 0h]: [0][1][2][3] • McBSP Registers: [0][1]

11.10.6.2 MCBSP_DXR Register (Offset = 4h) [reset = 0h]

The data transmit register (MCBSP_DXR) contains the value to be loaded into the data transmit shift register (XSR). The MCBSP_DXR is shown in Figure 11-758 and described in Table 11-1635.

MCBSP_DXR is accessible via the peripheral bus and via the EDMA bus. Both the CPUs and the EDMA can access MCBSP_DXR in all the memory-mapped locations.

Table 11-1634. MCBSP_DXR Instances

Instance	Physical Address
MCBSP	0234 6004h

Figure 11-758. MCBSP_DXR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DX																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1635. MCBSP_DXR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DX	R/W	0h	Data transmit register value to be loaded into the data transmit shift register (XSR). Value is 0-FFFF FFFFh.

Table 11-1636. Register Call Summary for MCBSP_DXR

McBSP Functional Description • Clearing the Serial Port Control Register (MCBSP_SPCR): [0][1][2] • Interrupt Events and Requests: [0] • Data Packing Using Frame Length and Element Length: [0][1] • Transmit With Data Overwrite: [0][1][2][3] • Transmit Ready Status: XEVT and XRDY: [0][1][2][3][4][5] • μ-Law/A-Law Companding Hardware Operation: [0][1] • Transmit Empty: XEMPTY: [0][1][2][3][4][5][6][7][8][9] • Transmit Ready Status: XINT and XRDY: [0][1][2][3][4][5] • Transmit Operation: [0][1] • Stopping Clocks: [0][1][2][3]
McBSP Registers • MCBSP_SPCR Register (Offset = 8h) [reset = 0h]: [0] • MCBSP_DXR Register (Offset = 4h) [reset = 0h]: [0][1][2][3] • McBSP Registers: [0][1]
McBSP Programming Guide • General Initialization: [0] • Special Case: External Device is the Transmit Frame Master: [0][1][2][3][4] • Initialization Procedure When External Device is Frame Sync Master: [0]

11.10.6.3 MCBSP_SPCR Register (Offset = 8h) [reset = 0h]

The serial port is configured via the serial port control register (MCBSP_SPCR) and the pin control register (MCBSP_PCR). The MCBSP_SPCR contains McBSP status control bits. The MCBSP_SPCR is shown in Figure 11-759 and described in Table 11-1638.

Table 11-1637. MCBSP_SPCR Instances

Instance	Physical Address
MCBSP	0234 6008h

Figure 11-759. MCBSP_SPCR Register

31	30	29	28	27	26	25	24
RESERVED						FREE	SOFT
R-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
FRST	GRST	XINTM		XSYNCERR	XEMPTY	XRDY	XRST
R/W-0h	R/W-0h	R/W-0h		R/W-0h	R-0h	R-0h	R/W-0h
15	14	13	12	11	10	9	8
DLB	RJUST		CLKSTP		RESERVED		
R/W-0h	R/W-0h		R-0h		R-0h		
7	6	5	4	3	2	1	0
DXENA	RESERVED	RINTM		RSYNCERR	RFULL	RRDY	RRST
R/W-0h	R-0h	R/W-0h		R/W-0h	R-0h	R-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1638. MCBSP_SPCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
25	FREE	R/W	0h	Free-running enable mode bit. This bit is used in conjunction with SOFT bit to determine state of serial port clock during emulation halt. • 0 = Free-running mode is disabled. During emulation halt, SOFT bit determines operation of McBSP. • 1 = Free-running mode is enabled. During emulation halt, serial clocks continue to run.
24	SOFT	R/W	0h	Soft bit enable mode bit. This bit is used in conjunction with FREE bit to determine state of serial port clock during emulation halt. This bit has no effect if FREE = 1. • 0 = Soft mode is disabled. Serial port clock stops immediately during emulation halt, thus aborting any transmissions. • 1 = Soft mode is enabled. During emulation halt, serial port clock stops after completion of current transmission.
23	FRST	R/W	0h	Frame-sync generator reset bit. • 0 = Frame-synchronization logic is reset. Frame-sync signal (FSG) is not generated by the sample-rate generator. • 1 = Frame-sync signal (FSG) is generated after (FPER + 1) number of CLKG clocks; that is, all frame counters are loaded with their programmed values.
22	GRST	R/W	0h	Sample-rate generator reset bit. • 0 = Sample-rate generator is reset. • 1 = Sample-rate generator is taken out of reset. CLKG is driven as per programmed value in sample-rate generator register (MCBSP_SRGR).

Table 11-1638. MCBSP_SPCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	XINTM	R/W	0h	Transmit interrupt (XINT) mode bit. • 0 = XINT is driven by XRDY (end-of-word). • 1h = Reserved • 2h = XINT is generated by a new frame synchronization. • 3h = XINT is generated by XSYNCERR.
19	XSYNCERR	R/W	0h	Transmit synchronization error bit. Writing a 1 to XSYNCERR sets the error condition when the transmitter is enabled (XRST = 1). Thus, it is used mainly for testing purposes or if this operation is desired. • 0 = No synchronization error is detected. • 1 = Synchronization error is detected.
18	XEMPTY	R	0h	Transmit shift register empty bit. • 0 = XSR is empty. • 1 = XSR is not empty.
17	XRDY	R	0h	Transmitter ready bit. • 0 = Transmitter is not ready. • 1 = Transmitter is ready for new data in MCBSP_DXR.
16	XRST	R/W	0h	Transmitter reset bit resets or enables the transmitter. • 0 = Serial port transmitter is disabled and in reset state. • 1 = Serial port transmitter is enabled.
15	DLB	R/W	0h	Digital loop back mode enable bit. • 0 = Digital loop back mode is disabled. • 1 = Digital loop back mode is enabled.
14-13	RJUST	R/W	0h	Receive sign-extension and justification mode bit. • 0 = Right-justify and zero-fill MSBs in MCBSP_DRR. • 1h = Right-justify and sign-extend MSBs in MCBSP_DRR. • 2h = Left-justify and zero-fill LSBs in MCBSP_DRR. • 3h = Reserved
12-11	CLKSTP	R	0h	Clock stop mode bit. • 0 = Clock stop mode is disabled. Normal clocking for non-SPI mode. • 1h-3h = Reserved
10-8	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
7	DXENA	R/W	0h	DX enabler bit. See Section 11.10.4.7.4 for details on the DX enabler bit. • 0 = DX enabler is off. • 1 = DX enabler is on.
6	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
5-4	RINTM	R/W	0h	Receive interrupt (RINT) mode bit. • 0 = RINT is driven by RRDY (end-of-word). • 1h = Reserved • 2h = RINT is generated by a new frame synchronization. • 3h = RINT is generated by RSYNCERR.
3	RSYNCERR	R/W	0h	Receive synchronization error bit. Writing a 1 to RSYNCERR sets the error condition when the receiver is enabled (RRST = 1). Thus, it is used mainly for testing purposes or if this operation is desired. • 0 = No synchronization error is detected. • 1 = Synchronization error is detected.
2	RFULL	R	0h	Receive shift register full bit. • 0 = RBR is not in overrun condition. • 1 = MCBSP_DRR is not read, RBR is full, and RSR is also full with new word.

Table 11-1638. MCBSP_SPCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	RRDY	R	0h	Receiver ready bit. • 0 = Receiver is not ready. • 1 = Receiver is ready with data to be read from MCBSP_DRR.
0	RRST	R/W	0h	Receiver reset bit resets or enables the receiver. • 0 = The serial port receiver is disabled and in reset state. • 1 = The serial port receiver is enabled.

Table 11-1639. Register Call Summary for MCBSP_SPCR

McBSP Functional Description
- Receive Data Justification and Sign Extension: RJUST: [0][1][2]
- Interrupt Events: RINT and XINT: [0][1]
- Resetting the Serial Port: RRST, XRST, GRST, and RESET: [0][1][2][3]
- Software Reset Considerations: [0][1]
- Hardware Reset Considerations: [0]
- Power Management: [0][1]
- Emulation Considerations: [0][1][2]
- Receive Ready Status: RINT and RRDY: [0]
- Interrupt Events and Requests: [0][1]
- Transmit Ready Status: XINT and XRDY: [0]
- μ-Law/A-Law Companding Hardware Operation: [0][1]
- Transmit Operation: [0]
- Unexpected Receive Frame Synchronization: RSYNCERR: [0][1][2][3]
- Frame Sync Ignore and Unexpected Frame Sync Pulses: [0][1]
- Serial Port Exception Conditions: [0][1][2][3]
- Receive Overrun: RFULL: [0]
- Sample Rate Generator Clocking and Framing: [0]
- Receive Operation: [0]
- Using Interrupts Between Block Transfers: [0][1]
- DX Enabler: DXENA: [0]
- Bit Clock and Frame Synchronization: [0]
- Receive Ready Status: REVT and RRDY: [0][1]
- Transmit Ready Status: XEVT and XRDY: [0][1]
- Digital Loopback Mode: DLB: [0][1]
- Disabling/Enabling Versus Masking/Unmasking: [0][1][2]
- Receive Clock Selection: DLB, CLKRM: [0][1]
- Frame Period (FPER) and Frame Width (FWID): [0]
- Frame Detection: [0]
- Receive Frame Synchronization Selection: DLB and FSRM: [0][1]
- Frame Sync Generation: [0]
- Clearing the Serial Port Control Register (MCBSP_SPCR): [0][1][2]
- Unexpected Transmit Frame Synchronization: XSYNCERR: [0][1][2][3]

McBSP Registers
- MCBSP_SPCR Register (Offset = 8h) [reset = 0h]: [0][1][2]
- MCBSP_PCR Register (Offset = 24h) [reset = 0h]: [0][1][2][3]
- McBSP Registers: [0]

Table 11-1639. Register Call Summary for MCBSP_SPCR (continued)

McBSP Programming Guide
• General Initialization: [0][1][2] • Take the Receiver Out of Reset (Step 3): [0] • How to Detect First Frame Sync: [0] • Take the Receiver Out of Reset (Step 3): [0] • Initialization Procedure When External Device is Frame Sync Master: [0][1][2][3][4][5] • Place the Transmitter in Reset (Step 1): [0] • Place the Receiver in Reset (Step 1): [0] • Programming the MCBSP Registers for the Desired Receiver Configuration (Step 2): [0][1][2] • Programming the MCBSP Registers for the Desired Transmitter Operation (Step 2): [0][1]

11.10.6.4 MCBSP_RCR Register (Offset = Ch) [reset = 0h]

The receive control register (MCBSP_RCR) configures parameters of the receive operations. The MCBSP_RCR is shown in Figure 11-760 and described in Table 11-1641.

Table 11-1640. MCBSP_RCR Instances

Instance	Physical Address
MCBSP	0234 600Ch

Figure 11-760. MCBSP_RCR Register

31	30	29	28	27	26	25	24
RPHASE	RFRLEN2						
R/W-0h	R/W-0h						
23	22	21	20	19	18	17	16
RWDLEN2			RCOMPAND		RFIG	RDATDLY	
R/W-0h			R/W-0h		R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	RFRLEN1						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
RWDLEN1			RWDREVRS	RESERVED			
R/W-0h			R/W-0h	R-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1641. MCBSP_RCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RPHASE	R/W	0h	Receive phases bit. • 0 = Single-phase frame • 1 = Dual-phase frame
30-24	RFRLEN2	R/W	0h	Specifies the receive frame length (number of words) in phase 2. • 0 = 1 word in phase 2 • 1h = 2 words in phase 2 • 2h = 3 words in phase 2 • ... • 7Fh = 128 words in phase 2
23-21	RWDLEN2	R/W	0h	Specifies the receive word length (number of bits) in phase 2. • 0 = Receive word length is 8 bits. • 1h = Receive word length is 12 bits. • 2h = Receive word length is 16 bits. • 3h = Receive word length is 20 bits. • 4h = Receive word length is 24 bits. • 5h = Receive word length is 32 bits. • 6h-7h = Reserved
20-19	RCOMPAND	R/W	0h	Receive companding mode bit. Modes other than 00 are only enabled when RWDLEN1/2 bit is 000 (indicating 8-bit data). • 0 = No companding, data transfer starts with MSB first. • 1h = No companding, 8-bit data transfer starts with LSB first. • 2h = Compand using ì-law for receive data. • 3h = Compand using A-law for receive data.
18	RFIG	R/W	0h	Receive frame ignore bit. • 0 = Receive frame-synchronization pulses after the first pulse restarts the transfer. • 1 = Receive frame-synchronization pulses after the first pulse are ignored.

Table 11-1641. MCBSP_RCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17-16	RDATDLY	R/W	0h	Receive data delay bit. • 0 = 0-bit data delay • 1h = = 1-bit data delay • 2h = 2-bit data delay • 3h = Reserved
15	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
14-8	RFRLEN1	R/W	0h	Specifies the receive frame length (number of words) in phase 1. • 0 = 1 word in phase 1 • 1h = 2 words in phase 1 • 2h = 3 words in phase 1 • ... • 7Fh = 128 words in phase 1
7-5	RWDLEN1	R/W	0h	Specifies the receive word length (number of bits) in phase 1. • 0 = Receive word length is 8 bits. • 1h = Receive word length is 12 bits. • 2h = Receive word length is 16 bits. • 3h = Receive word length is 20 bits. • 4h = Receive word length is 24 bits. • 5h = Receive word length is 32 bits. • 6h-7h = Reserved
4	RWDREVRS	R/W	0h	Receive 32-bit reversal enable bit. • 0 = 32-bit reversal is disabled. • 1 = 32-bit reversal is enabled. 32-bit data is received LSB first. RWDLEN1/2 bit should be set to 5h (32-bit operation); RCOMPAND bit should be set to 1h (transfer starts with LSB first); otherwise, operation is undefined.
3-0	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

Table 11-1642. Register Call Summary for MCBSP_RCR

McBSP Functional Description • Frame Synchronization Phases: [0][1][2] • Receive Operation: [0] • 32-Bit Reversal: RWDREVRS, XWDREVRS: [0][1][2] • Element Length: RWDLEN1/2 and XWDLEN1/2: [0] • Data Delay: RDATDLY and XDATDLY: [0]
McBSP Registers • McBSP Registers: [0] • MCBSP_RCR Register (Offset = Ch) [reset = 0h]: [0][1]
McBSP Programming Guide • Programming the MCBSP Registers for the Desired Receiver Configuration (Step 2): [0][1][2][3]

11.10.6.5 MCBSP_XCR Register (Offset = 10h) [reset = 0h]

The transmit control register (MCBSP_XCR) configures parameters of the transmit operations. The MCBSP_XCR is shown in Figure 11-761 and described in Table 11-1644.

Table 11-1643. MCBSP_XCR Instances

Instance	Physical Address
MCBSP	0234 6010h

Figure 11-761. MCBSP_XCR Register

31	30	29	28	27	26	25	24
XPHASE	XFRLEN2						
R/W-0h	R/W-0h						
23	22	21	20	19	18	17	16
XWDLEN2			XCOMPAND		XFIG	XDATDLY	
R/W-0h			R/W-0h		R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	XFRLEN1						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
XWDLEN1			XWDREVRS	RESERVED			
R/W-0h			R/W-0h	R-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1644. MCBSP_XCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	XPHASE	R/W	0h	Transmit phases bit. • 0 = Single-phase frame • 1 = Dual-phase frame
30-24	XFRLEN2	R/W	0h	Specifies the transmit frame length (number of words) in phase 2. • 0 = 1 word in phase 2 • 1h = 2 words in phase 2 • 2h = 3 words in phase 2 • ... • 7Fh = 128 words in phase 2
23-21	XWDLEN2	R/W	0h	Specifies the transmit word length (number of bits) in phase 2. • 0 = Transmit word length is 8 bits. • 1h = Transmit word length is 12 bits. • 2h = Transmit word length is 16 bits. • 3h = Transmit word length is 20 bits. • 4h = Transmit word length is 24 bits. • 5h = Transmit word length is 32 bits. • 6h-7h = Reserved
20-19	XCOMPAND	R/W	0h	Transmit companding mode bit. Modes other than 00 are only enabled when XWDLEN1/2 bit is 000 (indicating 8-bit data). • 0 = No companding, data transfer starts with MSB first. • 1h = No companding, 8-bit data transfer starts with LSB first. • 2h = Compand using ì-law for transmit data. • 3h = Compand using A-law for transmit data.
18	XFIG	R/W	0h	Transmit frame ignore bit. • 0 = Transmit frame-synchronization pulses after the first pulse restarts the transfer. • 1 = Transmit frame-synchronization pulses after the first pulse are ignored.

Table 11-1644. MCBSP_XCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17-16	XDATDLY	R/W	0h	Transmit data delay bit. • 0 = 0-bit data delay • 1h = 1-bit data delay • 2h = 2-bit data delay • 3h = Reserved
15	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
14-8	XFRLEN1	R/W	0h	Specifies the transmit frame length (number of words) in phase 1. • 0 = 1 word in phase 1 • 1h = 2 words in phase 1 • 2h = 3 words in phase 1 • ... • 7Fh = 128 words in phase 1
7-5	XWDLEN1	R/W	0h	Specifies the transmit word length (number of bits) in phase 1. • 0 = Transmit word length is 8 bits. • 1h = Transmit word length is 12 bits. • 2h = Transmit word length is 16 bits. • 3h = Transmit word length is 20 bits. • 4h = Transmit word length is 24 bits. • 5h = Transmit word length is 32 bits. • 6h-7h = Reserved
4	XWDREVRS	R/W	0h	Transmit 32-bit reversal feature enable bit. • 0 = 32-bit reversal is disabled. • 1 = 32-bit reversal is enabled. 32-bit data is transmitted LSB first. XWDLEN1/2 bit should be set to 5h (32-bit operation); XCOMPAND bit should be set to 1h (transfer starts with LSB first); otherwise, operation is undefined.
3-0	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

Table 11-1645. Register Call Summary for MCBSP_XCR

McBSP Functional Description • Activity on McBSP Pins for Different Values of XMCM: [0] • Data Delay: RDATDLY and XDATDLY: [0] • Frame Synchronization Phases: [0][1][2] • Element Length: RWDLEN1/2 and XWDLEN1/2: [0] • Transmit Operation: [0] • 32-Bit Reversal: RWDREVRS, XWDREVRS: [0][1][2]
McBSP Registers • McBSP Registers: [0] • MCBSP_XCR Register (Offset = 10h) [reset = 0h]: [0][1]
McBSP Programming Guide • Programming the MCBSP Registers for the Desired Receiver Configuration (Step 2): [0] • Programming the MCBSP Registers for the Desired Transmitter Operation (Step 2): [0][1][2][3][4]

11.10.6.6 MCBSP_SRGR Register (Offset = 14h) [reset = 20000001h]

The sample rate generator register (MCBSP_SRGR) controls the operation of various features of the sample rate generator. The MCBSP_SRGR is shown in Figure 11-762 and described in Table 11-1647.

Table 11-1646. MCBSP_SRGR Instances

Instance	Physical Address
MCBSP	0234 6014h

Figure 11-762. MCBSP_SRGR Register

31	30	29	28	27	26	25	24
GSYNC	CLKSP	CLKSM	FSGM	FPER			
R/W-0h	R/W-0h	R/W-1h	R/W-0h	R/W-0h			
23	22	21	20	19	18	17	16
FPER							
R/W-0h							
15	14	13	12	11	10	9	8
FWID							
R/W-0h							
7	6	5	4	3	2	1	0
CLKGDV							
R/W-1h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1647. MCBSP_SRGR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GSYNC	R/W	0h	Sample-rate generator clock synchronization bit is only used when CLKS drives the sample-rate generator clock (CLKSM = 0). • 0 = The sample-rate generator clock (CLKG) is free running. • 1 = The sample-rate generator clock (CLKG) is running; however, CLKG is resynchronized and frame-sync signal (FSG) is generated only after detecting the receive frame-synchronization signal (FSR). Also, frame period (FPER) is a don't care because the period is dictated by the external frame-sync pulse.
30	CLKSP	R/W	0h	CLKS polarity clock edge select bit is only used when CLKS drives the sample-rate generator clock (CLKSM = 0). • 0 = Rising edge of CLKS generates CLKG and FSG. • 1 = Falling edge of CLKS generates CLKG and FSG.
29	CLKSM	R/W	1h	Sample rate generator input clock mode bit. The sample rate generator can accept an input clock signal and divide it down according to CLKGDV to produce an output clock signal, CLKG. The frequency of CLKG is: *CLKG frequency = Input clock frequency/(CLKGDV + 1)* CLKSM is used in conjunction with the SCLKME bit in the pin control register (MCBSP_PCR) to determine the source for the input clock. A DSP reset selects the McBSP internal input clock as the input clock and forces the CLKG frequency to 1/2 the McBSP internal input clock frequency. • 0 = The input clock for the sample rate generator is taken from the CLKS or from the CLKR pin, depending on the value of the SCLKME bit in MCBSP_PCR • 1 = The input clock for the sample rate generator is taken from the McBSP internal input clock or from the CLKX pin, depending on the value of the SCLKME bit in MCBSP_PCR For more information about choosing an Input Clock for Sample Rate Generator with SCLKME bit see Table 11-1595.

Table 11-1647. MCBSP_SRGR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
28	FSGM	R/W	0h	Sample-rate generator transmit frame-synchronization mode bit is only used when FSXM = 1 in MCBSP_PCR. • 0 = Transmit frame-sync signal (FSX) is generated on every DXR-to-XSR copy. When FSGM = 0, FWID bit and FPER bit are ignored. • 1 = Transmit frame-sync signal (FSX) is driven by the sample-rate generator frame-sync signal (FSG).
27-16	FPER	R/W	0h	0-FFFh = Frame period value plus 1 specifies when the next frame-sync signal becomes active. Range is 1 to 4096 sample-rate generator clock (CLKG) periods.
15-8	FWID	R/W	0h	0-FFh = Frame width value plus 1 specifies the width of the frame-sync pulse (FSG) during its active period.
7-0	CLKGDV	R/W	1h	0-FFh = Sample-rate generator clock (CLKG) divider value is used as the divide-down number to generate the required sample-rate generator clock frequency.

Table 11-1648. Register Call Summary for MCBSP_SRGR

McBSP Functional Description • Bit Clock Polarity: CLKSP: [0][1] • Bit Clock and Frame Synchronization: [0][1][2] • Transmit Frame Synchronization Selection: FSXM and FSGM: [0] • Sample Rate Generator Data Bit Clock Rate: CLKGDV: [0] • Frame Period (FPER) and Frame Width (FWID): [0][1][2][3] • Receive Frame Synchronization Selection: DLB and FSRM: [0] • Power Management: [0] • Clearing the Serial Port Control Register (MCBSP_SPCR): [0] • Frame Sync Generation: [0][1] • Input Clock Source Mode: CLKSM and SCLKME: [0][1] • Data Clock Generation: [0] • Frame and Clock Operation: [0] • Transmit Empty: XEMPTY: [0]
McBSP Registers • MCBSP_SPCR Register (Offset = 8h) [reset = 0h]: [0] • MCBSP_SRGR Register (Offset = 14h) [reset = 20000001h]: [0][1] • MCBSP_PCR Register (Offset = 24h) [reset = 0h]: [0][1][2][3] • McBSP Registers: [0]
McBSP Programming Guide • General Initialization: [0] • Initialization Procedure When External Device is Frame Sync Master: [0][1] • Programming the MCBSP Registers for the Desired Transmitter Operation (Step 2): [0][1][2][3][4][5][6] • Programming the MCBSP Registers for the Desired Receiver Configuration (Step 2): [0][1][2][3][4][5][6]

11.10.6.7 MCBSP_MCR Register (Offset = 18h) [reset = 0h]

The multichannel control register (MCBSP_MCR) has control and status bits for multichannel selection operation in the receiver (with an R prefix) and the same type of bits for the transmitter (with an X prefix). The MCBSP_MCR is shown in Figure 11-763 and described in Table 11-1650. This I/O-mapped register enables you to:

- Enable all channels or only selected channels for reception (RMCM).
- Choose which channels are enabled/disabled and masked/unmasked for transmission (XMCM).
- Specify whether two partitions (32 channels at a time) or eight partitions (128 channels at a time) can be used (RMCME for reception, XMCME for transmission).
- Assign blocks of 16 channels to partitions A and B when the 2-partition mode is selected (RPABLK and RPBBLK for reception, XPABLK and XPBBLK for transmission).
- Determine which block of 16 channels is currently involved in a data transfer (RCBLK for reception, XCBLK for transmission).

Table 11-1649. MCBSP_MCR Instances

Instance	Physical Address
MCBSP	0234 6018h

Figure 11-763. MCBSP_MCR Register

31	30	29	28	27	26	25	24
RESERVED						XMCME	XPBBLK
R-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
XPBBLK	XPABLK			XCBLK		XMCM	
R/W-0h	R/W-0h			R-0h		R/W-0h	
15	14	13	12	11	10	9	8
RESERVED						RMCME	RPBBLK
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RPBBLK	RPABLK			RCBLK		RESERVED	RMCM
R/W-0h	R/W-0h			R-0h		R-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1650. MCBSP_MCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved.
25	XMCME	R/W	0h	Transmit multichannel partition mode bit. XMCME is only applicable if channels can be individually disabled/enabled or masked/unmasked for transmission (XMCM is nonzero). XMCME determines whether only 32 channels or all 128 channels are to be individually selectable. 0 = 2-partition mode. Only partitions A and B are used. You can control up to 32 channels in the transmit multichannel selection mode selected with the XMCM bit. **If XMCM = 1 or 2h:** assign 16 channels to partition A with the XPABLK bit and assign 16 channels to partition B with the XPBBLK bit. **If XMCM = 3h:** (for symmetric transmission and reception), assign 16 channels to receive partition A with the RPABLK bit and assign 16 channels to receive partition B with the RPBBLK bit. 1 = 8-partition mode. All partitions (A through H) are used. You can control up to 128 channels in the transmit multichannel selection mode selected with the XMCM bit. You control the channels with the appropriate enhanced transmit channel enable register (XCEREn): MCBSP_XCERE0: Channels 0 through 31 MCBSP_XCERE1: Channels 32 through 63 MCBSP_XCERE2: Channels 64 through 95 MCBSP_XCERE3: Channels 96 through 127
24-23	XPBBLK	R/W	0h	Transmit partition B block bit. XPBBLK is only applicable if channels can be individually disabled/enabled and masked/unmasked (XMCM is nonzero) and the 2-partition mode is selected (XMCME = 0). Under these conditions, the McBSP transmitter can transmit or withhold data in any of the 32 channels that are assigned to partitions A and B of the transmitter. The 128 transmit channels of the McBSP are divided equally among 8 blocks (0 through 7). When XPBBLK is applicable, use the XPBBLK bit to assign one of the odd-numbered blocks (1, 3, 5, or 7) to partition B; use the XPABLK bit to assign one of the even-numbered blocks (0, 2, 4, or 6) to partition A. If you want to use more than 32 channels, you can change block assignments dynamically. You can assign a new block to one partition while the transmitter is handling activity in the other partition. For example, while the block in partition A is active, you can change which block is assigned to partition B. The XCBLK bit is regularly updated to indicate which block is active. When XMCM = 3h (for symmetric transmission and reception), the transmitter uses the receive block bits (RPABLK and RPBBLK) rather than the transmit block bits (XPABLK and XPBBLK). • 0 = Block 1: channels 16 through 31 • 1h = Block 3: channels 48 through 63 • 2h = Block 5: channels 80 through 95 • 3h = Block 7: channels 112 through 127
22-21	XPABLK	R/W	0h	Transmit partition A block bit. XPABLK is only applicable if channels can be individually disabled/enabled and masked/unmasked (XMCM is nonzero) and the 2-partition mode is selected (XMCME = 0). Under these conditions, the McBSP transmitter can transmit or withhold data in any of the 32 channels that are assigned to partitions A and B of the transmitter. See the XPBBLK bit description for more information about assigning blocks to partitions A and B. • 0 = Block 0: channels 0 through 15 • 1h = Block 2: channels 32 through 47 • 2h = Block 4: channels 64 through 79 • 3h = Block 6: channels 96 through 111

Table 11-1650. MCBSP_MCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20-18	XCBLK	R	0h	Transmit current block indicator. XCBLK indicates which block of 16 channels is involved in the current McBSP transmission. • 0 = Block 0: channels 0 through 15 • 1h = Block 1: channels 16 through 31 • 2h = Block 2: channels 32 through 47 • 3h = Block 3: channels 48 through 63 • 4h = Block 4: channels 64 through 79 • 5h = Block 5: channels 80 through 95 • 6h = Block 6: channels 96 through 111 • 7h = Block 7: channels 112 through 127
17-16	XMCM	R/W	0h	Transmit multichannel selection mode bit. XMCM determines whether all channels or only selected channels are enabled and unmasked for transmission. • 0 = Transmit multichannel selection is off. All channels are enabled and unmasked. No channels can be disabled or masked. • 1h = All channels are disabled unless they are selected in the appropriate enhanced transmit channel enable register (XCERE *n*). If enabled, a channel in this mode is also unmasked. The XMCME bit determines whether 32 channels or 128 channels are selectable in XCERE *n*. • 2h = All channels are enabled, but they are masked unless they are selected in the appropriate enhanced transmit channel enable register (XCERE *n*). The XMCME bit determines whether 32 channels or 128 channels are selectable in XCERE *n*. • 3h = This mode is used for symmetric transmission and reception. All channels are disabled for transmission unless they are enabled for reception in the appropriate enhanced receive channel enable register (RCERE *n*). Once enabled, they are masked unless they are also selected in the appropriate enhanced transmit channel enable register (XCERE *n*). The XMCME bit determines whether 32 channels or 128 channels are selectable in RCERE *n* and XCERE *n*.
15-10	RESERVED	R	0h	Reserved.
9	RMCME	R/W	0h	Receive multichannel partition mode bit. RMCME is only applicable if channels can be individually disabled/enabled for reception (RMCM = 1). RMCME determines whether only 32 channels or all 128 channels are to be individually selectable. 0 = 2-partition mode. Only partitions A and B are used. You can control up to 32 channels in the receive multichannel selection mode (RMCM = 1). Assign 16 channels to partition A with the RPABLK bit and assign 16 channels to partition B with the RPBBLK bit. You control the channels with the enhanced receive channel enable register partition A/B (MCBSP_RCERE0). 1 = You can control up to 128 channels in the receive multichannel selection mode. You control the channels with the appropriate enhanced receive channel enable register (RCEREn): MCBSP_RCERE0: Channels 0 through 31 MCBSP_RCERE1: Channels 32 through 63 MCBSP_RCERE2: Channels 64 through 95 MCBSP_RCERE3: Channels 96 through 127

Table 11-1650. MCBSP_MCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8-7	RPBBLK	R/W	0h	Receive partition B block bit. RPBBLK is only applicable if channels can be individually disabled/enabled (RMCM = 1) and the 2-partition mode is selected (RMCME = 0). Under these conditions, the McBSP receiver can accept or ignore data in any of the 32 channels that are assigned to partitions A and B of the receiver. The 128 receive channels of the McBSP are divided equally among 8 blocks (0 through 7). When RPBBLK is applicable, use the RPBBLK bit to assign one of the odd-numbered blocks (1, 3, 5, or 7) to partition B; use the RPABLK bit to assign one of the even-numbered blocks (0, 2, 4, or 6) to partition A. If you want to use more than 32 channels, you can change block assignments dynamically. You can assign a new block to one partition while the receiver is handling activity in the other partition. For example, while the block in partition A is active, you can change which block is assigned to partition B. The RCBLK bit is regularly updated to indicate which block is active. When XMCM = 3h (for symmetric transmission and reception), the transmitter uses the receive block bits (RPABLK and RPBBLK) rather than the transmit block bits (XPABLK and XPBBLK). • 0 = Block 1: channels 16 through 31 • 1h = Block 3: channels 48 through 63 • 2h = Block 5: channels 80 through 95 • 3h = Block 7: channels 112 through 127
6-5	RPABLK	R/W	0h	Receive partition A block bit. RPABLK is only applicable if channels can be individually disabled/enabled (RMCM = 1) and the 2-partition mode is selected (RMCME = 0). Under these conditions, the McBSP receiver can accept or ignore data in any of the 32 channels that are assigned to partitions A and B of the receiver. See the RPBBLK bit description for more information about assigning blocks to partitions A and B. • 0 = Block 0: channels 0 through 15 • 1h = Block 2: channels 32 through 47 • 2h = Block 4: channels 64 through 79 • 3h = Block 6: channels 96 through 111
4-2	RCBLK	R	0h	Receive current block indicator. RCBLK indicates which block of 16 channels is involved in the current McBSP reception. • 0 = Block 0: channels 0 through 15 • 1h = Block 1: channels 16 through 31 • 2h = Block 2: channels 32 through 47 • 3h = Block 3: channels 48 through 63 • 4h = Block 4: channels 64 through 79 • 5h = Block 5: channels 80 through 95 • 6h = Block 6: channels 96 through 111 • 7h = Block 7: channels 112 through 127
1	RESERVED	R	0h	Reserved.
0	RMCM	R/W	0h	Receive multichannel selection mode bit. RMCM determines whether all channels or only selected channels are enabled for reception. • 0 = All 128 channels are enabled. • 1 = Multichannel selection mode. Channels can be individually enabled or disabled. The only channels enabled are those selected in the appropriate enhanced receive channel enable register (RCERE *n*). The way channels are assigned to RCERE *n* depends on the number of receive channel partitions (2 or 8), as defined by the RMCME bit.

Table 11-1651. Register Call Summary for MCBSP_MCR

McBSP Functional Description • Activity on McBSP Pins for Different Values of XMCM: [0] • Receive Multi-channel Selection Mode: [0][1] • Assigning Blocks to Partitions A and B: [0][1][2][3][4][5][6][7] • Transmit Multi-channel Selection Mode: [0][1][2][3][4] • Frame Length: RFRLEN1/2 and XFRLEN1/2: [0]
McBSP Registers • MCBSP_XCERE2 Register (Offset = 34h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25][26][27][28][29][30][31][32][33][34][35] • MCBSP_XCERE1 Register (Offset = 2Ch) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25][26][27][28][29][30][31][32][33][34][35] • MCBSP_XCERE0 Register (Offset = 20h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25][26][27][28][29][30][31][32][33][34][35] • MCBSP_RCERE3 Register (Offset = 38h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25][26][27][28][29][30][31][32][33][34][35][36] • MCBSP_XCERE3 Register (Offset = 3Ch) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25][26][27][28][29][30][31][32][33][34][35] • McBSP Registers: [0] • MCBSP_RCERE0 Register (Offset = 1Ch) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25][26][27][28][29][30][31][32][33][34][35][36] • MCBSP_RCERE2 Register (Offset = 30h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25][26][27][28][29][30][31][32][33][34][35][36] • MCBSP_RCERE1 Register (Offset = 28h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25][26][27][28][29][30][31][32][33][34][35][36] • MCBSP_MCR Register (Offset = 18h) [reset = 0h]: [0][1]
McBSP Programming Guide • Programming the MCBSP Registers for the Desired Receiver Configuration (Step 2): [0] • Programming the MCBSP Registers for the Desired Transmitter Operation (Step 2): [0]

11.10.6.8 MCBSP_RCERE0 Register (Offset = 1Ch) [reset = 0h]

The enhanced receive channel enable register MCBSP_RCERE0 is shown in Figure 11-764 and described in Table 11-1653. The RCERE*n* are used to enable any of 128 elements for receive. MCBSP_RCERE0 is the only register used in normal mode (up to 32 channels can be selected in partitions A and B, RMCME = XMCME = 0 in MCBSP_MCR). MCBSP_RCERE0-MCBSP_RCERE3 are used when in enhanced mode (up to 128 channels can be selected in all partitions, RMCME = XMCME = 1 in MCBSP_MCR).

The receive multichannel partition mode (RMCME) bit in the multichannel control register (MCBSP_MCR) is only applicable if channels can be individually disabled/enabled for reception (RMCM = 1). The RMCME bit determines whether only 32 channels or all 128 channels are to be individually selectable:

- **When RMCME = 0:** Only partitions A and B are used. MCBSP_RCERE0 is used to enable any of the 32 elements for a receive. Of the 32 elements, 16 channels belong to a subframe in partition A and 16 channels belong to a subframe in partition B. The RCE0-RCE15 bits enable elements within the 16-channel elements in partition A and the RCE16-RCE31 bits enable elements within the 16-channel elements in partition B. The 16 channels in partition A are assigned with the RPABLK bit in MCBSP_MCR and the 16 channels in partition B are assigned with the RPBBLK bit in MCBSP_MCR.
- **When RMCME = 1:** All partitions are used. MCBSP_RCERE0 is used to enable any of the 32 elements in channels 0 through 31 for a receive. Of the 32 elements, channels 0 to 15 belong to a subframe in partition A and channels 16 to 31 belong to a subframe in partition B. The RCE0-RCE15 bits enable elements within the 16-channel elements in partition A and the RCE16-RCE31 bits enable elements within the 16-channel elements in partition B.

Section 11.10.4.7.7.1 shows the 128 channels in a multichannel data stream and their corresponding enable bits in RCEREn.

Table 11-1652. MCBSP_RCERE0 Instances

Instance	Physical Address
MCBSP	0234 601Ch

Figure 11-764. MCBSP_RCERE0 Register

31	30	29	28	27	26	25	24
RCE31	RCE30	RCE29	RCE28	RCE27	RCE26	RCE25	RCE24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RCE23	RCE22	RCE21	RCE20	RCE19	RCE18	RCE17	RCE16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RCE15	RCE14	RCE13	RCE12	RCE11	RCE10	RCE9	RCE8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RCE7	RCE6	RCE5	RCE4	RCE3	RCE2	RCE1	RCE0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1653. MCBSP_RCERE0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RCE31	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE31. • 1 = Enable the channel that is mapped to RCERE31.

Table 11-1653. MCBSP_RCERE0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
30	RCE30	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE30. • 1 = Enable the channel that is mapped to RCERE30.
29	RCE29	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE29. • 1 = Enable the channel that is mapped to RCERE29.
28	RCE28	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE28. • 1 = Enable the channel that is mapped to RCERE28.
27	RCE27	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE27. • 1 = Enable the channel that is mapped to RCERE27.
26	RCE26	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE26. • 1 = Enable the channel that is mapped to RCERE26.
25	RCE25	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE25. • 1 = Enable the channel that is mapped to RCERE25.
24	RCE24	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE24. • 1 = Enable the channel that is mapped to RCERE24.
23	RCE23	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE23. • 1 = Enable the channel that is mapped to RCERE23.
22	RCE22	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE22. • 1 = Enable the channel that is mapped to RCERE22.
21	RCE21	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE21. • 1 = Enable the channel that is mapped to RCERE21.
20	RCE20	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE20. • 1 = Enable the channel that is mapped to RCERE20.
19	RCE19	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE19. • 1 = Enable the channel that is mapped to RCERE19.
18	RCE18	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE18. • 1 = Enable the channel that is mapped to RCERE18.

Table 11-1653. MCBSP_RCERE0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	RCE17	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE17. • 1 = Enable the channel that is mapped to RCERE17.
16	RCE16	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE16. • 1 = Enable the channel that is mapped to RCERE16.
15	RCE15	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE15. • 1 = Enable the channel that is mapped to RCERE15.
14	RCE14	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE14. • 1 = Enable the channel that is mapped to RCERE14.
13	RCE13	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE13. • 1 = Enable the channel that is mapped to RCERE13.
12	RCE12	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE12. • 1 = Enable the channel that is mapped to RCERE12.
11	RCE11	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE11. • 1 = Enable the channel that is mapped to RCERE11.
10	RCE10	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE10. • 1 = Enable the channel that is mapped to RCERE10.
9	RCE9	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE9. • 1 = Enable the channel that is mapped to RCERE9.
8	RCE8	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE8. • 1 = Enable the channel that is mapped to RCERE8.
7	RCE7	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE7. • 1 = Enable the channel that is mapped to RCERE7.
6	RCE6	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE6. • 1 = Enable the channel that is mapped to RCERE6.
5	RCE5	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE5. • 1 = Enable the channel that is mapped to RCERE5.

Table 11-1653. MCBSP_RCERE0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	RCE4	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE4. • 1 = Enable the channel that is mapped to RCERE4.
3	RCE3	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE3. • 1 = Enable the channel that is mapped to RCERE3.
2	RCE2	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE2. • 1 = Enable the channel that is mapped to RCERE2.
1	RCE1	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE1. • 1 = Enable the channel that is mapped to RCERE1.
0	RCE0	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE0. • 1 = Enable the channel that is mapped to RCERE0.

Table 11-1654. Register Call Summary for MCBSP_RCERE0

McBSP Functional Description • Using Eight Partitions: [0][1] • Assigning Blocks to Partitions A and B: [0][1][2]
McBSP Registers • MCBSP_RCERE3 Register (Offset = 38h) [reset = 0h]: [0][1][2][3] • McBSP Registers: [0] • MCBSP_RCERE0 Register (Offset = 1Ch) [reset = 0h]: [0][1][2][3][4] • MCBSP_RCERE2 Register (Offset = 30h) [reset = 0h]: [0][1][2][3] • MCBSP_RCERE1 Register (Offset = 28h) [reset = 0h]: [0][1][2][3] • MCBSP_MCR Register (Offset = 18h) [reset = 0h]: [0][1]

11.10.6.9 MCBSP_RCERE1 Register (Offset = 28h) [reset = 0h]

The enhanced receive channel enable register MCBSP_RCERE1 is shown in Figure 11-765 and described in Table 11-1656. The RCERE*n* are used to enable any of 128 elements for receive. MCBSP_RCERE0 is the only register used in normal mode (up to 32 channels can be selected in partitions A and B, RMCME = XMCME = 0 in MCBSP_MCR). MCBSP_RCERE0-MCBSP_RCERE3 are used when in enhanced mode (up to 128 channels can be selected in all partitions, RMCME = XMCME = 1 in MCBSP_MCR).

The receive multichannel partition mode (RMCME) bit in the multichannel control register (MCBSP_MCR) is only applicable if channels can be individually disabled/enabled for reception (RMCM = 1). The RMCME bit determines whether only 32 channels or all 128 channels are to be individually selectable:

- **When RMCME = 0:** Only partitions A and B are used. MCBSP_RCERE0 is used to enable any of the 32 elements for a receive. Of the 32 elements, 16 channels belong to a subframe in partition A and 16 channels belong to a subframe in partition B. The RCE0-RCE15 bits enable elements within the 16-channel elements in partition A and the RCE16-RCE31 bits enable elements within the 16-channel elements in partition B. The 16 channels in partition A are assigned with the RPABLK bit in MCBSP_MCR and the 16 channels in partition B are assigned with the RPBBLK bit in MCBSP_MCR.
- **When RMCME = 1:** All partitions are used. MCBSP_RCERE0 is used to enable any of the 32 elements in channels 0 through 31 for a receive. Of the 32 elements, channels 0 to 15 belong to a subframe in partition A and channels 16 to 31 belong to a subframe in partition B. The RCE0-RCE15 bits enable elements within the 16-channel elements in partition A and the RCE16-RCE31 bits enable elements within the 16-channel elements in partition B.

Section 11.10.4.7.7.1 shows the 128 channels in a multichannel data stream and their corresponding enable bits in RCEREn.

Table 11-1655. MCBSP_RCERE1 Instances

Instance	Physical Address
MCBSP	0234 6028h

Figure 11-765. MCBSP_RCERE1 Register

31	30	29	28	27	26	25	24
RCE31	RCE30	RCE29	RCE28	RCE27	RCE26	RCE25	RCE24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RCE23	RCE22	RCE21	RCE20	RCE19	RCE18	RCE17	RCE16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RCE15	RCE14	RCE13	RCE12	RCE11	RCE10	RCE9	RCE8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RCE7	RCE6	RCE5	RCE4	RCE3	RCE2	RCE1	RCE0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1656. MCBSP_RCERE1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RCE31	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE31. • 1 = Enable the channel that is mapped to RCERE31.

Table 11-1656. MCBSP_RCERE1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
30	RCE30	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE30. • 1 = Enable the channel that is mapped to RCERE30.
29	RCE29	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE29. • 1 = Enable the channel that is mapped to RCERE29.
28	RCE28	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE28. • 1 = Enable the channel that is mapped to RCERE28.
27	RCE27	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE27. • 1 = Enable the channel that is mapped to RCERE27.
26	RCE26	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE26. • 1 = Enable the channel that is mapped to RCERE26.
25	RCE25	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE25. • 1 = Enable the channel that is mapped to RCERE25.
24	RCE24	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE24. • 1 = Enable the channel that is mapped to RCERE24.
23	RCE23	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE23. • 1 = Enable the channel that is mapped to RCERE23.
22	RCE22	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE22. • 1 = Enable the channel that is mapped to RCERE22.
21	RCE21	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE21. • 1 = Enable the channel that is mapped to RCERE21.
20	RCE20	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE20. • 1 = Enable the channel that is mapped to RCERE20.
19	RCE19	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE19. • 1 = Enable the channel that is mapped to RCERE19.
18	RCE18	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE18. • 1 = Enable the channel that is mapped to RCERE18.

Table 11-1656. MCBSP_RCERE1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	RCE17	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE17. • 1 = Enable the channel that is mapped to RCERE17.
16	RCE16	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE16. • 1 = Enable the channel that is mapped to RCERE16.
15	RCE15	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE15. • 1 = Enable the channel that is mapped to RCERE15.
14	RCE14	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE14. • 1 = Enable the channel that is mapped to RCERE14.
13	RCE13	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE13. • 1 = Enable the channel that is mapped to RCERE13.
12	RCE12	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE12. • 1 = Enable the channel that is mapped to RCERE12.
11	RCE11	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE11. • 1 = Enable the channel that is mapped to RCERE11.
10	RCE10	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE10. • 1 = Enable the channel that is mapped to RCERE10.
9	RCE9	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE9. • 1 = Enable the channel that is mapped to RCERE9.
8	RCE8	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE8. • 1 = Enable the channel that is mapped to RCERE8.
7	RCE7	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE7. • 1 = Enable the channel that is mapped to RCERE7.
6	RCE6	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE6. • 1 = Enable the channel that is mapped to RCERE6.
5	RCE5	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE5. • 1 = Enable the channel that is mapped to RCERE5.

Table 11-1656. MCBSP_RCERE1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	RCE4	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE4. • 1 = Enable the channel that is mapped to RCERE4.
3	RCE3	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE3. • 1 = Enable the channel that is mapped to RCERE3.
2	RCE2	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE2. • 1 = Enable the channel that is mapped to RCERE2.
1	RCE1	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE1. • 1 = Enable the channel that is mapped to RCERE1.
0	RCE0	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE0. • 1 = Enable the channel that is mapped to RCERE0.

Table 11-1657. Register Call Summary for MCBSP_RCERE1

McBSP Functional Description • Using Eight Partitions: [0][1]
McBSP Registers • MCBSP_MCR Register (Offset = 18h) [reset = 0h]: [0] • McBSP Registers: [0] • MCBSP_RCERE1 Register (Offset = 28h) [reset = 0h]: [0]

11.10.6.10 MCBSP_RCERE2 Register (Offset = 30h) [reset = 0h]

The enhanced receive channel enable register MCBSP_RCERE2 is shown in Figure 11-766 and described in Table 11-1659. The RCERE*n* are used to enable any of 128 elements for receive. MCBSP_RCERE0 is the only register used in normal mode (up to 32 channels can be selected in partitions A and B, RMCME = XMCME = 0 in MCBSP_MCR). MCBSP_RCERE0-MCBSP_RCERE3 are used when in enhanced mode (up to 128 channels can be selected in all partitions, RMCME = XMCME = 1 in MCBSP_MCR).

The receive multichannel partition mode (RMCME) bit in the multichannel control register (MCBSP_MCR) is only applicable if channels can be individually disabled/enabled for reception (RMCM = 1). The RMCME bit determines whether only 32 channels or all 128 channels are to be individually selectable:

- **When RMCME = 0:** Only partitions A and B are used. MCBSP_RCERE0 is used to enable any of the 32 elements for a receive. Of the 32 elements, 16 channels belong to a subframe in partition A and 16 channels belong to a subframe in partition B. The RCE0-RCE15 bits enable elements within the 16-channel elements in partition A and the RCE16-RCE31 bits enable elements within the 16-channel elements in partition B. The 16 channels in partition A are assigned with the RPABLK bit in MCBSP_MCR and the 16 channels in partition B are assigned with the RPBBLK bit in MCBSP_MCR.
- **When RMCME = 1:** All partitions are used. MCBSP_RCERE0 is used to enable any of the 32 elements in channels 0 through 31 for a receive. Of the 32 elements, channels 0 to 15 belong to a subframe in partition A and channels 16 to 31 belong to a subframe in partition B. The RCE0-RCE15 bits enable elements within the 16-channel elements in partition A and the RCE16-RCE31 bits enable elements within the 16-channel elements in partition B.

Section 11.10.4.7.7.1 shows the 128 channels in a multichannel data stream and their corresponding enable bits in RCEREn.

Table 11-1658. MCBSP_RCERE2 Instances

Instance	Physical Address
MCBSP	0234 6030h

Figure 11-766. MCBSP_RCERE2 Register

31	30	29	28	27	26	25	24
RCE31	RCE30	RCE29	RCE28	RCE27	RCE26	RCE25	RCE24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RCE23	RCE22	RCE21	RCE20	RCE19	RCE18	RCE17	RCE16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RCE15	RCE14	RCE13	RCE12	RCE11	RCE10	RCE9	RCE8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RCE7	RCE6	RCE5	RCE4	RCE3	RCE2	RCE1	RCE0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1659. MCBSP_RCERE2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RCE31	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE31. • 1 = Enable the channel that is mapped to RCERE31.

Table 11-1659. MCBSP_RCERE2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
30	RCE30	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE30. • 1 = Enable the channel that is mapped to RCERE30.
29	RCE29	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE29. • 1 = Enable the channel that is mapped to RCERE29.
28	RCE28	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE28. • 1 = Enable the channel that is mapped to RCERE28.
27	RCE27	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE27. • 1 = Enable the channel that is mapped to RCERE27.
26	RCE26	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE26. • 1 = Enable the channel that is mapped to RCERE26.
25	RCE25	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE25. • 1 = Enable the channel that is mapped to RCERE25.
24	RCE24	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE24. • 1 = Enable the channel that is mapped to RCERE24.
23	RCE23	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE23. • 1 = Enable the channel that is mapped to RCERE23.
22	RCE22	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE22. • 1 = Enable the channel that is mapped to RCERE22.
21	RCE21	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE21. • 1 = Enable the channel that is mapped to RCERE21.
20	RCE20	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE20. • 1 = Enable the channel that is mapped to RCERE20.
19	RCE19	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE19. • 1 = Enable the channel that is mapped to RCERE19.
18	RCE18	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE18. • 1 = Enable the channel that is mapped to RCERE18.

Table 11-1659. MCBSP_RCERE2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	RCE17	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE17. • 1 = Enable the channel that is mapped to RCERE17.
16	RCE16	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE16. • 1 = Enable the channel that is mapped to RCERE16.
15	RCE15	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE15. • 1 = Enable the channel that is mapped to RCERE15.
14	RCE14	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE14. • 1 = Enable the channel that is mapped to RCERE14.
13	RCE13	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE13. • 1 = Enable the channel that is mapped to RCERE13.
12	RCE12	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE12. • 1 = Enable the channel that is mapped to RCERE12.
11	RCE11	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE11. • 1 = Enable the channel that is mapped to RCERE11.
10	RCE10	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE10. • 1 = Enable the channel that is mapped to RCERE10.
9	RCE9	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE9. • 1 = Enable the channel that is mapped to RCERE9.
8	RCE8	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE8. • 1 = Enable the channel that is mapped to RCERE8.
7	RCE7	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE7. • 1 = Enable the channel that is mapped to RCERE7.
6	RCE6	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE6. • 1 = Enable the channel that is mapped to RCERE6.
5	RCE5	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE5. • 1 = Enable the channel that is mapped to RCERE5.

Table 11-1659. MCBSP_RCERE2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	RCE4	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE4. • 1 = Enable the channel that is mapped to RCERE4.
3	RCE3	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE3. • 1 = Enable the channel that is mapped to RCERE3.
2	RCE2	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE2. • 1 = Enable the channel that is mapped to RCERE2.
1	RCE1	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE1. • 1 = Enable the channel that is mapped to RCERE1.
0	RCE0	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE0. • 1 = Enable the channel that is mapped to RCERE0.

Table 11-1660. Register Call Summary for MCBSP_RCERE2

McBSP Functional Description • Using Eight Partitions: [0][1]
McBSP Registers • MCBSP_RCERE2 Register (Offset = 30h) [reset = 0h]: [0] • MCBSP_MCR Register (Offset = 18h) [reset = 0h]: [0] • McBSP Registers: [0]

11.10.6.11 MCBSP_RCERE3 Register (Offset = 38h) [reset = 0h]

The enhanced receive channel enable register MCBSP_RCERE3 is shown in Figure 11-767 and described in Table 11-1662. The RCERE*n* are used to enable any of 128 elements for receive. MCBSP_RCERE0 is the only register used in normal mode (up to 32 channels can be selected in partitions A and B, RMCME = XMCME = 0 in MCBSP_MCR). MCBSP_RCERE0-MCBSP_RCERE3 are used when in enhanced mode (up to 128 channels can be selected in all partitions, RMCME = XMCME = 1 in MCBSP_MCR).

The receive multichannel partition mode (RMCME) bit in the multichannel control register (MCBSP_MCR) is only applicable if channels can be individually disabled/enabled for reception (RMCM = 1). The RMCME bit determines whether only 32 channels or all 128 channels are to be individually selectable:

- **When RMCME = 0:** Only partitions A and B are used. MCBSP_RCERE0 is used to enable any of the 32 elements for a receive. Of the 32 elements, 16 channels belong to a subframe in partition A and 16 channels belong to a subframe in partition B. The RCE0-RCE15 bits enable elements within the 16-channel elements in partition A and the RCE16-RCE31 bits enable elements within the 16-channel elements in partition B. The 16 channels in partition A are assigned with the RPABLK bit in MCBSP_MCR and the 16 channels in partition B are assigned with the RPBBLK bit in MCBSP_MCR.
- **When RMCME = 1:** All partitions are used. MCBSP_RCERE0 is used to enable any of the 32 elements in channels 0 through 31 for a receive. Of the 32 elements, channels 0 to 15 belong to a subframe in partition A and channels 16 to 31 belong to a subframe in partition B. The RCE0-RCE15 bits enable elements within the 16-channel elements in partition A and the RCE16-RCE31 bits enable elements within the 16-channel elements in partition B.

Section 11.10.4.7.7.1 shows the 128 channels in a multichannel data stream and their corresponding enable bits in RCEREn.

Table 11-1661. MCBSP_RCERE3 Instances

Instance	Physical Address
MCBSP	0234 6038h

Figure 11-767. MCBSP_RCERE3 Register

31	30	29	28	27	26	25	24
RCE31	RCE30	RCE29	RCE28	RCE27	RCE26	RCE25	RCE24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RCE23	RCE22	RCE21	RCE20	RCE19	RCE18	RCE17	RCE16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RCE15	RCE14	RCE13	RCE12	RCE11	RCE10	RCE9	RCE8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RCE7	RCE6	RCE5	RCE4	RCE3	RCE2	RCE1	RCE0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1662. MCBSP_RCERE3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RCE31	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE31. • 1 = Enable the channel that is mapped to RCERE31.

Table 11-1662. MCBSP_RCERE3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
30	RCE30	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE30. • 1 = Enable the channel that is mapped to RCERE30.
29	RCE29	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE29. • 1 = Enable the channel that is mapped to RCERE29.
28	RCE28	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE28. • 1 = Enable the channel that is mapped to RCERE28.
27	RCE27	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE27. • 1 = Enable the channel that is mapped to RCERE27.
26	RCE26	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE26. • 1 = Enable the channel that is mapped to RCERE26.
25	RCE25	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE25. • 1 = Enable the channel that is mapped to RCERE25.
24	RCE24	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE24. • 1 = Enable the channel that is mapped to RCERE24.
23	RCE23	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE23. • 1 = Enable the channel that is mapped to RCERE23.
22	RCE22	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE22. • 1 = Enable the channel that is mapped to RCERE22.
21	RCE21	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE21. • 1 = Enable the channel that is mapped to RCERE21.
20	RCE20	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE20. • 1 = Enable the channel that is mapped to RCERE20.
19	RCE19	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE19. • 1 = Enable the channel that is mapped to RCERE19.
18	RCE18	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE18. • 1 = Enable the channel that is mapped to RCERE18.

Table 11-1662. MCBSP_RCERE3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	RCE17	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE17. • 1 = Enable the channel that is mapped to RCERE17.
16	RCE16	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE16. • 1 = Enable the channel that is mapped to RCERE16.
15	RCE15	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE15. • 1 = Enable the channel that is mapped to RCERE15.
14	RCE14	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE14. • 1 = Enable the channel that is mapped to RCERE14.
13	RCE13	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE13. • 1 = Enable the channel that is mapped to RCERE13.
12	RCE12	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE12. • 1 = Enable the channel that is mapped to RCERE12.
11	RCE11	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE11. • 1 = Enable the channel that is mapped to RCERE11.
10	RCE10	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE10. • 1 = Enable the channel that is mapped to RCERE10.
9	RCE9	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE9. • 1 = Enable the channel that is mapped to RCERE9.
8	RCE8	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE8. • 1 = Enable the channel that is mapped to RCERE8.
7	RCE7	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE7. • 1 = Enable the channel that is mapped to RCERE7.
6	RCE6	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE6. • 1 = Enable the channel that is mapped to RCERE6.
5	RCE5	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE5. • 1 = Enable the channel that is mapped to RCERE5.

Table 11-1662. MCBSP_RCERE3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	RCE4	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE4. • 1 = Enable the channel that is mapped to RCERE4.
3	RCE3	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE3. • 1 = Enable the channel that is mapped to RCERE3.
2	RCE2	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE2. • 1 = Enable the channel that is mapped to RCERE2.
1	RCE1	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE1. • 1 = Enable the channel that is mapped to RCERE1.
0	RCE0	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCBSP_MCR). • 0 = Disable the channel that is mapped to RCERE0. • 1 = Enable the channel that is mapped to RCERE0.

Table 11-1663. Register Call Summary for MCBSP_RCERE3

McBSP Functional Description • Using Eight Partitions: [0][1]
McBSP Registers • MCBSP_RCERE3 Register (Offset = 38h) [reset = 0h]: [0][1] • McBSP Registers: [0] • MCBSP_RCERE0 Register (Offset = 1Ch) [reset = 0h]: [0] • MCBSP_RCERE2 Register (Offset = 30h) [reset = 0h]: [0] • MCBSP_RCERE1 Register (Offset = 28h) [reset = 0h]: [0] • MCBSP_MCR Register (Offset = 18h) [reset = 0h]: [0]

11.10.6.12 MCBSP_XCERE0 Register (Offset = 20h) [reset = 0h]

The enhanced transmit channel enable register MCBSP_XCERE0 is shown in Figure 11-768 and described in Table 11-1665. The XCERE*n* are used to enable any of 128 elements for transmit. MCBSP_XCERE0 is the only register used in normal mode (up to 32 channels can be selected in partitions A and B, RMCME = XMCME = 0 in MCBSP_MCR). MCBSP_XCERE0-MCBSP_XCERE3 are used when in enhanced mode (up to 128 channels can be selected in all partitions, RMCME = XMCME = 1 in MCBSP_MCR).

The transmit multichannel partition mode (XMCME) bit in the multichannel control register (MCBSP_MCR) is only applicable if channels can be individually disabled/enabled or masked/unmasked for transmission (XMCM is nonzero). The XMCME bit determines whether only 32 channels or all 128 channels are to be individually selectable:

- **When XMCME = 0:** Only partitions A and B are used. MCBSP_XCERE0 is used to enable any of the 32 elements for a transmit. Of the 32 elements, 16 channels belong to a subframe in partition A and 16 channels belong to a subframe in partition B. The XCE0-XCE15 bits enable elements within the 16-channel elements in partition A and the XCE16-XCE31 bits enable elements within the 16-channel elements in partition B. You can control up to 32 channels in the transmit multichannel selection mode selected with the XMCM bit in MCBSP_MCR.
- **When XMCME = 1:** All partitions are used. MCBSP_XCERE0 is used to enable any of the 32 elements in channels 0 through 31 for a transmit. Of the 32 elements, channels 0 to 15 belong to a subframe in partition A and channels 16 to 31 belong to a subframe in partition B. The XCE0-XCE15 bits enable elements within the 16-channel elements in partition A and the XCE16-XCE31 bits enable elements within the 16-channel elements in partition B.

Section 11.10.4.7.8.3 shows the 128 channels in a multichannel data stream and their corresponding enable bits in XCEREn.

Table 11-1664. MCBSP_XCERE0 Instances

Instance	Physical Address
MCBSP	0234 6020h

Figure 11-768. MCBSP_XCERE0 Register

31	30	29	28	27	26	25	24
XCE31	XCE30	XCE29	XCE28	XCE27	XCE26	XCE25	XCE24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
XCE23	XCE22	XCE21	XCE20	XCE19	XCE18	XCE17	XCE16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
XCE15	XCE14	XCE13	XCE12	XCE11	XCE10	XCE9	XCE8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
XCE7	XCE6	XCE5	XCE4	XCE3	XCE2	XCE1	XCE0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1665. MCBSP_XCERE0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	XCE31	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE31. – 1 = Enable and unmask the channel that is mapped to XCE31. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE31. – 1 = Unmask the channel that is mapped to XCE31. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE31. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE31. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
30	XCE30	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE30. – 1 = Enable and unmask the channel that is mapped to XCE30. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE30. – 1 = Unmask the channel that is mapped to XCE30. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE30. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE30. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
29	XCE29	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE29. – 1 = Enable and unmask the channel that is mapped to XCE29. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE29. – 1 = Unmask the channel that is mapped to XCE29. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE29. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE29. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1665. MCBSP_XCERE0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
28	XCE28	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE28. – 1 = Enable and unmask the channel that is mapped to XCE28. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE28. – 1 = Unmask the channel that is mapped to XCE28. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE28. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE28. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
27	XCE27	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE27. – 1 = Enable and unmask the channel that is mapped to XCE27. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE27. – 1 = Unmask the channel that is mapped to XCE27. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE27. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE27. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
26	XCE26	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE26. – 1 = Enable and unmask the channel that is mapped to XCE26. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE26. – 1 = Unmask the channel that is mapped to XCE26. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE26. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE26. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1665. MCBSP_XCERE0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	XCE25	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE25. – 1 = Enable and unmask the channel that is mapped to XCE25. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE25. – 1 = Unmask the channel that is mapped to XCE25. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE25. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE25. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
24	XCE24	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE24. – 1 = Enable and unmask the channel that is mapped to XCE24. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE24. – 1 = Unmask the channel that is mapped to XCE24. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE24. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE24. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
23	XCE23	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE23. – 1 = Enable and unmask the channel that is mapped to XCE23. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE23. – 1 = Unmask the channel that is mapped to XCE23. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE23. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE23. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1665. MCBSP_XCERE0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	XCE22	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE22. – 1 = Enable and unmask the channel that is mapped to XCE22. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE22. – 1 = Unmask the channel that is mapped to XCE22. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE22. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE22. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
21	XCE21	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE21. – 1 = Enable and unmask the channel that is mapped to XCE21. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE21. – 1 = Unmask the channel that is mapped to XCE21. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE21. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE21. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
20	XCE20	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE20. – 1 = Enable and unmask the channel that is mapped to XCE20. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE20. – 1 = Unmask the channel that is mapped to XCE20. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE20. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE20. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1665. MCBSP_XCERE0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	XCE19	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE19. – 1 = Enable and unmask the channel that is mapped to XCE19. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE19. – 1 = Unmask the channel that is mapped to XCE19. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE19. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE19. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
18	XCE18	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE18. – 1 = Enable and unmask the channel that is mapped to XCE18. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE18. – 1 = Unmask the channel that is mapped to XCE18. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE18. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE18. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
17	XCE17	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE17. – 1 = Enable and unmask the channel that is mapped to XCE17. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE17. – 1 = Unmask the channel that is mapped to XCE17. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE17. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE17. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1665. MCBSP_XCERE0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	XCE16	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE16. – 1 = Enable and unmask the channel that is mapped to XCE16. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE16. – 1 = Unmask the channel that is mapped to XCE16. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE16. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE16. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
15	XCE15	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE15. – 1 = Enable and unmask the channel that is mapped to XCE15. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE15. – 1 = Unmask the channel that is mapped to XCE15. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE15. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE15. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
14	XCE14	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE14. – 1 = Enable and unmask the channel that is mapped to XCE14. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE14. – 1 = Unmask the channel that is mapped to XCE14. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE14. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE14. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1665. MCBSP_XCERE0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	XCE13	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE13. – 1 = Enable and unmask the channel that is mapped to XCE13. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE13. – 1 = Unmask the channel that is mapped to XCE13. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE13. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE13. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
12	XCE12	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE12. – 1 = Enable and unmask the channel that is mapped to XCE12. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE12. – 1 = Unmask the channel that is mapped to XCE12. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE12. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE12. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
11	XCE11	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE11. – 1 = Enable and unmask the channel that is mapped to XCE11. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE11. – 1 = Unmask the channel that is mapped to XCE11. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE11. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE11. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1665. MCBSP_XCERE0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	XCE10	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE10. – 1 = Enable and unmask the channel that is mapped to XCE10. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE10. – 1 = Unmask the channel that is mapped to XCE10. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE10. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE10. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
9	XCE9	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE9. – 1 = Enable and unmask the channel that is mapped to XCE9. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE9. – 1 = Unmask the channel that is mapped to XCE9. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE9. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE9. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
8	XCE8	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE8. – 1 = Enable and unmask the channel that is mapped to XCE8. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE8. – 1 = Unmask the channel that is mapped to XCE8. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE8. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE8. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1665. MCBSP_XCERE0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	XCE7	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE7. – 1 = Enable and unmask the channel that is mapped to XCE7. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE7. – 1 = Unmask the channel that is mapped to XCE7. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE7. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE7. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
6	XCE6	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE6. – 1 = Enable and unmask the channel that is mapped to XCE6. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE6. – 1 = Unmask the channel that is mapped to XCE6. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE6. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE6. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
5	XCE5	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE5. – 1 = Enable and unmask the channel that is mapped to XCE5. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE5. – 1 = Unmask the channel that is mapped to XCE5. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE5. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE5. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1665. MCBSP_XCERE0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	XCE4	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE4. – 1 = Enable and unmask the channel that is mapped to XCE4. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE4. – 1 = Unmask the channel that is mapped to XCE4. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE4. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE4. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
3	XCE3	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE3. – 1 = Enable and unmask the channel that is mapped to XCE3. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE3. – 1 = Unmask the channel that is mapped to XCE3. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE3. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE3. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
2	XCE2	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE2. – 1 = Enable and unmask the channel that is mapped to XCE2. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE2. – 1 = Unmask the channel that is mapped to XCE2. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE2. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE2. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1665. MCBSP_XCERE0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	XCE1	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE1. – 1 = Enable and unmask the channel that is mapped to XCE1. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE1. – 1 = Unmask the channel that is mapped to XCE1. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE1. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE1. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
0	XCE0	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE0. – 1 = Enable and unmask the channel that is mapped to XCE0. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE0. – 1 = Unmask the channel that is mapped to XCE0. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE0. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE0. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1666. Register Call Summary for MCBSP_XCERE0

McBSP Functional Description • Using Eight Partitions: [0][1] • Assigning Blocks to Partitions A and B: [0][1][2]
McBSP Registers • MCBSP_XCERE0 Register (Offset = 20h) [reset = 0h]: [0][1][2][3][4] • MCBSP_XCERE1 Register (Offset = 2Ch) [reset = 0h]: [0][1][2][3] • MCBSP_XCERE2 Register (Offset = 34h) [reset = 0h]: [0][1][2][3] • MCBSP_XCERE3 Register (Offset = 3Ch) [reset = 0h]: [0][1][2][3] • McBSP Registers: [0] • MCBSP_MCR Register (Offset = 18h) [reset = 0h]: [0]

11.10.6.13 MCBSP_XCERE1 Register (Offset = 2Ch) [reset = 0h]

The enhanced transmit channel enable register MCBSP_XCERE1 is shown in Figure 11-769 and described in Table 11-1668. The XCERE*n* are used to enable any of 128 elements for transmit. MCBSP_XCERE0 is the only register used in normal mode (up to 32 channels can be selected in partitions A and B, RMCME = XMCME = 0 in MCBSP_MCR). MCBSP_XCERE0-MCBSP_XCERE3 are used when in enhanced mode (up to 128 channels can be selected in all partitions, RMCME = XMCME = 1 in MCBSP_MCR).

The transmit multichannel partition mode (XMCME) bit in the multichannel control register (MCBSP_MCR) is only applicable if channels can be individually disabled/enabled or masked/unmasked for transmission (XMCM is nonzero). The XMCME bit determines whether only 32 channels or all 128 channels are to be individually selectable:

- **When XMCME = 0:** Only partitions A and B are used. MCBSP_XCERE0 is used to enable any of the 32 elements for a transmit. Of the 32 elements, 16 channels belong to a subframe in partition A and 16 channels belong to a subframe in partition B. The XCE0-XCE15 bits enable elements within the 16-channel elements in partition A and the XCE16-XCE31 bits enable elements within the 16-channel elements in partition B. You can control up to 32 channels in the transmit multichannel selection mode selected with the XMCM bit in MCBSP_MCR.
- **When XMCME = 1:** All partitions are used. MCBSP_XCERE0 is used to enable any of the 32 elements in channels 0 through 31 for a transmit. Of the 32 elements, channels 0 to 15 belong to a subframe in partition A and channels 16 to 31 belong to a subframe in partition B. The XCE0-XCE15 bits enable elements within the 16-channel elements in partition A and the XCE16-XCE31 bits enable elements within the 16-channel elements in partition B.

Section 11.10.4.7.8.3 shows the 128 channels in a multichannel data stream and their corresponding enable bits in XCEREn.

Table 11-1667. MCBSP_XCERE1 Instances

Instance	Physical Address
MCBSP	0234 602Ch

Figure 11-769. MCBSP_XCERE1 Register

31	30	29	28	27	26	25	24
XCE31	XCE30	XCE29	XCE28	XCE27	XCE26	XCE25	XCE24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
XCE23	XCE22	XCE21	XCE20	XCE19	XCE18	XCE17	XCE16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
XCE15	XCE14	XCE13	XCE12	XCE11	XCE10	XCE9	XCE8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
XCE7	XCE6	XCE5	XCE4	XCE3	XCE2	XCE1	XCE0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1668. MCBSP_XCERE1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	XCE31	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE31. – 1 = Enable and unmask the channel that is mapped to XCE31. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE31. – 1 = Unmask the channel that is mapped to XCE31. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE31. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE31. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
30	XCE30	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE30. – 1 = Enable and unmask the channel that is mapped to XCE30. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE30. – 1 = Unmask the channel that is mapped to XCE30. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE30. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE30. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
29	XCE29	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE29. – 1 = Enable and unmask the channel that is mapped to XCE29. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE29. – 1 = Unmask the channel that is mapped to XCE29. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE29. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE29. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1668. MCBSP_XCERE1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
28	XCE28	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE28. – 1 = Enable and unmask the channel that is mapped to XCE28. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE28. – 1 = Unmask the channel that is mapped to XCE28. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE28. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE28. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
27	XCE27	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE27. – 1 = Enable and unmask the channel that is mapped to XCE27. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE27. – 1 = Unmask the channel that is mapped to XCE27. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE27. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE27. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
26	XCE26	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE26. – 1 = Enable and unmask the channel that is mapped to XCE26. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE26. – 1 = Unmask the channel that is mapped to XCE26. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE26. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE26. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1668. MCBSP_XCERE1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	XCE25	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE25. – 1 = Enable and unmask the channel that is mapped to XCE25. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE25. – 1 = Unmask the channel that is mapped to XCE25. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE25. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE25. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
24	XCE24	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE24. – 1 = Enable and unmask the channel that is mapped to XCE24. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE24. – 1 = Unmask the channel that is mapped to XCE24. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE24. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE24. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
23	XCE23	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE23. – 1 = Enable and unmask the channel that is mapped to XCE23. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE23. – 1 = Unmask the channel that is mapped to XCE23. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE23. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE23. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1668. MCBSP_XCERE1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	XCE22	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE22. – 1 = Enable and unmask the channel that is mapped to XCE22. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE22. – 1 = Unmask the channel that is mapped to XCE22. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE22. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE22. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
21	XCE21	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE21. – 1 = Enable and unmask the channel that is mapped to XCE21. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE21. – 1 = Unmask the channel that is mapped to XCE21. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE21. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE21. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
20	XCE20	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE20. – 1 = Enable and unmask the channel that is mapped to XCE20. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE20. – 1 = Unmask the channel that is mapped to XCE20. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE20. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE20. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1668. MCBSP_XCERE1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	XCE19	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE19. – 1 = Enable and unmask the channel that is mapped to XCE19. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE19. – 1 = Unmask the channel that is mapped to XCE19. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE19. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE19. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
18	XCE18	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE18. – 1 = Enable and unmask the channel that is mapped to XCE18. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE18. – 1 = Unmask the channel that is mapped to XCE18. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE18. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE18. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
17	XCE17	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE17. – 1 = Enable and unmask the channel that is mapped to XCE17. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE17. – 1 = Unmask the channel that is mapped to XCE17. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE17. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE17. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1668. MCBSP_XCERE1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	XCE16	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE16. – 1 = Enable and unmask the channel that is mapped to XCE16. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE16. – 1 = Unmask the channel that is mapped to XCE16. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE16. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE16. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
15	XCE15	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE15. – 1 = Enable and unmask the channel that is mapped to XCE15. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE15. – 1 = Unmask the channel that is mapped to XCE15. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE15. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE15. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
14	XCE14	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE14. – 1 = Enable and unmask the channel that is mapped to XCE14. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE14. – 1 = Unmask the channel that is mapped to XCE14. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE14. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE14. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1668. MCBSP_XCERE1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	XCE13	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE13. – 1 = Enable and unmask the channel that is mapped to XCE13. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE13. – 1 = Unmask the channel that is mapped to XCE13. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE13. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE13. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
12	XCE12	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE12. – 1 = Enable and unmask the channel that is mapped to XCE12. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE12. – 1 = Unmask the channel that is mapped to XCE12. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE12. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE12. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
11	XCE11	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE11. – 1 = Enable and unmask the channel that is mapped to XCE11. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE11. – 1 = Unmask the channel that is mapped to XCE11. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE11. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE11. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1668. MCBSP_XCERE1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	XCE10	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE10. – 1 = Enable and unmask the channel that is mapped to XCE10. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE10. – 1 = Unmask the channel that is mapped to XCE10. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE10. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE10. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
9	XCE9	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE9. – 1 = Enable and unmask the channel that is mapped to XCE9. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE9. – 1 = Unmask the channel that is mapped to XCE9. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE9. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE9. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
8	XCE8	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE8. – 1 = Enable and unmask the channel that is mapped to XCE8. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE8. – 1 = Unmask the channel that is mapped to XCE8. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE8. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE8. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1668. MCBSP_XCERE1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	XCE7	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE7. – 1 = Enable and unmask the channel that is mapped to XCE7. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE7. – 1 = Unmask the channel that is mapped to XCE7. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE7. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE7. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
6	XCE6	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE6. – 1 = Enable and unmask the channel that is mapped to XCE6. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE6. – 1 = Unmask the channel that is mapped to XCE6. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE6. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE6. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
5	XCE5	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE5. – 1 = Enable and unmask the channel that is mapped to XCE5. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE5. – 1 = Unmask the channel that is mapped to XCE5. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE5. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE5. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1668. MCBSP_XCERE1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	XCE4	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE4. – 1 = Enable and unmask the channel that is mapped to XCE4. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE4. – 1 = Unmask the channel that is mapped to XCE4. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE4. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE4. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
3	XCE3	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE3. – 1 = Enable and unmask the channel that is mapped to XCE3. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE3. – 1 = Unmask the channel that is mapped to XCE3. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE3. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE3. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
2	XCE2	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE2. – 1 = Enable and unmask the channel that is mapped to XCE2. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE2. – 1 = Unmask the channel that is mapped to XCE2. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE2. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE2. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1668. MCBSP_XCERE1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	XCE1	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE1. – 1 = Enable and unmask the channel that is mapped to XCE1. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE1. – 1 = Unmask the channel that is mapped to XCE1. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE1. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE1. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
0	XCE0	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE0. – 1 = Enable and unmask the channel that is mapped to XCE0. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE0. – 1 = Unmask the channel that is mapped to XCE0. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE0. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE0. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1669. Register Call Summary for MCBSP_XCERE1

McBSP Functional Description • Using Eight Partitions: [0][1]
McBSP Registers • MCBSP_XCERE1 Register (Offset = 2Ch) [reset = 0h]: [0] • MCBSP_MCR Register (Offset = 18h) [reset = 0h]: [0] • McBSP Registers: [0]

11.10.6.14 MCBSP_XCERE2 Register (Offset = 34h) [reset = 0h]

The enhanced transmit channel enable register MCBSP_XCERE2 is shown in Figure 11-770 and described in Table 11-1671. The XCERE*n* are used to enable any of 128 elements for transmit. MCBSP_XCERE0 is the only register used in normal mode (up to 32 channels can be selected in partitions A and B, RMCME = XMCME = 0 in MCBSP_MCR). MCBSP_XCERE0-MCBSP_XCERE3 are used when in enhanced mode (up to 128 channels can be selected in all partitions, RMCME = XMCME = 1 in MCBSP_MCR).

The transmit multichannel partition mode (XMCME) bit in the multichannel control register (MCBSP_MCR) is only applicable if channels can be individually disabled/enabled or masked/unmasked for transmission (XMCM is nonzero). The XMCME bit determines whether only 32 channels or all 128 channels are to be individually selectable:

- **When XMCME = 0:** Only partitions A and B are used. MCBSP_XCERE0 is used to enable any of the 32 elements for a transmit. Of the 32 elements, 16 channels belong to a subframe in partition A and 16 channels belong to a subframe in partition B. The XCE0-XCE15 bits enable elements within the 16-channel elements in partition A and the XCE16-XCE31 bits enable elements within the 16-channel elements in partition B. You can control up to 32 channels in the transmit multichannel selection mode selected with the XMCM bit in MCBSP_MCR.
- **When XMCME = 1:** All partitions are used. MCBSP_XCERE0 is used to enable any of the 32 elements in channels 0 through 31 for a transmit. Of the 32 elements, channels 0 to 15 belong to a subframe in partition A and channels 16 to 31 belong to a subframe in partition B. The XCE0-XCE15 bits enable elements within the 16-channel elements in partition A and the XCE16-XCE31 bits enable elements within the 16-channel elements in partition B.

Section 11.10.4.7.8.3 shows the 128 channels in a multichannel data stream and their corresponding enable bits in XCEREn.

Table 11-1670. MCBSP_XCERE2 Instances

Instance	Physical Address
MCBSP	0234 6034h

Figure 11-770. MCBSP_XCERE2 Register

31	30	29	28	27	26	25	24
XCE31	XCE30	XCE29	XCE28	XCE27	XCE26	XCE25	XCE24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
XCE23	XCE22	XCE21	XCE20	XCE19	XCE18	XCE17	XCE16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
XCE15	XCE14	XCE13	XCE12	XCE11	XCE10	XCE9	XCE8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
XCE7	XCE6	XCE5	XCE4	XCE3	XCE2	XCE1	XCE0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1671. MCBSP_XCERE2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	XCE31	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE31. – 1 = Enable and unmask the channel that is mapped to XCE31. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE31. – 1 = Unmask the channel that is mapped to XCE31. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE31. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE31. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
30	XCE30	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE30. – 1 = Enable and unmask the channel that is mapped to XCE30. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE30. – 1 = Unmask the channel that is mapped to XCE30. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE30. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE30. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
29	XCE29	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE29. – 1 = Enable and unmask the channel that is mapped to XCE29. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE29. – 1 = Unmask the channel that is mapped to XCE29. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE29. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE29. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1671. MCBSP_XCERE2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
28	XCE28	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE28. – 1 = Enable and unmask the channel that is mapped to XCE28. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE28. – 1 = Unmask the channel that is mapped to XCE28. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE28. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE28. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
27	XCE27	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE27. – 1 = Enable and unmask the channel that is mapped to XCE27. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE27. – 1 = Unmask the channel that is mapped to XCE27. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE27. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE27. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
26	XCE26	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE26. – 1 = Enable and unmask the channel that is mapped to XCE26. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE26. – 1 = Unmask the channel that is mapped to XCE26. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE26. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE26. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1671. MCBSP_XCERE2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	XCE25	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE25. – 1 = Enable and unmask the channel that is mapped to XCE25. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE25. – 1 = Unmask the channel that is mapped to XCE25. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE25. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE25. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
24	XCE24	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE24. – 1 = Enable and unmask the channel that is mapped to XCE24. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE24. – 1 = Unmask the channel that is mapped to XCE24. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE24. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE24. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
23	XCE23	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE23. – 1 = Enable and unmask the channel that is mapped to XCE23. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE23. – 1 = Unmask the channel that is mapped to XCE23. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE23. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE23. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1671. MCBSP_XCERE2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	XCE22	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE22. – 1 = Enable and unmask the channel that is mapped to XCE22. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE22. – 1 = Unmask the channel that is mapped to XCE22. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE22. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE22. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
21	XCE21	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE21. – 1 = Enable and unmask the channel that is mapped to XCE21. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE21. – 1 = Unmask the channel that is mapped to XCE21. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE21. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE21. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
20	XCE20	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE20. – 1 = Enable and unmask the channel that is mapped to XCE20. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE20. – 1 = Unmask the channel that is mapped to XCE20. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE20. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE20. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1671. MCBSP_XCERE2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	XCE19	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE19. – 1 = Enable and unmask the channel that is mapped to XCE19. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE19. – 1 = Unmask the channel that is mapped to XCE19. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE19. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE19. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
18	XCE18	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE18. – 1 = Enable and unmask the channel that is mapped to XCE18. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE18. – 1 = Unmask the channel that is mapped to XCE18. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE18. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE18. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
17	XCE17	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE17. – 1 = Enable and unmask the channel that is mapped to XCE17. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE17. – 1 = Unmask the channel that is mapped to XCE17. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE17. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE17. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1671. MCBSP_XCERE2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	XCE16	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE16. – 1 = Enable and unmask the channel that is mapped to XCE16. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE16. – 1 = Unmask the channel that is mapped to XCE16. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE16. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE16. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
15	XCE15	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE15. – 1 = Enable and unmask the channel that is mapped to XCE15. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE15. – 1 = Unmask the channel that is mapped to XCE15. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE15. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE15. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
14	XCE14	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE14. – 1 = Enable and unmask the channel that is mapped to XCE14. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE14. – 1 = Unmask the channel that is mapped to XCE14. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE14. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE14. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1671. MCBSP_XCERE2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	XCE13	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE13. – 1 = Enable and unmask the channel that is mapped to XCE13. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE13. – 1 = Unmask the channel that is mapped to XCE13. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE13. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE13. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
12	XCE12	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE12. – 1 = Enable and unmask the channel that is mapped to XCE12. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE12. – 1 = Unmask the channel that is mapped to XCE12. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE12. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE12. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
11	XCE11	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE11. – 1 = Enable and unmask the channel that is mapped to XCE11. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE11. – 1 = Unmask the channel that is mapped to XCE11. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE11. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE11. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1671. MCBSP_XCERE2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	XCE10	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE10. – 1 = Enable and unmask the channel that is mapped to XCE10. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE10. – 1 = Unmask the channel that is mapped to XCE10. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE10. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE10. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
9	XCE9	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE9. – 1 = Enable and unmask the channel that is mapped to XCE9. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE9. – 1 = Unmask the channel that is mapped to XCE9. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE9. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE9. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
8	XCE8	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE8. – 1 = Enable and unmask the channel that is mapped to XCE8. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE8. – 1 = Unmask the channel that is mapped to XCE8. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE8. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE8. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1671. MCBSP_XCERE2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	XCE7	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE7. – 1 = Enable and unmask the channel that is mapped to XCE7. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE7. – 1 = Unmask the channel that is mapped to XCE7. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE7. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE7. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
6	XCE6	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE6. – 1 = Enable and unmask the channel that is mapped to XCE6. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE6. – 1 = Unmask the channel that is mapped to XCE6. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE6. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE6. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
5	XCE5	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE5. – 1 = Enable and unmask the channel that is mapped to XCE5. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE5. – 1 = Unmask the channel that is mapped to XCE5. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE5. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE5. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1671. MCBSP_XCERE2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	XCE4	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE4. – 1 = Enable and unmask the channel that is mapped to XCE4. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE4. – 1 = Unmask the channel that is mapped to XCE4. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE4. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE4. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
3	XCE3	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE3. – 1 = Enable and unmask the channel that is mapped to XCE3. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE3. – 1 = Unmask the channel that is mapped to XCE3. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE3. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE3. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
2	XCE2	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE2. – 1 = Enable and unmask the channel that is mapped to XCE2. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE2. – 1 = Unmask the channel that is mapped to XCE2. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE2. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE2. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1671. MCBSP_XCERE2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	XCE1	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE1. – 1 = Enable and unmask the channel that is mapped to XCE1. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE1. – 1 = Unmask the channel that is mapped to XCE1. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE1. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE1. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
0	XCE0	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE0. – 1 = Enable and unmask the channel that is mapped to XCE0. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE0. – 1 = Unmask the channel that is mapped to XCE0. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE0. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE0. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1672. Register Call Summary for MCBSP_XCERE2

McBSP Functional Description • Using Eight Partitions: [0][1]
McBSP Registers • MCBSP_XCERE2 Register (Offset = 34h) [reset = 0h]: [0] • MCBSP_MCR Register (Offset = 18h) [reset = 0h]: [0] • McBSP Registers: [0]

11.10.6.15 MCBSP_XCERE3 Register (Offset = 3Ch) [reset = 0h]

The enhanced transmit channel enable register MCBSP_XCERE3 is shown in Figure 11-771 and described in Table 11-1674. The XCERE*n* are used to enable any of 128 elements for transmit. MCBSP_XCERE0 is the only register used in normal mode (up to 32 channels can be selected in partitions A and B, RMCME = XMCME = 0 in MCBSP_MCR). MCBSP_XCERE0-MCBSP_XCERE3 are used when in enhanced mode (up to 128 channels can be selected in all partitions, RMCME = XMCME = 1 in MCBSP_MCR).

The transmit multichannel partition mode (XMCME) bit in the multichannel control register (MCBSP_MCR) is only applicable if channels can be individually disabled/enabled or masked/unmasked for transmission (XMCM is nonzero). The XMCME bit determines whether only 32 channels or all 128 channels are to be individually selectable:

- **When XMCME = 0:** Only partitions A and B are used. MCBSP_XCERE0 is used to enable any of the 32 elements for a transmit. Of the 32 elements, 16 channels belong to a subframe in partition A and 16 channels belong to a subframe in partition B. The XCE0-XCE15 bits enable elements within the 16-channel elements in partition A and the XCE16-XCE31 bits enable elements within the 16-channel elements in partition B. You can control up to 32 channels in the transmit multichannel selection mode selected with the XMCM bit in MCBSP_MCR.
- **When XMCME = 1:** All partitions are used. MCBSP_XCERE0 is used to enable any of the 32 elements in channels 0 through 31 for a transmit. Of the 32 elements, channels 0 to 15 belong to a subframe in partition A and channels 16 to 31 belong to a subframe in partition B. The XCE0-XCE15 bits enable elements within the 16-channel elements in partition A and the XCE16-XCE31 bits enable elements within the 16-channel elements in partition B.

Section 11.10.4.7.8.3 shows the 128 channels in a multichannel data stream and their corresponding enable bits in XCEREn.

Table 11-1673. MCBSP_XCERE3 Instances

Instance	Physical Address
MCBSP	0234 603Ch

Figure 11-771. MCBSP_XCERE3 Register

31	30	29	28	27	26	25	24
XCE31	XCE30	XCE29	XCE28	XCE27	XCE26	XCE25	XCE24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
XCE23	XCE22	XCE21	XCE20	XCE19	XCE18	XCE17	XCE16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
XCE15	XCE14	XCE13	XCE12	XCE11	XCE10	XCE9	XCE8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
XCE7	XCE6	XCE5	XCE4	XCE3	XCE2	XCE1	XCE0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1674. MCBSP_XCERE3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	XCE31	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE31. – 1 = Enable and unmask the channel that is mapped to XCE31. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE31. – 1 = Unmask the channel that is mapped to XCE31. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE31. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE31. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
30	XCE30	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE30. – 1 = Enable and unmask the channel that is mapped to XCE30. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE30. – 1 = Unmask the channel that is mapped to XCE30. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE30. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE30. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
29	XCE29	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE29. – 1 = Enable and unmask the channel that is mapped to XCE29. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE29. – 1 = Unmask the channel that is mapped to XCE29. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE29. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE29. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1674. MCBSP_XCERE3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
28	XCE28	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE28. – 1 = Enable and unmask the channel that is mapped to XCE28. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE28. – 1 = Unmask the channel that is mapped to XCE28. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE28. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE28. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
27	XCE27	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE27. – 1 = Enable and unmask the channel that is mapped to XCE27. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE27. – 1 = Unmask the channel that is mapped to XCE27. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE27. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE27. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
26	XCE26	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE26. – 1 = Enable and unmask the channel that is mapped to XCE26. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE26. – 1 = Unmask the channel that is mapped to XCE26. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE26. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE26. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1674. MCBSP_XCERE3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	XCE25	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE25. – 1 = Enable and unmask the channel that is mapped to XCE25. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE25. – 1 = Unmask the channel that is mapped to XCE25. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE25. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE25. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
24	XCE24	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE24. – 1 = Enable and unmask the channel that is mapped to XCE24. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE24. – 1 = Unmask the channel that is mapped to XCE24. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE24. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE24. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
23	XCE23	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE23. – 1 = Enable and unmask the channel that is mapped to XCE23. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE23. – 1 = Unmask the channel that is mapped to XCE23. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE23. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE23. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1674. MCBSP_XCERE3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	XCE22	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE22. – 1 = Enable and unmask the channel that is mapped to XCE22. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE22. – 1 = Unmask the channel that is mapped to XCE22. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE22. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE22. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
21	XCE21	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE21. – 1 = Enable and unmask the channel that is mapped to XCE21. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE21. – 1 = Unmask the channel that is mapped to XCE21. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE21. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE21. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
20	XCE20	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE20. – 1 = Enable and unmask the channel that is mapped to XCE20. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE20. – 1 = Unmask the channel that is mapped to XCE20. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE20. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE20. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1674. MCBSP_XCERE3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	XCE19	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE19. – 1 = Enable and unmask the channel that is mapped to XCE19. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE19. – 1 = Unmask the channel that is mapped to XCE19. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE19. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE19. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
18	XCE18	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE18. – 1 = Enable and unmask the channel that is mapped to XCE18. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE18. – 1 = Unmask the channel that is mapped to XCE18. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE18. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE18. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
17	XCE17	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE17. – 1 = Enable and unmask the channel that is mapped to XCE17. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE17. – 1 = Unmask the channel that is mapped to XCE17. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE17. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE17. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1674. MCBSP_XCERE3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	XCE16	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE16. – 1 = Enable and unmask the channel that is mapped to XCE16. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE16. – 1 = Unmask the channel that is mapped to XCE16. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE16. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE16. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
15	XCE15	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE15. – 1 = Enable and unmask the channel that is mapped to XCE15. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE15. – 1 = Unmask the channel that is mapped to XCE15. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE15. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE15. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
14	XCE14	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE14. – 1 = Enable and unmask the channel that is mapped to XCE14. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE14. – 1 = Unmask the channel that is mapped to XCE14. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE14. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE14. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1674. MCBSP_XCERE3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	XCE13	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE13. – 1 = Enable and unmask the channel that is mapped to XCE13. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE13. – 1 = Unmask the channel that is mapped to XCE13. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE13. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE13. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
12	XCE12	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE12. – 1 = Enable and unmask the channel that is mapped to XCE12. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE12. – 1 = Unmask the channel that is mapped to XCE12. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE12. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE12. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
11	XCE11	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE11. – 1 = Enable and unmask the channel that is mapped to XCE11. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE11. – 1 = Unmask the channel that is mapped to XCE11. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE11. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE11. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1674. MCBSP_XCERE3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	XCE10	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE10. – 1 = Enable and unmask the channel that is mapped to XCE10. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE10. – 1 = Unmask the channel that is mapped to XCE10. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE10. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE10. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
9	XCE9	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE9. – 1 = Enable and unmask the channel that is mapped to XCE9. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE9. – 1 = Unmask the channel that is mapped to XCE9. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE9. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE9. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
8	XCE8	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE8. – 1 = Enable and unmask the channel that is mapped to XCE8. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE8. – 1 = Unmask the channel that is mapped to XCE8. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE8. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE8. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1674. MCBSP_XCERE3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	XCE7	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE7. – 1 = Enable and unmask the channel that is mapped to XCE7. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE7. – 1 = Unmask the channel that is mapped to XCE7. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE7. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE7. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
6	XCE6	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE6. – 1 = Enable and unmask the channel that is mapped to XCE6. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE6. – 1 = Unmask the channel that is mapped to XCE6. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE6. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE6. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
5	XCE5	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE5. – 1 = Enable and unmask the channel that is mapped to XCE5. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE5. – 1 = Unmask the channel that is mapped to XCE5. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE5. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE5. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1674. MCBSP_XCERE3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	XCE4	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE4. – 1 = Enable and unmask the channel that is mapped to XCE4. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE4. – 1 = Unmask the channel that is mapped to XCE4. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE4. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE4. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
3	XCE3	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE3. – 1 = Enable and unmask the channel that is mapped to XCE3. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE3. – 1 = Unmask the channel that is mapped to XCE3. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE3. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE3. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
2	XCE2	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE2. – 1 = Enable and unmask the channel that is mapped to XCE2. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE2. – 1 = Unmask the channel that is mapped to XCE2. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE2. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE2. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1674. MCBSP_XCERE3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	XCE1	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE1. – 1 = Enable and unmask the channel that is mapped to XCE1. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE1. – 1 = Unmask the channel that is mapped to XCE1. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE1. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE1. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.
0	XCE0	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCBSP_MCR • **When XMCM = 1h (all channels disabled unless selected):** – 0 = Disable and mask the channel that is mapped to XCE0. – 1 = Enable and unmask the channel that is mapped to XCE0. • **When XMCM = 2h (all channels enabled but masked unless selected):** – 0 = Mask the channel that is mapped to XCE0. – 1 = Unmask the channel that is mapped to XCE0. • **When XMCM = 3h (all channels masked unless selected):** – 0 = Mask the channel that is mapped to XCE0. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin. – 1 = Unmask the channel that is mapped to XCE0. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

Table 11-1675. Register Call Summary for MCBSP_XCERE3

McBSP Functional Description • Using Eight Partitions: [0][1]
McBSP Registers • MCBSP_XCERE0 Register (Offset = 20h) [reset = 0h]: [0] • MCBSP_XCERE1 Register (Offset = 2Ch) [reset = 0h]: [0] • MCBSP_XCERE2 Register (Offset = 34h) [reset = 0h]: [0] • MCBSP_XCERE3 Register (Offset = 3Ch) [reset = 0h]: [0][1] • McBSP Registers: [0] • MCBSP_MCR Register (Offset = 18h) [reset = 0h]: [0]

11.10.6.16 MCBSP_PCR Register (Offset = 24h) [reset = 0h]

The serial port is configured via the serial port control register (MCBSP_SPCR) and the pin control register (MCBSP_PCR). The MCBSP_PCR contains McBSP status control bits. The MCBSP_PCR is shown in Figure 11-772 and described in Table 11-1677.

Table 11-1676. MCBSP_PCR Instances

Instance	Physical Address
MCBSP	0234 6024h

Figure 11-772. MCBSP_PCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				FSXM	FSRM	CLKXM	CLKRM
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
SCLKME	RESERVED			FSXP	FSRP	CLKXP	CLKRP
R/W-0h	R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1677. MCBSP_PCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect. If writing to bit [12] and bit [13] in this field, always write the default value of 0 to ensure proper McBSP operation.
11	FSXM	R/W	0h	Transmit frame-synchronization mode bit. • 0 = Frame-synchronization signal is derived from an external source. • 1 = Frame-synchronization signal is determined by FSGM bit in MCBSP_SRGR.
10	FSRM	R/W	0h	Receive frame-synchronization mode bit. • 0 = Frame-synchronization signal is derived from an external source. FSR is an input pin. • 1 = Frame-synchronization signal is generated internally by the sample-rate generator. FSR is an output pin.
9	CLKXM	R/W	0h	Transmit clock mode bit. When CLKSTP bit in MCBSP_SPCR is cleared to 0: • 0 = CLKX is an input pin and is driven by an external clock. • 1 = CLKX is an output pin and is driven by the internal sample-rate generator.
8	CLKRM	R/W	0h	Receive clock mode bit. • **Digital loop back mode is disabled (DLB = 0 in MCBSP_SPCR):** – 0 = CLKR is an input pin and is driven by an external clock. – 1 = CLKR is an output pin and is driven by the internal sample-rate generator. • **Digital loop back mode is enabled (DLB = 1 in MCBSP_SPCR):** – 0 = Receive clock (not the CLKR pin) is driven by transmit clock (CLKX) that is based on CLKXM bit. CLKR pin is in high-impedance state. – 1 = CLKR is an output pin and is driven by the transmit clock. The transmit clock is based on CLKXM bit.

Table 11-1677. MCBSP_PCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	SCLKME	R/W	0h	Sample rate generator input clock mode bit. The sample rate generator can produce a clock signal, CLKG. The frequency of CLKG is: *CLKG frequency = Input clock frequency/(CLKGDV + 1)* SCLKME is used in conjunction with the CLKSM bit in the sample rate generator register (MCBSP_SRGR) to select the input clock. A DSP reset selects the McBSP internal input clock as the input clock and forces the CLKG frequency to 1/2 the McBSP internal input clock frequency. 0 = The input clock for the sample rate generator is taken from the CLKS or from the McBSP internal input clock, depending on the value of the CLKSM bit in MCBSP_SRGR 1 = The input clock for the sample rate generator is taken from the CLKR pin or from the CLKX pin, depending on the value of the CLKSM bit in MCBSP_SRGR For more information about choosing an Input Clock for Sample Rate Generator with CLKSM bit see Table 11-1595.
6-4	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect. If writing to bit [6] in this field, always write the default value of 0 to ensure proper McBSP operation.
3	FSXP	R/W	0h	Transmit frame-synchronization polarity bit. • 0 = Transmit frame-synchronization pulse is active high. • 1 = Transmit frame-synchronization pulse is active low.
2	FSRP	R/W	0h	Receive frame-synchronization polarity bit. • 0 = Receive frame-synchronization pulse is active high. • 1 = Receive frame-synchronization pulse is active low.
1	CLKXP	R/W	0h	Transmit clock polarity bit. • 0 = Transmit data driven on rising edge of CLKX. • 1 = Transmit data driven on falling edge of CLKX.
0	CLKRP	R/W	0h	Receive clock polarity bit. • 0 = Receive data sampled on falling edge of CLKR. • 1 = Receive data sampled on rising edge of CLKR.

Table 11-1678. Register Call Summary for MCBSP_PCR

McBSP Functional Description
- Bit Clock Polarity: CLKSP: [0]
- Bit Clock and Frame Synchronization: [0][1]
- Transmit Frame Synchronization Selection: FSXM and FSGM: [0]
- Clearing the Serial Port Control Register (MCBSP_SPCR): [0][1]
- Receive Clock Selection: DLB, CLKRM: [0][1]
- Receive Frame Synchronization Selection: DLB and FSRM: [0]
- Power Management: [0]
- Transmit Clock Selection: CLKXM: [0][1]
- Stopping Clocks: [0]
- Input Clock Source Mode: CLKSM and SCLKME: [0][1]
- Data Clock Generation: [0]
- Frame and Clock Operation: [0][1]
- Transmit Empty: XEMPTY: [0]

McBSP Registers
- MCBSP_SPCR Register (Offset = 8h) [reset = 0h]: [0]
- MCBSP_SRGR Register (Offset = 14h) [reset = 20000001h]: [0][1][2][3]
- MCBSP_PCR Register (Offset = 24h) [reset = 0h]: [0][1][2]
- McBSP Registers: [0]

Table 11-1678. Register Call Summary for MCBSP_PCR (continued)

McBSP Programming Guide
• General Initialization: [0] • Programming the MCBSP Registers for the Desired Receiver Configuration (Step 2): [0][1][2][3][4][5] • Programming the MCBSP Registers for the Desired Transmitter Operation (Step 2): [0][1][2][3][4][5]

11.10.6.17 MCBSP_BFIFOREV Register (Offset = 0h) [reset = 44311100h]

The Buffer FIFO (BFIFO) revision identification register (MCBSP_BFIFOREV) contains revision data for the Buffer FIFO (BFIFO). The MCBSP_BFIFOREV is shown in Figure 11-773 and described in Table 11-1680.

Table 11-1679. MCBSP_BFIFOREV Instances

Instance	Physical Address
MCBSP	0234 6080h

Figure 11-773. MCBSP_BFIFOREV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-44311100h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1680. MCBSP_BFIFOREV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	44311100h	TI internal data. Identifies revision of Buffer FIFO.

Table 11-1681. Register Call Summary for MCBSP_BFIFOREV

McBSP Registers
- MCBSP_BFIFOREV Register (Offset = 0h) [reset = 44311100h]: [0][1]
- McBSP Registers: [0]

11.10.6.18 MCBSP_WFIFOCTL Register (Offset = 10h) [reset = 204h]

The Write FIFO control register (MCBSP_WFIFOCTL) is shown in Figure 11-774 and described in Table 11-1683.

NOTE: The WNUMEVT and WNUMDMA values must be set prior to enabling the Write FIFO. If the Write FIFO is to be enabled, it must be enabled prior to taking the McBSP out of reset.

Table 11-1682. MCBSP_WFIFOCTL Instances

Instance	Physical Address
MCBSP	0234 6090h

Figure 11-774. MCBSP_WFIFOCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							WENA
R-0h							R/W-0h
15	14	13	12	11	10	9	8
WNUMEVT							
R/W-2h							
7	6	5	4	3	2	1	0
WNUMDMA							
R/W-4h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1683. MCBSP_WFIFOCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	WENA	R/W	0h	Write FIFO enable bit. • 0 = Write FIFO is disabled. The WLVL bit in the Write FIFO status register (MCBSP_WFIFOSTS) is reset to 0 and the pointers are initialized, that is, the Write FIFO is "flushed." • 1 = Write FIFO is enabled. If the Write FIFO is to be enabled, it must be enabled prior to taking the McBSP out of reset.
15-8	WNUMEVT	R/W	2h	Write word count per DMA event (32-bit). When the Write FIFO has space for at least *WNUMEVT* words of data, then an XEVT (transmit DMA event) is generated to the host/DMA controller. This value must be set prior to enabling the Write FIFO. • 0 = 0 words • 1h = 1 word • 2h = 2 words ... • 40h = 64 words • 41h-FFh = Reserved
7-0	WNUMDMA	R/W	4h	Write word count per transfer (32-bit words). Upon a transmit DMA event from the McBSP, WNUMDMA words are transferred from the Write FIFO to the McBSP. This value must be set prior to enabling the Write FIFO. • 0 = 0 words • 1 = 1 word • 2h-FFh = Reserved

Table 11-1684. Register Call Summary for MCBSP_WFIFOCTL

McBSP Functional Description • McBSP Buffer FIFO (BFIFO): [0] • BFIFO Data Transmission: [0]
McBSP Registers • MCBSP_WFIFOCTL Register (Offset = 10h) [reset = 204h]: [0] • McBSP Registers: [0]

11.10.6.19 MCBSP_WFIFOSTS Register (Offset = 14h) [reset = 0h]

The Write FIFO status register (MCBSP_WFIFOSTS) is shown in Figure 11-775 and described in Table 11-1686.

Table 11-1685. MCBSP_WFIFOSTS Instances

Instance	Physical Address
MCBSP	0234 6094h

Figure 11-775. MCBSP_WFIFOSTS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								WLVL							
R-0h																								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1686. MCBSP_WFIFOSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	WLVL	R	0h	Write level. Number of 32-bit words currently in the Write FIFO. • 0 = 0 words currently in the Write FIFO. • 1h = 1 word currently in the Write FIFO. • 2h = 2 words currently in the Write FIFO. ... • 40 = 64 words currently in the Write FIFO. • 41h-FFh = Reserved

Table 11-1687. Register Call Summary for MCBSP_WFIFOSTS

McBSP Registers
- MCBSP_WFIFOCTL Register (Offset = 10h) [reset = 204h]: [0]
- McBSP Registers: [0]
- MCBSP_WFIFOSTS Register (Offset = 14h) [reset = 0h]: [0]

11.10.6.20 MCBSP_RFIFOCTL Register (Offset = 18h) [reset = 204h]

The Read FIFO control register (MCBSP_RFIFOCTL) is shown in Figure 11-776 and described in Table 11-1689.

NOTE: The RNUMEVT and RNUMDMA values must be set prior to enabling the Read FIFO. If the Read FIFO is to be enabled, it must be enabled prior to taking the McBSP out of reset.

Table 11-1688. MCBSP_RFIFOCTL Instances

Instance	Physical Address
MCBSP	0234 6098h

Figure 11-776. MCBSP_RFIFOCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							RENA
R-0h							R/W-0h
15	14	13	12	11	10	9	8
RNUMEVT							
R/W-2h							
7	6	5	4	3	2	1	0
RNUMDMA							
R/W-4h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1689. MCBSP_RFIFOCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	RENA	R/W	0h	Read FIFO enable bit. • 0 = Read FIFO is disabled. The RLVL bit in the Read FIFO status register (MCBSP_RFIFOSTS) is reset to 0 and the pointers are initialized, that is, the Read FIFO is "flushed." • 1 = Read FIFO is enabled. If the Read FIFO is to be enabled, it must be enabled prior to taking the McBSP out of reset.
15-8	RNUMEVT	R/W	2h	Read word count per DMA event (32-bit). When the Read FIFO contains at least *RNUMEVT* words of data, then an REVT (receive DMA event) is generated to the host/DMA controller. This value must be set prior to enabling the Read FIFO. • 0 = 0 words • 1h = 1 word • 2h = 2 words ... • 40h = 64 words • 41h-FFh = Reserved
7-0	RNUMDMA	R/W	4h	Read word count per transfer (32-bit words). Upon a receive DMA event from the McBSP, the Read FIFO reads *RNUMDMA* words from the McBSP. This value must be set prior to enabling the Read FIFO. • 0 = 0 words • 1 = 1 word • 2h-FFh = Reserved

Table 11-1690. Register Call Summary for MCBSP_RFIFOCTL

McBSP Functional Description • McBSP Buffer FIFO (BFIFO): [0] • BFIFO Data Reception: [0]
McBSP Registers • MCBSP_RFIFOCTL Register (Offset = 18h) [reset = 204h]: [0] • McBSP Registers: [0]

11.10.6.21 MCBSP_RFIFOSTS Register (Offset = 1Ch) [reset = 0h]

The Read FIFO status register (MCBSP_RFIFOSTS) is shown in Figure 11-777 and described in Table 11-1692.

Table 11-1691. MCBSP_RFIFOSTS Instances

Instance	Physical Address
MCBSP	0234 609Ch

Figure 11-777. MCBSP_RFIFOSTS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								RLVL							
R-0h																								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1692. MCBSP_RFIFOSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	RLVL	R	0h	Read level. Number of 32-bit words currently in the Read FIFO. • 0 = 0 words currently in the Read FIFO. • 1h = 1 word currently in the Read FIFO. • 2h = 2 words currently in the Read FIFO. ... • 40h = 64 words currently in the Read FIFO. • 41h-FFh = Reserved

Table 11-1693. Register Call Summary for MCBSP_RFIFOSTS

McBSP Registers
- MCBSP_RFIFOCTL Register (Offset = 18h) [reset = 204h]: [0]
- MCBSP_RFIFOSTS Register (Offset = 1Ch) [reset = 0h]: [0]
- McBSP Registers: [0]

11.11 Media Local Bus (MLB)

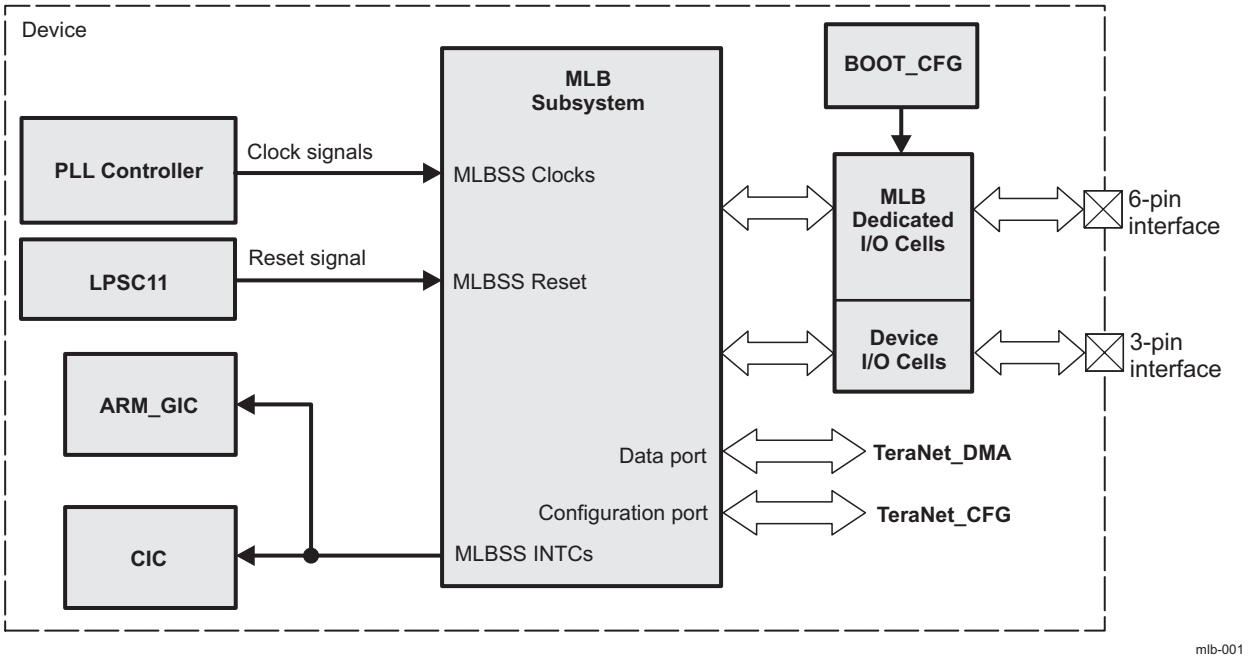
This chapter describes the function, operation, and configuration of the Media Local Bus (MLB) subsystem in the device.

11.11.1 MLB Overview

The Media Local Bus subsystem (MLBSS) is based on a module designed by SMSC. This module provides a MediaLB/MediaLB+ controller and an interface to other MediaLB/MediaLB+ devices. The MediaLB/MediaLB+ interface allows also connection to a MOST (Media Oriented Systems Transport) network controller.

Figure 11-778 shows the MLBSS overview.

Figure 11-778. MLB Overview



11.11.1.1 MLB Features

The MLBSS supports the following features:

- 3-pin MediaLB 3.3V LVCMOS I/Os compliant to MediaLB Physical Layer and Link Layer Specification v4.2
- 6-pin MediaLB+ low-voltage differential signaling (LVDS) I/Os (3 differential pairs) compliant to MediaLB Physical Layer and Link Layer Specification v4.2
- MediaLB core functionality compliant to MediaLB Physical Layer and Link Layer Specification v4.2
- Supports 256/512/1024Fs in 3-pin mode and 2048Fs in 6-pin mode
- Supports all types of transfer (synchronous stream data, asynchronous packet data, control message data, and isochronous data) over 64 logical channels
- Supports single 32-bit TeraNet_CFG slave interface for configuration
- Supports single 32-bit TeraNet_DMA master interface with burst capability for DMA transfers into system memory. The maximum burst size is 32 Bytes
- Has 16 KB buffer for all types of transfers in the subsystem
- Dedicated BOOT_CFG bits for controlling the MLBSS priority on the system interconnect

The MLBSS does not support:

- 5-pin mode

- Digital Transmission Content Protection (DTCP) cipher accelerators

11.11.2 MLB Environment

Figure 11-779 shows the MLBSS associated signals available on device pads.

Figure 11-779. MLB Subsystem Environment

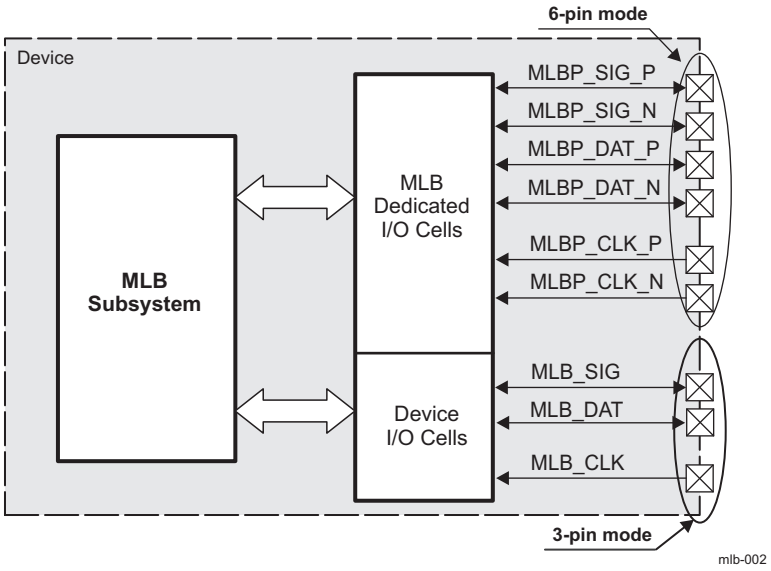


Table 11-1694 describes the MLBSS I/O signals used in both 3-pin and 6-pin modes.

Table 11-1694. MLB Subsystem I/O Description

Signal/Pad Name	I/O[1]	Description
	6-pin mode	
MLBP_SIG_P	I/O	Differential pair signal line
MLBP_SIG_N	I/O	
MLBP_DAT_P	I/O	Differential pair data line
MLBP_DAT_N	I/O	
MLBP_CLK_P	I	Differential pair clock line
MLBP_CLK_N	I	
	3-pin mode	
MLB_SIG	I/O	Single ended signal line
MLB_DAT	I/O	Single ended data line
MLB_CLK	I	Single ended clock line

(1) I = Input; O = Output; I/O = Bidirectional

11.11.2.1 MLB I/O Cell Controls

NOTE: There are two I/O cells specially intended for MLB when 6-pin mode is selected. For more information about MLB I/O cell controls see Section 5.1.3.1.11, *Registers for Control of the MLB IOs*, in Section 5.1, *Control Module (BOOT_CFG)*.

11.11.3 MLB Integration

Figure 11-780 shows the integration of the MLB subsystem in the device.

Figure 11-780. MLB Subsystem Integration

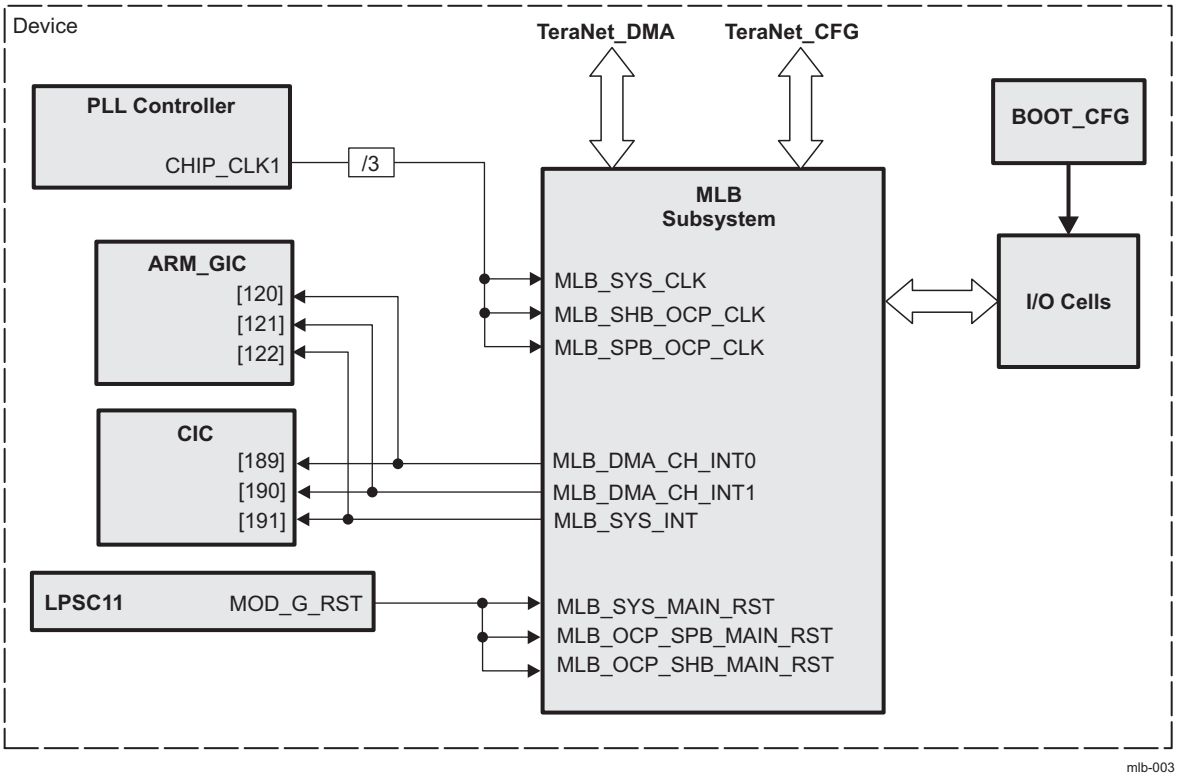


Table 11-1695 through Table 11-1697 summarize the integration of the MLB subsystem in the device.

Table 11-1695. MLB Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
MLB	PD5	LPSC11	TeraNet_DMA (master) TeraNet_CFG (slave)

Table 11-1696. MLB Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
MLB	MLB_SYS_CLK	CHIP_CLK1/3	PLL Controller	Functional clock for the MLB subsystem
	MLB_SHB_OCP_CLK	CHIP_CLK1/3	PLL Controller	Interface clock for the data port of the MLB subsystem
	MLB_SPB_OCP_CLK	CHIP_CLK1/3	PLL Controller	Interface clock for the configuration port of the MLB subsystem
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
MLB	MLB_SYS_MAIN_RST	MOD_G_RST	LPSC11	Asynchronous reset for the MLB subsystem
	MLB_OCP_SPB_MAIN_RST			
	MLB_OCP_SHB_MAIN_RST			

Table 11-1697. MLB Hardware Requests

Interrupt Requests				
Module Instance	**Event Name**	**Mapped To Input Event [Number]**		**Description**
		ARM GIC	**CIC**	
MLB	MLB_DMA_CH_INT0	[120]	[189]	MLB DMA channel interrupt 0
	MLB_DMA_CH_INT1	[121]	[190]	MLB DMA channel interrupt 1
	MLB_SYS_INT	[122]	[191]	MLB system interrupt

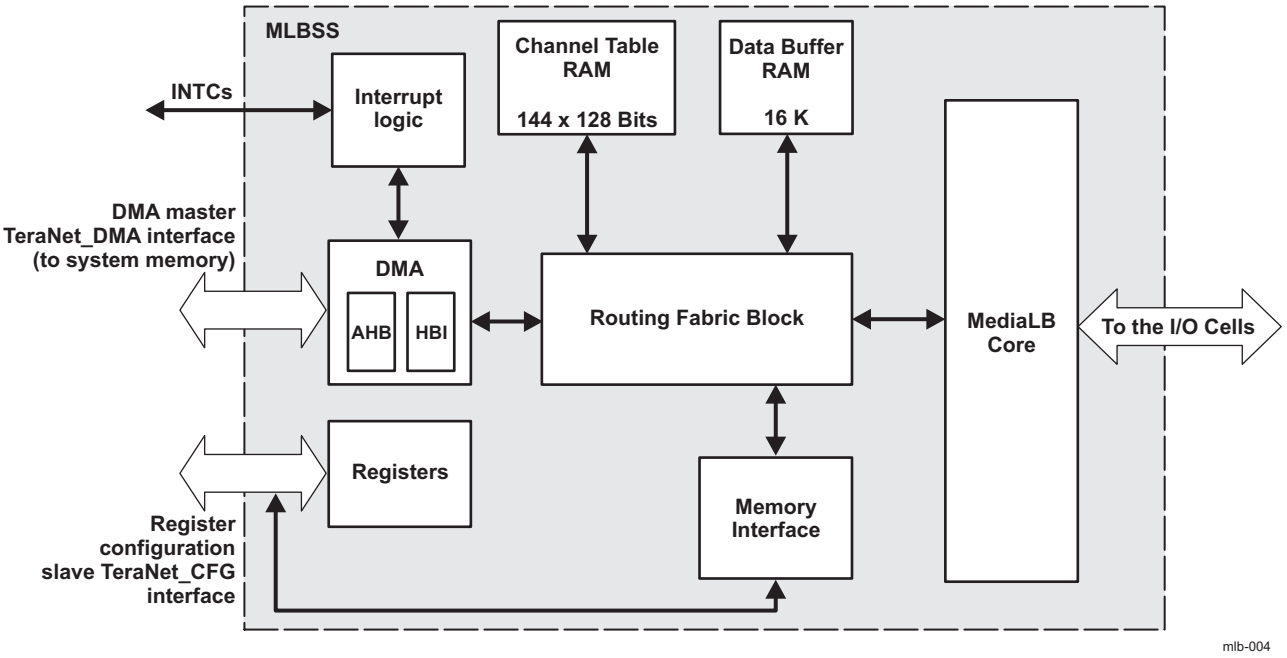
11.11.4 MLB Functional Description

11.11.4.1 Block Diagram

Figure 11-781 shows the MLBSS block diagram. It includes the following functional blocks:

- MediaLB core – Implements the physical and link layer requirements of either a 3-pin or the 6-pin interface. Serial-to-parallel/parallel-to-serial are also implemented along with MediaLB frame synchronization.
- Routing fabric block – Manages the flow of data between the MediaLB core and the DMA block, implementing bus arbiter and muxing logic to the channel table RAM and the data buffer RAM.
- Channel table RAM (CTR) – Used for storing channel descriptors for managing accesses to dynamic buffers in the data buffer RAM.
- Data buffer RAM (DBR) – Provides dynamic circular buffering between the transmit and receive devices.
- Memory interface – Implements a bridge between the configuration TeraNet_CFG slave interface and the channel table RAM or data buffer RAM interfaces.
- DMA – Implements a bus bridge between the DMA master and the routing fabric block.
- Registers – Used for configuration.
- Interrupt logic.

Figure 11-781. MLBSS Structural Overview



AHB - Advanced High-performance Bus

HBI - Host Bus Interface

11.11.4.1.1 MediaLB Core Block

Although the MediaLB block supports both a MediaLB 3-pin and 6-pin interface, only one of these interfaces is active at any given time. The selection is done through the MLB_MLBC0[5] MLBPEN bit. Table 11-1694 describes the signals associated with 3-pin and 6-pin modes. Both MediaLB interfaces provide real-time access to all network data types including streaming, packet, control, and isochronous data.

The MediaLB 3-pin interface supports the MediaLB protocol for single-ended 3-pin mode with a maximum data rate of 1024Fs.

The MediaLB 6-pin interface supports the MediaLB protocol for high-speed differential 6-pin mode, with a maximum data rate of 2048Fs.

There is a set of physical channels implemented for exchanging data over the MediaLB 3-pin or 6-pin interface. These physical channels (called quadlets) are 4-byte wide and can also be grouped together to form logical channels. These logical channels are referenced using channel addresses, and define a uni-directional path between a specific MediaLB device transmitting data and another specific MediaLB device receiving data. In other words, the logical channel is a group of multiple quadlets, which selects a unique pair of MediaLB devices with a unique channel address. The logical channels, configured by the system software can be of any combination of channel types (synchronous, asynchronous, control, or isochronous) and direction (transmit/receive).

The MediaLB channel addresses are mapped to the logical channels as shown in Table 11-1698.

Table 11-1698. MediaLB Channel Address to Logical Channel Mapping

MediaLB Channel Address	Logical Channel Number
02h	1
04h	2
06h	3
…	…
007Ch	62
007Eh	63
01FEh	0 (1)

(1) The logical channel 0 is the system channel and is reserved.

Thus, 64 logical channels can be supported and one of them is the system channel which is reserved.

NOTE: The MediaLB channel address should always be an even number.

11.11.4.1.2 Routing Fabric Block

The routing fabric block manages the flow of data between the MediaLB core and the DMA block. Bus multiplexers and a bus arbiter are implemented in the routing fabric block for accessing the channel table RAM and data buffer RAM. Each DMA controller in the routing fabric uses channel descriptors to manage access to dynamic buffers in the data buffer RAM. These channel descriptors are stored in the channel table RAM.

11.11.4.1.3 Data Buffer RAM

The data buffer RAM is 8-bit × 16k entries deep and provides dynamic circular buffering between the transmit and receive devices. The size and location of each data buffer is defined by software in the channel descriptor table, which is located in the channel table RAM.

The DMA controllers in the routing fabric are responsible for ensuring that the circular buffers do not overflow or underflow. Each channel type (that is, synchronous, isochronous, asynchronous and control) has full and empty detection.

11.11.4.1.4 Channel Table RAM

The channel table RAM is 128-bit × 144-entry (max). It allows system software to dynamically configure channel routing and allocate data buffers in the data buffer RAM.

The channel table RAM is logically divided into three subtables:

- Channel descriptor table
- DMA descriptor table
- Channel allocation table

The details of the channel descriptor table and channel allocation table are described in Table 11-1699.

The details of the DMA descriptor table are described in Table 11-1709.

Table 11-1699. Channel Table RAM Address Mapping

Label	Address	Bits 127-96	Bits 95-64	Bits 63-32	Bits 31-0
Channel Descriptor Table (CDT):					
Channel descriptor table	00h	CDT0[127-0], CL = 0			
	01h	CDT1[127-0], CL = 1			
	02h	CDT2[127-0], CL = 2			
	…	…			
	3Dh	CDT61[127-0], CL = 61			
	3Eh	CDT62[127-0], CL = 62			
	3Fh	CDT63[127-0], CL = 63			
DMA Descriptor Table (DDT):					
DMA descriptor table	40h	DDT0[127-0], CAT64			
	41h	DDT1[127-0], CAT65			
	42h	DDT2[127-0], CAT66			
	…	…			
	7Dh	DDT61[127-0], CAT125			
	7Eh	DDT62[127-0], CAT126			
	7Fh	DDT63[127-0], CAT127			
Channel Allocation Table (CAT):					
Channel allocation table for MediaLB	80h	CAT7	CAT6	CAT5	CAT4	CAT3	CAT2	CAT1	CAT0
	...	...	...	...	...	...	...	...	...
	87h	CAT63	CAT62	CAT61	CAT60	CAT59	CAT58	CAT57	CAT56
Channel allocation table for DMA	88h	CAT71	CAT70	CAT69	CAT68	CAT67	CAT66	CAT65	CAT64
	...	...	...	...	...	...	...	...	...
	8Fh	CAT127	CAT126	CAT125	CAT124	CAT123	CAT122	CAT121	CAT120

NOTE: A fixed relationship between DMA descriptor table entries and DMA channel allocation table entries exists. When using DMA channel 0 (CAT64) software should program DDT0. When using DMA channel 1 (CAT65) software should program DDT1, in case of CAT66 DDT2 should be programmed and so on. In case of CAT127 software should program DDT63.

The following base addresses are valid:

- Channel descriptor table has base address 00h
- DMA descriptor table has base address 40h
- Channel allocation table for MediaLB has base address 80h
- Channel allocation table for DMA has base address 88h

11.11.4.1.4.1 Channel Allocation Table

The channel allocation table comprises 16 entries of the channel table RAM. Each entry is 16-bit. The first entry starts at address 80h and the sixteenth entry at address 8Fh. Each 16-bit channel allocation table entry represents a logical connection to or from a transmit/receive device (that is, MediaLB or DMA channel). All entries are indexed according to a fixed physical address assigned to every Rx/Tx channel. This is shown in Table 11-1700. The value stored in a channel allocation table entry includes a 6-bit connection label (CL[5-0]), which provides a pointer to the channel descriptor table. To complete a logical channel and form a routing connection, system software must assign the same connection label to both the Rx and Tx channels.

Table 11-1700. Channel Allocation Table Entry Map

Peripheral	Number of Tx Channels Allowed	Number of Rx Channels Allowed	Channel Allocation Table Start Index	Channel Allocation Table End Index	Entries
MediaLB	0 to 64	64 - (Number of Tx channels)	0	63	64
DMA	0 to 64	64 - (Number of Tx channels)	64	127	64

The format of a full channel allocation table entry is shown in Table 11-1701. All reserved bits should be written to zero.

Table 11-1701. Format Of Channel Allocation Table Entry

Channel Type	15	14	13	12	11	10 8	7 6	5 0
Isochronous	Reserved	FCE	Reserved	RNW	CE	CT[2-0] = 3	Reserved	CL[5-0]
Asynchronous	Reserved		MT	RNW	CE	CT[2-0] = 2	Reserved	CL[5-0]
Control	Reserved		MT	RNW	CE	CT[2-0] = 1	Reserved	CL[5-0]
Synchronous	Reserved	MFE	MT	RNW	CE	CT[2-0] = 0	Reserved	CL[5-0]

The field descriptions of a channel allocation table entry are shown in Table 11-1702.

Table 11-1702. Field Descriptions Of Channel Allocation Table Entry

Field	Description
CL[5-0]	Connection Label (offset into channel descriptor table)
CT[2-0]	Channel type
	111 = Reserved
	110 = Reserved
	101 = Reserved
	100 = Reserved
	011 = Isochronous
	010 = Asynchronous
	001 = Control
	000 = Synchronous
CE	Channel enable
	0 = Disabled
	1= Enabled
RNW	Read, not Write (1)
	1 = Read
	0 = Write
MT	Mute Bit (2)
	1= Enabled
	0 = Disabled
FCE	Flow control enable (3)
	1 = Enabled
	0 = Disabled

(1) For a Tx channel (from the host to the MediaLB interface):
- DMA channel allocation table entry: RNW = 0 (write)
- MediaLB channel allocation table entry: RNW = 1 (read)

For a Rx channel (data from MediaLB to host):
- DMA channel allocation table entry: RNW = 1 (read)
- MediaLB channel allocation table entry: RNW = 0 (write)

(2) When set for synchronous channels, the MT bit forces Rx channels to write zeros into the channel data buffer, and Tx channels to output zeros on the physical interface. When set for asynchronous and control channels, the MT bit causes DMA to halt at a packet boundary. Not valid for isochronous channels.

(3) The FCE bit is used by MediaLB isochronous Rx channels only.

Table 11-1702. Field Descriptions Of Channel Allocation Table Entry (continued)

Field	Description
MFE	Multi-frame per sub-buffer enable [4]
	1 = Enabled
	0 = Disabled
Reserved	Reserved. Software writes zeros to all reserved bits when the entry is initialized. The reserved bits are read-only after initialization.

[4] The MFE bit is used by MediaLB synchronous channels only.

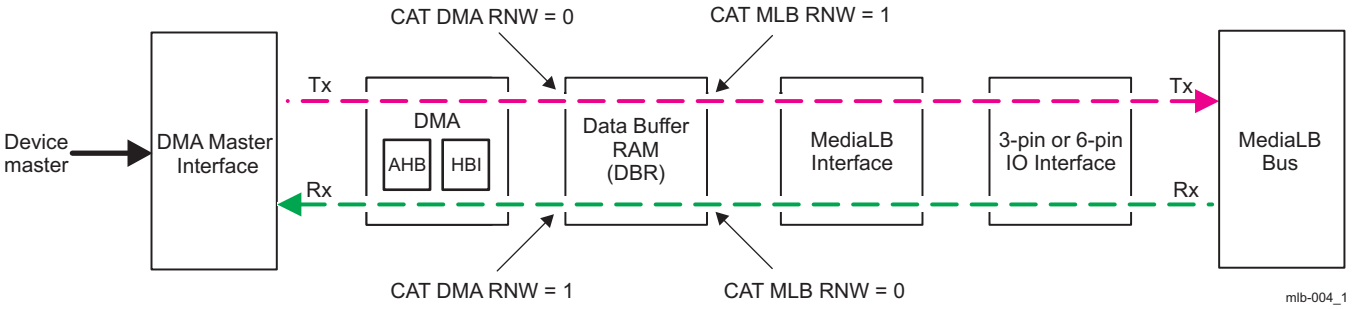
11.11.4.1.4.1.1 Channel Setup

Data direction in the MLBSS is in reference to the DBR; therefore, the data direction of DMA entries corresponding to the same channel is reversed for the CAT DMA and the CAT MLB.

- For a Tx channel (from Device master to the MediaLB interface):
  - CAT DMA entry: RNW = 0 (write)
  - CAT MLB entry: RNW = 1 (read)
- Conversely, for a Rx channel (data from MediaLB to Device master):
  - CAT DMA entry: RNW = 1 (read)
  - CAT MLB entry: RNW = 0 (write)

Figure 11-782 illustrates the directional relationship in the MLBSS.

Figure 11-782. MLBSS DBR Directional Relationship



11.11.4.1.4.2 Channel Descriptor Table

The channel descriptor table comprises 64 channel table RAM entries. Each entry is 128-bit. The first entry starts at address 00h and the 64th entry at address 3Fh. Each 128-bit entry of the channel descriptor table is referenced by a connection label and contains information about data buffer in the data buffer RAM (that is, buffer size, or address pointers). The format of each channel descriptor table entry depends on the channel type (synchronous, isochronous, asynchronous, or control).

Software must write all reserved channel descriptor bits to ’0’ when the entry is initialized.

Synchronous Channel Descriptor Table

The MLBSS provides two modes of operation (standard and multi-frame per sub-buffer) to provide flexibility for implementing synchronous channels.

Channels set up for standard mode require less buffer space. For channels configured for standard mode, the host controller must transfer one full frame of streaming data in/out of the data buffer of each streaming channel.

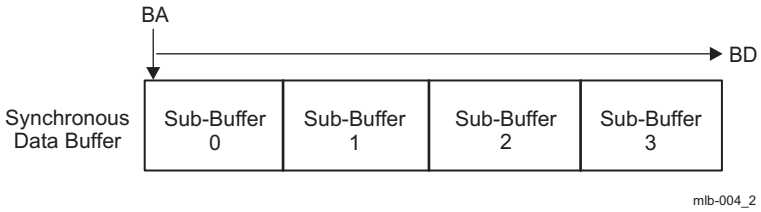
Channels set up for multiple-frames per sub-buffer mode require more buffer space. For channels configured for multiple-frames per sub-buffer mode, the host controller must transfer N full frames of streaming data in/out of the data buffer of each streaming channel.

To set up a channel in multi-frame per sub-buffer mode:

- Program the MLB_MLBC0[17-15] FCNT bit field to select the number of frames per sub-buffer
- Program the channel allocation table to enable multi-frame sub-buffering (MFE = 1) for each particular channel
- Set the buffer depth (BD[11-0]) in the channel descriptor table
- Repeat for additional synchronous channels

A sample synchronous data buffer is shown in Figure 11-783. Each data buffer contains four sub-buffers and each sub-buffer contains space for 1 to 64 frames of data, determined by MLB_MLBC0[17-15] FCNT bit field.

Figure 11-783. Synchronous Data Buffer Structure



The format of a synchronous channel descriptor table entry is shown in Table 11-1703. All reserved bits should be written to zero.

Table 11-1703. Format Of Synchronous Channel Descriptor Table Entry

Bit Offset	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	WSBC[1-0]		Reserved													
16	RSBC[1-0]		Reserved													
32	Reserved															
48	Reserved															
64	WSTS[3-0]				WPTR[11-0]											
80	RSTS[3-0]				RPTR[11-0]											
96	Reserved				BD[11-0]											
112	Reserved		BA[13-0]													

The field descriptions of a synchronous channel descriptor table entry are shown in Table 11-1704.

Table 11-1704. Field Descriptions Of Synchronous Channel Descriptor Table Entry

Field	Description	Details	Accessibility (1)
BA	Buffer base address	BA can start at any byte in the 16k data buffer RAM	r, w
BD	Buffer depth	1. BD = size of buffer in bytes - 1 2. Buffer end address = BA + BD 3. BD = 4 × m × bpf - 1, where: m = frames per sub-buffer (for MFE = 0, m = 1) bpf = bytes per frame	r, w
RPTR	Read pointer	1. Software initializes to zero, hardware updates 2. Counts the read address offset within a buffer 3. DMA read address = BA + RPTR	r, w, u
WPTR	Write pointer	1. Software initializes to zero, hardware updates 2. Counts the write address offset within a buffer 3. DMA write address = BA + WPTR	r, w, u

(1)

- r - software readable
- w - software writable
- u - updated periodically by hardware

Table 11-1704. Field Descriptions Of Synchronous Channel Descriptor Table Entry (continued)

Field	Description	Details	Accessibility [1]
RSBC	Read sub-buffer counter	1. Software initializes to zero, hardware updates 2. Counts the read sub-buffer offset 3. DMA uses for pointer management	r, w, u
WSBC	Write sub-buffer counter	1. Software initializes to zero, hardware updates 2. Counts the write sub-buffer offset 3. DMA uses for pointer management	r, w, u
RSTS	Read status	1. Software initializes to zero, hardware updates 2. RSTS states: (Only for DMA associated with MLB) xxx0 = normal operation (no mute) xxx1 = normal operation (mute) xx0x = idle	r, w, u
WSTS	Write status	1. Software initializes to zero, hardware updates 2. WSTS states: (Only for DMA associated with MLB) xxx0 = normal operation (no mute) xxx1 = normal operation (mute) xx0x = idle 1xxx = command protocol error	r, w, u
Reserved	Reserved	Software writes a zero to all Reserved bits when the entry is initialized. The Reserved bits are Read-only after initialization.	r, w, u

Isochronous Channel Descriptor Table

The format of an isochronous channel descriptor table entry is shown in Table 11-1705. All reserved bits should be written to zero.

Table 11-1705. Format Of Isochronous Channel Descriptor Table Entry

Bit Offset	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	Reserved															
16	Reserved															
32	Reserved							BS[8-0]								
48	Reserved															
64	WSTS[2-0]			WPTR[12-0]												
80	RSTS[2-0]			RPTR[12-0]												
96	Reserved			BD[12-0]												
112	BF	Reserved	BA[13-0]													

The field descriptions of an isochronous channel descriptor table entry are shown in Table 11-1706.

Table 11-1706. Field Descriptions Of Isochronous Channel Descriptor Table Entry

Field	Description	Details	Accessibility [1]
BA	Buffer Base Address	BA can start at any byte in the 16k data buffer RAM	r, w

(1)
- r - software readable
- w - software writable
- u - updated periodically by hardware

Table 11-1706. Field Descriptions Of Isochronous Channel Descriptor Table Entry (continued)

Field	Description	Details	Accessibility [1]
BD	Buffer Depth	1. BD = size of buffer in bytes - 1 2. Buffer end address = BA + BD 3. Isochronous buffers must be large enough to hold at least 3 blocks (packets) of data 4. Buffer depth must be a integer multiple of blocks	r, w
BF	Buffer Full	1. Software initializes to zero, hardware updates 2. DMA write hardware sets BF when the buffer is full 3. DMA read hardware clears BF when the buffer is empty 4. BF is valid only when the buffer is full or empty, otherwise ignore	r, w, u
BS	Block Size	1. BS defines when to begin the DMA to the data buffer 2. BS = buffer block size in bytes - 1 3. For Rx channels, the DMA writes start when the number of empty bytes (SPACE) in the data buffer >= the block size 4. For Tx channels, the DMA reads start when the number of valid bytes (VALID) in the data buffer >= the block size	r, w, u
RPTR	Read Pointer	1. Software initializes to zero, hardware updates 2. Counts the read address offset within a buffer 3. DMA read address = BA + RPTR	r, w, u
WPTR	Write Pointer	1. Software initializes to zero, hardware updates 2. Counts the write address offset within a buffer 3. DMA write address = BA + WPTR	r, w, u
RSTS	Read status	1. Software initializes to zero, hardware updates 2. RSTS Status states (Only for DMA associated with MLB) xx1 = active xx0 = idle	r, w, u
WSTS	Write status	1. Software initializes to zero, hardware updates 2. WSTS states: (Only for DMA associated with MLB) xx1 = active xx= idle x1x = command protocol error 1xx = buffer overflow (FCE = 0 only)	r, w, u
Reserved	Reserved	Software writes a zero to all reserved bits when the entry is initialized. The reserved bits are read-only after initialization.	r, w, u

Asynchronous and Control Channel Description Table

The format of an asynchronous and control channel descriptor table entry is shown in Table 11-1707. All reserved bits should be written to zero.

Table 11-1707. Format Of Asynchronous and Control Channel Descriptor Table Entry

Bit Offset	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	WPC[4-0]					Reserved										
16	RPC[4-0]					Reserved										
32	Reserved	WPC[7-5]				Reserved										

Table 11-1707. Format Of Asynchronous and Control Channel Descriptor Table Entry (continued)

Bit Offset	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
48	Reserved	RPC[7-5]			Reserved											
64	WSTS[3-0]				WPTR[11-0]											
80	RSTS[3-0]				RPTR[11-0]											
96	RSTS[4]	WSTS[4]	Reserved		BD[11-0]											
112	Reserved		BA[13-0]													

The field descriptions of an asynchronous and control channel descriptor table entry are shown in Table 11-1708.

Table 11-1708. Field Descriptions Of Asynchronous And Control Channel Descriptor Table Entry

Field	Description	Details	Accessibility (1)
BA	Buffer Base Address	BA can start at any byte in the 16k data buffer RAM	r, w
BD	Buffer Depth	1. BD = size of buffer in bytes - 1 2. Buffer end address = BA + BD 3. BD >= max packet length - 1	r, w
RPC	Read Packet Count	1. Software initializes to zero, hardware updates 2. Used in conjunction with WPC, RPTR and WPTR to determine if the buffer is empty or full	r, w, u
WPC	Write Packet Count	1. Software initializes to zero, hardware updates 2. Used in conjunction with RPC, RPTR and WPTR to determine if the buffer is empty or full	r, w, u
RPTR	Read Pointer	1. Software initializes to zero, hardware updates 2. Counts the read address offset within a buffer 3. DMA read address = BA + RPTR	r, w, u
WPTR	Write Pointer	1. Software initializes to zero, hardware updates 2. Counts the write address offset within a buffer 3. DMA write address = BA + WPTR	r, w, u
RSTS	Read status	1. Software initializes to zero, hardware updates 2. Status states:(2) x0x00 = idle xx1xx = ReceiverProtocolError response received from Rx device 1xxxx = ReceiverBreak response received from Rx device	r, w, u
WSTS	Write status	1. Software initializes to zero, hardware updates 2. Status states:(2) x0x00 = idle xx1xx = command protocol error detected 1xxxx = AsyncBreak/ControlBreak command received from Tx device	r, w, u
Reserved	Reserved	Software writes a zero to all reserved bits when the entry is initialized. The Reserved bits are read-only after initialization.	r, w, u

(1)
- r - software readable
- w - software writable
- u - updated periodically by hardware

(2) Only valid for DMA pointers associated with the MediaLB core. Not valid for DMA block related pointers.

11.11.4.1.5 DMA Block

The DMA block manages data exchange between the local channel data buffers which reside in the MLBSS and a memory buffer residing in the system memory. To support system memory buffering, a ping-pong memory structure is implemented on a per channel basis using 128-bit descriptors for DMA descriptor table entries. 64 DMA descriptor table entries are directly mapped to the 64 DMA physical channels.

Each logical channel is assigned a separate 128-bit descriptor, defining data buffers in the system memory which are used by the DMA interface for this logical channel. The descriptors are stored at fixed addresses in the channel table RAM.

The description of the DMA descriptor table fields is shown in Table 11-1709.

Table 11-1709. Field Descriptions Of DMA Descriptor Table

Field	Numbers of bits	Description	Accessibilty (1)
CE	1	Channel enable: 0 = Disabled 1 = Enabled	r, w, u
LE	1	Endianess Enable: 0 = Big endian 1 = Little Endian	r, w
PG	1	Page pointer. Software initializes to zero, hardware writes thereafter. 0 = Ping buffer 1 = Pong buffer	r, w, u
RDY1	1	Buffer ready bit for ping buffer page: 0 = Not ready 1 = Ready	r, w
RDY2	1	Buffer ready bit for pong buffer page: 0 = Not ready 1 = Ready	r, w
DNE1	1	Buffer done bit for ping buffer page: 0 = Not done 1 = Done	r, u, c0
DNE2	1	Buffer done bit for pong buffer page: 0 = Not done 1 = Done	r, u, c0
ERR1	1	DMA error response detected for ping buffer page: 0 = No error 1 = Error	r, u, c0
ERR2	1	DMA error response detected for pong buffer page: 0 = No error 1 = Error	r, u, c0
PS1	1	Packet start bit for ping buffer page: 0 = No packet start 1 = Packet start Reserved for synchronous and isochronous channels.	r, w, u (both Tx and Rx)

(1)
- r - software readable
- w - software writable
- u - updated periodically by hardware
- c0 - software writes '0' to clear

Table 11-1709. Field Descriptions Of DMA Descriptor Table (continued)

Field	Numbers of bits	Description	Accessibilty [1]
PS2	1	Packet start bit for pong buffer page: 0 = No packet start 1 = Packet start Reserved for synchronous and isochronous channels.	r, w, u (both Tx and Rx)
MEP1	1	Most ethernet packet (MEP) indicator for ping buffer page: 0 = No MEP 1 = MEP MEP1 only valid for the first page of a segmented buffer. Reserved for control, synchronous and isochronous channels	r, w, u (both Tx and Rx)
MEP2	1	Most ethernet packet (MEP) indicator for pong buffer page: 0 = No MEP 1 = MEP MEP2 only valid for the first page of a segmented buffer. Reserved for control, synchronous and isochronous channels	r, w, u (both Tx and Rx)
BD1[2]	11 to 13	Buffer depth for ping buffer page: 11 or 12-bits for asynchronous and control channels. 13-bits for synchronous and isochronous channels.	r, w
BD2[2]	11 to 13	Buffer depth for pong buffer page: 11 or 12-bits for asynchronous and control channels. 13-bits for synchronous and isochronous channels.	r, w
BA1	32	Buffer base address for ping buffer page	r, w
BA2	32	Buffer base address for pong buffer page	r, w
Reserved	varies	Software writes a zero to all Reserved bits when the entry is initialized. The Reserved bits are Read-only after initialization.	r, w, u

(2) The buffer depth (BD1 and BD2) for synchronous channels must consider if multi-frame per sub-buffer mode (the MFE bit in the Channel Allocation Table) is enabled.

Data exchange across the DMA interface can be configured as little endian (LE = 1h) or big endian (LE = 0h). Figure 11-784 provides an overview of the endian options, chosen by the LE bit.

Figure 11-784. DMA Descriptor Table Endian Options

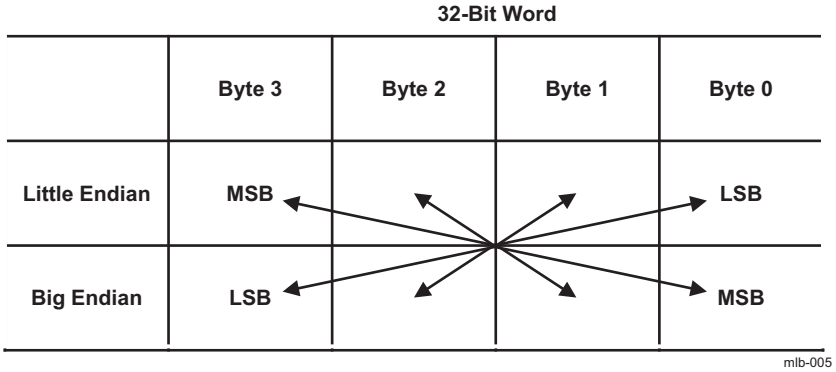
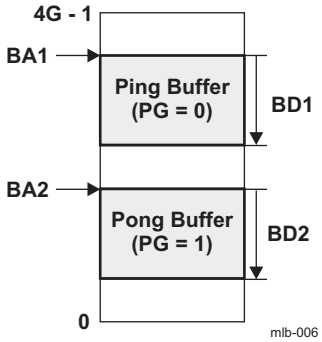


Figure 11-785 shows an example of ping-pong system memory structure. This system memory structure is similar for all channel types and shows the relationship between the BA1, BA2, BD1, BD2 and PG descriptor fields.

Figure 11-785. Ping-Pong System Memory Structure



Each DMA descriptor table entry (also referred to as a channel descriptor) holds a 32-bit BAn (n = 1 and 2) field which defines the start of each ping or pong buffer within system memory. The BDi (i = 1 and 2) field is used to indicate the size for the respective ping or pong page. The maximum size is 2k-entries for asynchronous and control channels, and 8k-entries for isochronous and synchronous channels.

11.11.4.1.5.1 Synchronous Channel Descriptor

The synchronous buffering scheme allows each ping or pong buffer to contain a single frame or a multiple number of frames. For this reason, the synchronous buffer depth (BDi) must be defined in terms of an integer number (n), frames per sub-buffer (m) and bytes per frame (bpf) of data (that is, BDi = n × m × bpf - 1).

Table 11-1710 shows the format for a synchronous DMA descriptor table entry. For more details about the field descriptions see Table 11-1709. Each synchronous channel buffer can be up to 8k-bytes deep.

NOTE: In synchronous transmit mode software should take care that the pong DDT is configured before ping DMA completion and ping DDT is configured before pong DMA completion. For more information about ping and pong DMA configuration see Table 11-1722 and Table 11-1723, respectively.

Table 11-1710. Format Of Synchronous DMA Descriptor Table Entry

Bit Offset	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	CE	LE	PG	Reserved												
16	Reserved															
32	RDY1	DNE1	ERR1	BD1[12-0]												
48	RDY2	DNE2	ERR2	BD2[12-0]												
64	BA1[15-0]															
80	BA1[31-16]															
96	BA2[15-0]															
112	BA2[31-16]															

11.11.4.1.5.2 Isochronous Channel Descriptors

The isochronous buffering scheme allows each ping or pong buffer to contain a single block or a multiple number of blocks. For this reason, the isochronous buffer depth (BDi) must be defined in terms of an integer number (n) and block size (BS) (that is, BDi = n × (BS + 1) - 1).

Table 11-1711 shows the format for an isochronous DMA descriptor table entry. For more details about the field descriptions see Table 11-1709. Each isochronous channel buffer can be up to 8k-bytes deep.

Table 11-1711. Format Of Isochronous DMA Descriptor Table Entry

Bit Offset	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	CE	LE	PG	Reserved												
16	Reserved															
32	RDY1	DNE1	ERR1	BD1[12-0]												
48	RDY2	DNE2	ERR2	BD2[12-0]												
64	BA1[15-0]															
80	BA1[31-16]															
96	BA2[15-0]															
112	BA2[31-16]															

11.11.4.1.5.3 Asynchronous and Control Channel Descriptors

Every asynchronous and control designates the first two bytes of each packet as the packet length. Each packet must be no more than 2048 bytes (packet length ≤ 2048).

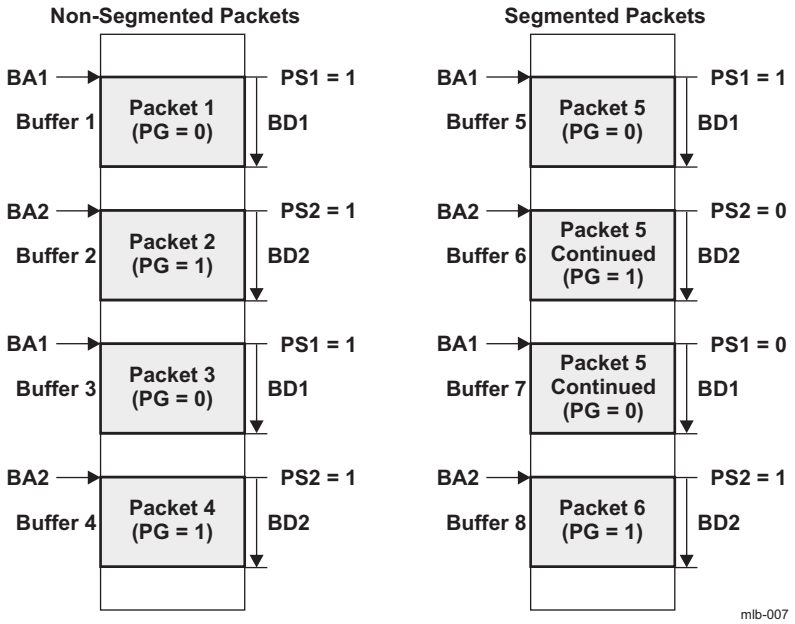
Software must set the buffer ready bit (RDYn) for each buffer as it programs the DMA. As hardware processes each buffer, it sets the done bit (DNEn) and generates an interrupt. When hardware finishes the buffer processing, it can begin processing another buffer if RDYn is set. The application is responsible for setting up and configuring the channel buffer descriptor prior to every DMA access on the channel.

Two-packet modes are supported by hardware for programming the DMA. These are single-packet mode and multiple-packet mode.

11.11.4.1.5.3.1 Single-Packet Mode

The single-packet mode asynchronous and control buffering scheme supports a maximum of one packet per buffer (that is, ping or pong). Both non-segmented and segmented data packets are allowed while using single-packet mode. Non-segmented packets are exchanged when only one buffer (that is ping or pong) is needed for packet transfer. Segmented packets are exchanged when a single packet is too long for one buffer and the packet must span multiple buffers. Figure 11-786 shows the memory space usage for both non-segmented and segmented asynchronous or control packets along with the packet start bit (PSn).

Figure 11-786. Single-Packet Mode Memory Space



While using single-packet mode, buffer done (DNEn) is set in hardware when a packet is done or the buffer is full.

Table 11-1712 shows the format for single-packet mode asynchronous and control DMA descriptor table entries. For more details about the field descriptions see Table 11-1709.

Table 11-1712. Format Of Single-Packet Asynchronous and Control DMA Descriptor Table Entry

Bit Offset	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	CE	LE	PG	Reserved												
16	Reserved															
32	RDY 1	DNE1	ERR 1	PS1	MEP1	BD1[10-0]										
48	RDY 2	DNE2	ERR 2	PS2	MEP2	BD2[10-0]										
64	BA1[15-0]															
80	BA1[31-16]															
96	BA2[15-0]															
112	BA2[31-16]															

11.11.4.1.5.3.2 Multiple-Packet Mode

The multiple-packet mode asynchronous and control buffering scheme supports more than one packet per system memory buffer, as shown in Figure 11-787. The multiple-packet mode reduces the interrupt rate for packet channels at the cost of increasing buffering and latency.

For Tx packet channels in multiple-packet mode, software sets the packet start bit (PSn) for every buffer. Setting PSn informs hardware that the first two bytes of the buffer contains the port message length (PML) of the first packet. After the first packet, hardware keeps track of where packets start and end within the current buffer. Software should not write to PSn while the buffer is active (RDYn = 1 and DNEn = 0). For Tx packet channels, the buffer is done (DNEn = 1) when the last byte of the last packet in the buffer is read from system memory. Software should set the buffer depth to contain the exact number of complete packets for that buffer. Segmented buffers are not supported for Tx packet channels in multiple-packet mode.

For Rx packet channels in multiple-packet mode, PSn has no meaning and should be ignored. Software is responsible for keeping track of where each packet starts and ends within the multiple-packet buffer via the packet PML. The buffer done bit (DNEn) is set in hardware for Rx channels when a buffer is full (see Buffer 1 in Figure 11-787) or if a packet ends exactly 1-byte before the end of the buffer (see Buffer 2 in Figure 11-787). Multiple-packet mode also supports segmented Rx packets spanning two or more buffers (see Buffers 3 – 6 in Figure 11-787).

Figure 11-787. Multi-Packet Mode System Memory

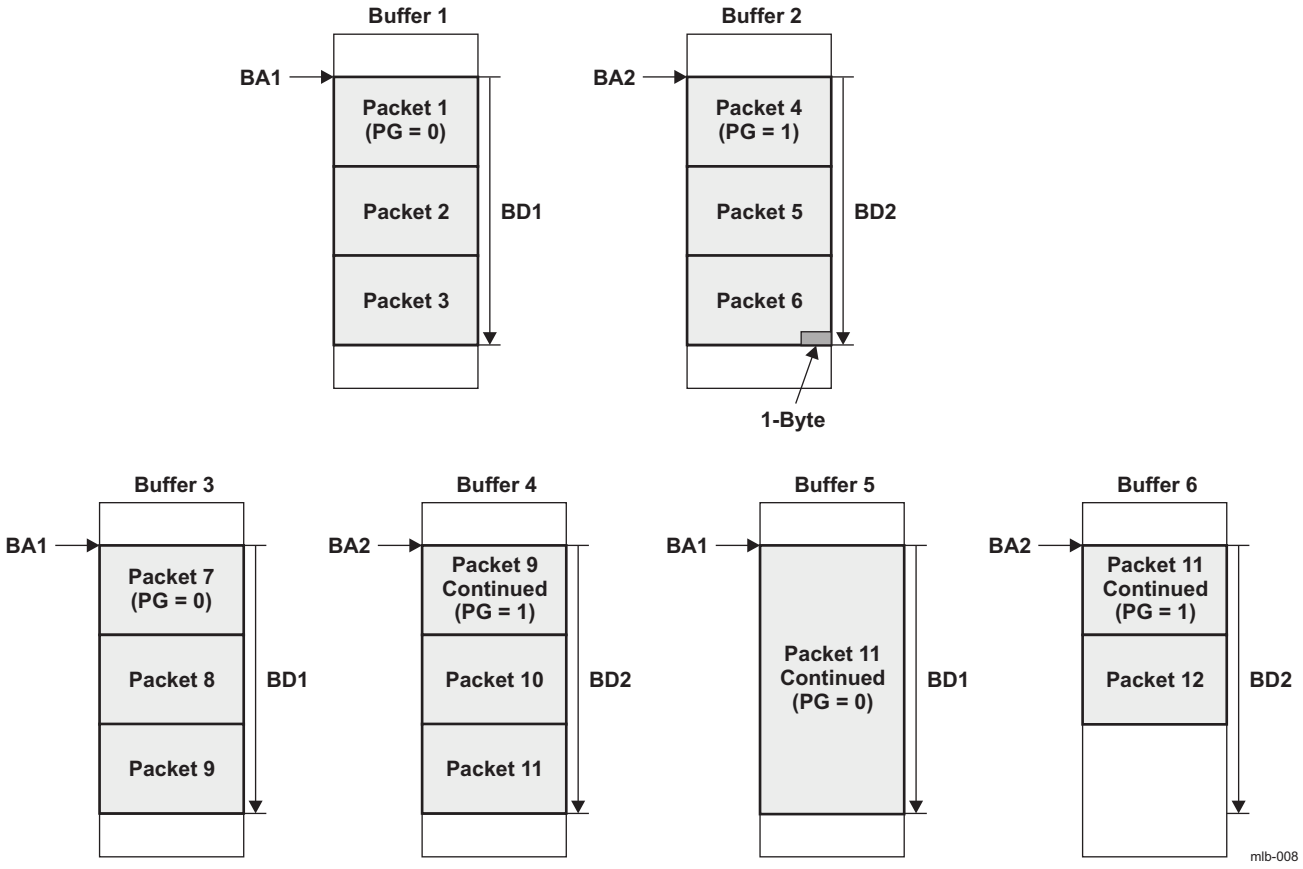


Table 11-1713. Format Of Multiple-Packet Asynchronous And Control DMA Descriptor Table Entry

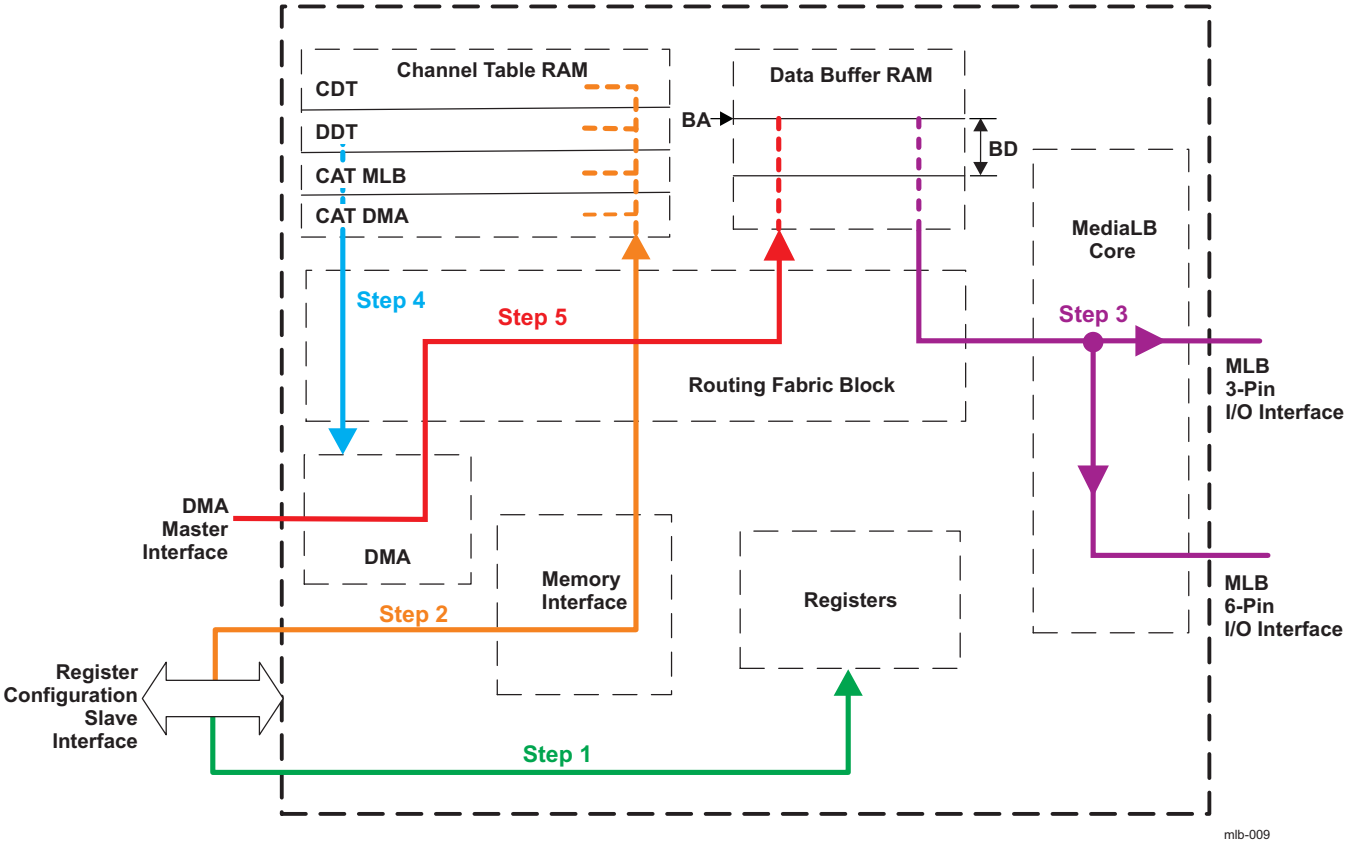
Bit Offset	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	CE	LE	PG	Reserved												
16	Reserved															
32	RDY1	DNE1	ERR1	PS1 (1)	BD1[11-0]											
48	RDY2	DNE2	ERR2	PS2 (1)	BD2[11-0]											
64	BA1[15-0]															
80	BA1[31-16]															
96	BA2[15-0]															
112	BA2[31-16]															

(1) PS1 and PS2 have no meaning and should be ignored in Rx mode.

11.11.4.2 Software and Data Flow for MLBSS

Figure 11-788 shows an overview of the MLBSS software and data flow when transmitting.

Figure 11-788. MLBSS Software and Data Tx Flow Overview

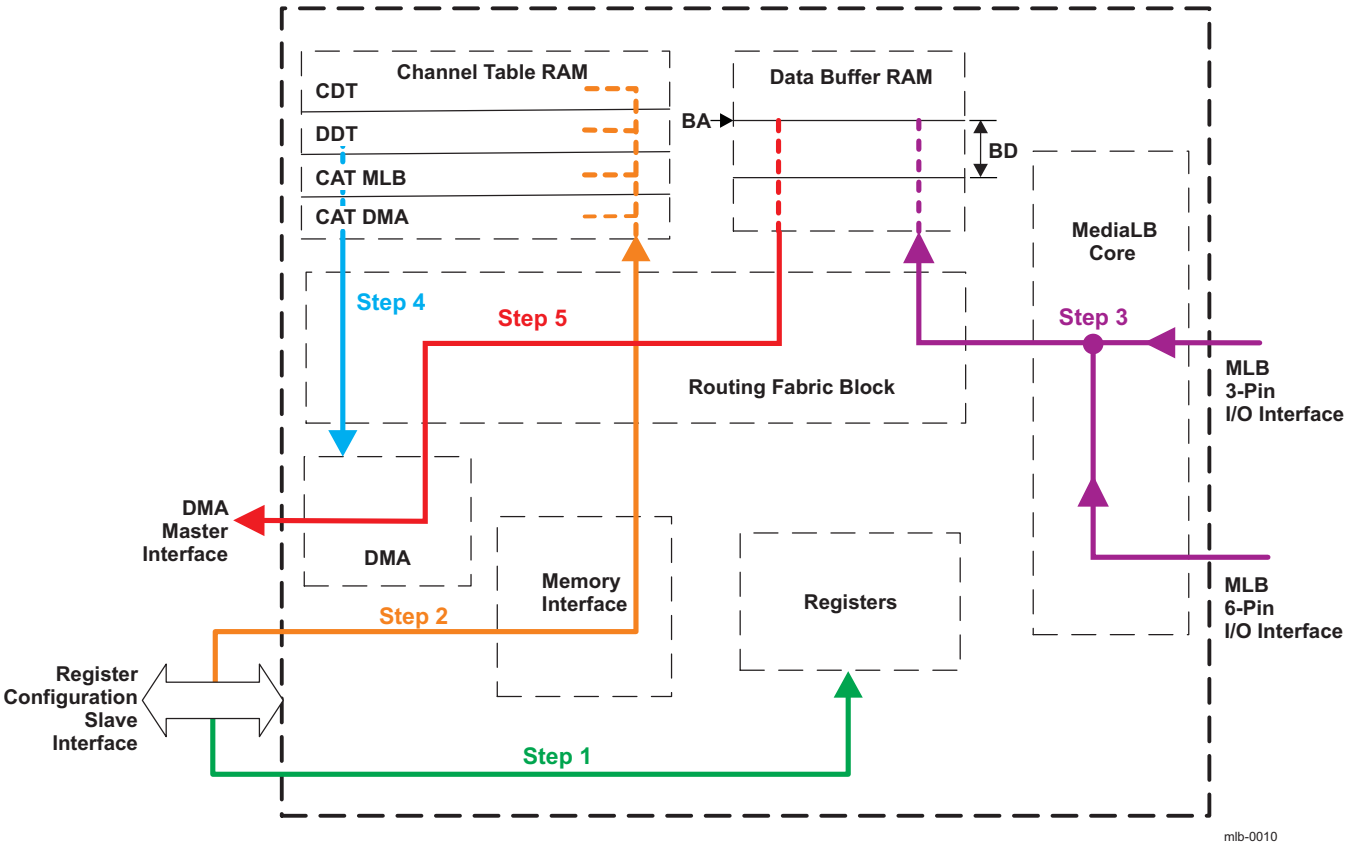


The MLBSS data Tx flow is defined in the following way:

1. Hardware is configured (see Table 11-1716).
2. Channel table RAM is cleared and configured (see Table 11-1717).
3. Data Tx begins through MediaLB 3-pin or 6-pin interface after MLBSS is synchronized to the incoming MLB framesync.
4. DMA reads from channel table RAM.
5. Data Tx is done through the DMA master interface.

Figure 11-789 shows an overview of the MLBSS software and data flow when receiving.

Figure 11-789. MLBSS Software and Data Rx Flow Overview



The MLBSS data Rx flow is defined in the following way:

1. Hardware is configured (see Table 11-1716).
2. Channel table RAM is cleared and configured (see Table 11-1717).
3. Data Rx begins through MediaLB 3-pin or 6-pin interface after MLBSS is synchronized to the incoming MLB framesync.
4. DMA reads from channel table RAM.
5. Data Rx is done through the DMA master interface.

Steps 1 and 2 from Figure 11-788 and Figure 11-789 are done by software. For more information see Section 11.11.5.1.2.1, *Channel Initialization*.

Steps 3-5 from Figure 11-788 and Figure 11-789 are handled by MLBSS internal hardware.

The top-level software tasks that the application must perform, can be placed in two categories:

- Channel initialization
- Channel servicing

For more information about channel initialization, see Section 11.11.5.1.2.1, *Channel Initialization*.

For more information about channel servicing, see Section 11.11.5.2.1, *Channel Servicing*.

11.11.4.2.1 Data Flow For Receive Channels

The data flow for receive channels is as follows:

1. After MLBSS is configured (see Section 11.11.5.1.2.1) and MediaLB has synchronized to the incoming MediaLB framesync (MLB_MLBC0[7] MLBLK = 1h), serial data is received either through the MLBP_DAT_P and MLBP_DAT_N pads or through the MLB_DAT pad, depending on the pin mode (6-pin or 3-pin) selected.

2. The MLBSS logic writes incoming data to the appropriate buffer in the data buffer RAM. The data buffer RAM address to which data is written is the BA field of the channel descriptor table entry. The channel descriptor table entry is specified by the CL field of the MediaLB channel allocation table entry for the channel. The MediaLB channel allocation table entry equals (1/2)*ChannelAddr.
3. The DMA access for a particular DMA channel (DMA channel allocation table entry) is started when there is sufficient data for the DMA channel in the data buffer RAM to write into system memory and the system memory buffer (ping or pong depending on the PG bit in DMA descriptor table entry) is marked as ready. Note that presence of a frame for synchronous/isochronous traffic (multiple frames in multi frame mode) or a packet for control/asynchronous traffic (multiple packets in multi packet mode) is deemed sufficient data to mark a DMA channel as ready.
4. The DMA hardware reads from the data buffer RAM memory address specified by the BA field of the channel descriptor table entry and writes to the system memory address specified in the DMA descriptor table. The CL field in the DMA channel allocation table entry is used to determine the appropriate channel descriptor table entry. Note that CL field of DMA channel allocation table and MLB channel allocation table entries must be the same, thus providing the common data buffer RAM address for MLB core to write into DMA and the DMA to read from the MLB core. Further, each DMA channel allocation table entry corresponds to a DMA descriptor table entry as per the following mapping:
   - Address of DMA descriptor table entry = 40h + Index of DMA channel allocation table entry.

11.11.4.2.2 Data Flow for Transmit Channels

The data flow for transmit channels is as follows:

1. The DMA access for a particular DMA channel (DMA channel allocation table entry) is started when data buffer RAM has sufficient buffer space and the system memory buffer (ping or pong depending on the PG bit in DMA descriptor table entry) is marked as ready for a particular DMA channel allocation table entry. Each DMA channel allocation table entry corresponds to a DMA descriptor table entry as per the following mapping:
   - Address of DMA descriptor table entry = 40h + Index of DMA channel allocation table entry.
2. The DMA reads from the system memory address specified in the DMA descriptor table and writes to the data buffer RAM memory address specified by the BA field of the channel descriptor table entry. The CL field in the DMA channel allocation table entry is used to determine the appropriate channel descriptor table entry. Note that CL field of DMA channel allocation table and MLB channel allocation table entries must be the same, thus providing the common data buffer RAM address for DMA to write into MLB core and the MLB core to read from the DMA.
3. Once the MLBSS is synchronized to the incoming MediaLB framesync (MLB_MLBC0[7] MLBLK = 1h) and a MediaLB channel address is received, the MediaLB core refers to the specific MediaLB channel allocation table entry and reads data from the data buffer RAM buffer. The data buffer RAM address from which data is read is the BA field of the channel descriptor table entry. The channel descriptor table entry is specified by the CL field of the MLB channel allocation table entry for the channel. The MediaLB channel allocation table entry equals (1/2)*ChannelAddr.
4. This data is serialized and transmitted through the MediaLB 3-pin or 6-pin interface.

11.11.4.3 MLB Priority On The System Interconnect

There is a register used for controlling the priority of the MLB subsystem on the system interconnect. This is done through the INITIATOR_PRIORITY0[6-4] MLB_PRI bit field which reside in the BOOT_CFG module. Setting this bit field to 0h means that the MLB traffic has highest priority over the other initiator traffics. A value of 7h is for lowest priority.

11.11.5 MLB Programming Guide

This section describes the programming sequences for the configuration and use of the MLBSS.

11.11.5.1 Global Initialization

11.11.5.1.1 Surrounding Modules Global Initialization

This section identifies the requirements for initializing the surrounding modules when the module is used for the first time after a device reset. This initialization of surrounding modules is based on the integration and environment of the MLB subsystem. For more information, see Section 11.11.3, *MLB Integration*, and Section 11.11.2, *MLB Environment*. Table 11-1714 shows the global initialization of MLBSS surrounding modules.

Table 11-1714. Global Initialization of Surrounding Modules

Surrounding Modules	Comments
PLL Controller	PLL Controller configuration must be done to enable the MLBSS clocks. See Section 5.4.5.3, *PLL Controller*.
ARM_GIC	Device INTCs must be configured to enable the interrupt request generation. For information about enabling ARM_GIC interrupts, see Section 6.1, *Arm Cortex-A15 Subsystem*.
CIC	Device INTCs must be configured to enable the interrupt request generation. For information about enabling CIC interrupts, see Chapter 9, *Interrupts*.
LPSC11	Module reset must be enabled. For more information about the module configuration, see Section 5.2, *Power Management*.
BOOT_CFG	The pad muxing must be configured in the BOOT_CFG. The MLBSS I/Os must be configured also in the BOOT_CFG. See Section 5.1.3.1.1, *Pad Configuration Registers* in Section 5.1, *Control Module (BOOT_CFG)*. See also section Multiplexing characteristics in the device Data Manual.

11.11.5.1.2 MLBSS Global Initialization

11.11.5.1.2.1 Channel Initialization

A channel initialization must be performed in order to ensure proper operation. Table 11-1715 shows the sequence.

Table 11-1715. Channel Initialization Steps

Step	Crossreference
Configuring the hardware	Table 11-1716
Programming the routing fabric block	Table 11-1717
Programming the DMA block	Table 11-1721
Synchronizing and unmuting the synchronous channel	Table 11-1724

Configuring The Hardware

To configure the MediaLB interface, the steps shown in Table 11-1716 should be followed.

Table 11-1716. Configuring The Hardware

Step	Register/ Bit Field/ Programming Model/ Comments	Value
Clear channel table RAM, to ensure that no stray interrupts are generated	-	0h
Clear the interrupt status bits for logical channels 0 to 63	MLB_ACSR0 and MLB_ACSR1	FFFF FFFFh
Mask all interrupts for DMA channels (from 0 to 63)	MLB_ACMR0 and MLB_ACMR1	0000 0000h

Table 11-1716. Configuring The Hardware (continued)

Step	Register/ Bit Field/ Programming Model/ Comments	Value
Select 3-pin or 6-pin MediaLB operation	MLB_MLBC0[5] MLBPEN	0h for 3-pin 1h for 6-pin
Select MediaLB clock speed	MLB_MLBC0[4-2] MLBCLK	-h
Enable the MediaLB	MLB_MLBC0[0] MLBEN	1h
Activate (unmask) all channels	MLB_HCMR0 and MLB_HCMR1	FFFF FFFFh
Enable the HBI block	MLB_HCTL[15] EN	1h
Set the DMA channel mask bits according to all active DMA channels	MLB_HCMR0 and MLB_HCMR1	-h
Select DMA Mode 1 to be used	MLB_ACTL[2] DMA_MODE	1h
Select software to clear the interrupts	MLB_ACTL[0] SCE	1h
Enable MLB interrupts, if required	MLB_MIEN	-h

Programming The Routing Fabric Block

The channel allocation table and channel descriptor table reside in the channel table RAM and are programmed indirectly through the memory interface block. The steps to be followed are shown in Table 11-1717.

Table 11-1717. Programming The Routing Fabric Block

Step	Register/ Bit Field/ Programming Model/ Comments	Value
1. Initialize all bits of the channel allocation table	-	0h
2. Select a connection label	CL	N = 0h to 3Fh
3. Program the channel descriptor table for channel N	Use the memory interface registers to do a write to the channel descriptor table. See Table 11-1718.	Mask to be used: FFFF FFFFh
4. Program the channel allocation table for inbound data buffer RAM access	Write to the data buffer RAM. See Table 11-1719.	-
5. Program the channel allocation table for outbound data buffer RAM access	Read from the data buffer RAM. See Table 11-1720.	-
6. Repeat steps 2-5 to initialize all logical channels	-	-

Table 11-1718. Subsequence - Program The Channel Descriptor Table

Step	Register/ Bit Field/ Programming Model/ Comments	Value
1. Set the 14-bit base address	BA	-h
2. Set the 12-bit or 13-bit buffer depth	BD (set this according to the BA so that it doesn't overflow the buffer)	BD = buffer depth in bytes - 1
IF: Synchronous channels		
Set the 12-bit or 13-bit buffer depth	BD	(BD + 1) = 4 × frames per sub-buffer (m) × bytes per frame (bpf)
ENDIF		
IF: Isochronous channels		
Set the 12-bit or 13-bit buffer depth	BD	(BD + 1) mod (BS + 1) = 0
ENDIF		
IF: Asynchronous channels		
Set the 12-bit or 13-bit buffer depth	BD	(BD + 1) >= max packet length (1024 for a MOST Data Packet (MDP))
ENDIF		

Table 11-1718. Subsequence - Program The Channel Descriptor Table (continued)

Step	Register/ Bit Field/ Programming Model/ Comments	Value
IF: Control channels		
Set the 12-bit or 13-bit buffer depth	BD	(BD + 1) >= max packet length (64)
ENDIF		
3. For isochronous channels, configure the block size	BS	BS = block size in bytes - 1
4. Write to all other bits of the channel descriptor table		0h

Table 11-1719. Subsequence - Write To The Data Buffer RAM

Step	Register/ Bit Field/ Programming Model/ Comments	Value
For Tx channels the DMA block does the inbound data buffer RAM access	A Tx channel is from MLBSS to external host	-
For Rx channels MediaLB does the inbound data buffer RAM access	An Rx channel is from external host to MLBSS	-
Set the channel direction	RNW	0h
Set the channel type	CT[2-0]	0h: synchronous 1h: control 2h: asynchronous 3h: isochronous
Set the connection label	CL[5-0]	CL[5-0] = N
IF: CT[2-0] = 0h (synchronous)		
Set the mute bit.	MT (the mute bit is set to avoid any flow of the garbage data that will be there in the data buffer prior to any transfer)	1h
ENDIF		
Set the channel enable bit	CE	1h
Clear all other bits of the channel allocation table	-	0h

Table 11-1720. Subsequence - Read From The Data Buffer RAM

Step	Register/ Bit Field/ Programming Model/ Comments	Value
For Tx channels MediaLB does the outbound data buffer RAM access	A Tx channel is from MLBSS to external host	-
For Rx channels the DMA block does the outbound data buffer RAM access	An Rx channel is from external host to MLBSS	-
Set the channel direction	RNW	1h
Set the channel type	CT[2-0]	0h: synchronous 1h: control 2h: asynchronous 3h: isochronous
Set the connection label	CL[5-0]	CL[5-0] = N
IF: CT[2-0] = 000 (synchronous),		
Set the mute bit.	MT	1h
ENDIF		
Set the channel enable bit	CE	1h
Clear all other bits of the channel allocation table	-	0h

Programming The DMA Block

The DMA descriptor table resides in the channel table RAM and is programmed through TeraNet_CFG slave interface using the registers of the memory interface block. The steps to be followed are shown in Table 11-1721.

Table 11-1721. Programming The DMA Block

Step [1]	Register/ Bit Field/ Programming Model/ Comments	Value
1. Initialize all bits of the DMA descriptor table	-	0h
2. Select a connection label	CL (Note that the CL number chosen in this step should be equal to the CL number chosen in step 2 of Table 11-1717)	N = 0h to 3Fh
3. Program the DMA block ping page for channel N [2]	See Table 11-1722	-
4. Program the DMA block pong page for channel N	See Table 11-1723	-
5. Select big or little endian	LE	-h
6. Select the active page	PG (Program this bit only once in the beginning of the configuration. It shouldn't be touched by the software after that. The hardware updates this bit as the transactions happen)	-h
7. Set the channel enable bit for all active logical channels	CE	1h
8. Repeat steps 2-7 for all active logical channels	-	-

(1) **Tip:** In order to program the MLB and DMA channel allocation table entries with respect to corresponding MLB and DMA channel address, the following steps are recommended:
- Given the MLB channel address is X (1-63), the MLB channel allocation table entry to be programmed should be X/2.
- Given the DMA channel address is X (0-63), the DMA channel allocation table entry to be programmed is X.

(2) For programming the 128-bit DMA descriptor table entry,
- IF: The DMA descriptor table is initialized with zeros and the PG bit is intended to be set to 0h, the following mask should only be used: MLB_MDWE0 = 000C 0000h
- ELSE: The following steps should be performed:
  - The mask to be used: = 000E 0000h.
  - Program the DMA descriptor table and the PG bit.
  - Change the MLB_MDWE0 mask value to 000C 0000h after the first configuration has been done to prevent any further writes by host to the PG bit during ongoing MLB transactions.
  - Write the FFFF FFFFh mask to the MLB_MDWE1, MLB_MDWE2 and MLB_MDWE3 registers

Table 11-1722. Subsequence - Program The DMA Block Ping Page

Step	Register/ Bit Field/ Programming Model/ Comments	Value
Set the 32-bit base address	BA1	-h
Set the 11-bit buffer depth	BD1	BD1 = buffer depth in bytes - 1
IF: Synchronous channels		
Set the 11-bit buffer depth	BD1	(BD1+ 1) = n × frames per sub-buffer (m) × bytes-per-frame (bpf)
ENDIF		
IF: Isochronous channels		
Set the 11-bit buffer depth	BD1	(BD1 + 1) mod (BS + 1) = 0
ENDIF		
IF: Asynchronous and control Rx channels		
Set the 11-bit buffer depth	BD1	5 <= (BD1 + 1) <= 4096 [1]
ENDIF		

(1) 4096 is the max packet length.

Table 11-1722. Subsequence - Program The DMA Block Ping Page (continued)

Step	Register/ Bit Field/ Programming Model/ Comments	Value
IF: Asynchronous and control Tx channels		
Set the 11-bit buffer depth	BD1	5 <= (BD1 + 1) <= 4096 (1)
Set the packet start bit if the page contains the start of the packet	PS1	1h
ENDIF		
Clear the page done bit	DNE1	0h
Clear the error bit	ERR1	0h
Set the page ready bit	RDY1	1h

Table 11-1723. Subsequence - Program The DMA Block Pong Page

Step	Register/ Bit Field/ Programming Model/ Comments	Value
Set the 32-bit base address	BA2	-h
Set the 11-bit buffer depth	BD2	BD2 = buffer depth in bytes - 1
IF: Synchronous channels		
Set the 11-bit buffer depth	BD2	(BD2+ 1) = n × frames per sub-buffer (m) × bytes-per-frame (bpf)
ENDIF		
IF: Isochronous channels		
Set the 11-bit buffer depth	BD2	(BD2 + 1) mod (BS + 1) = 0
ENDIF		
IF: Asynchronous and control Rx channels		
Set the 11-bit buffer depth	BD2	5 <= (BD2 + 1) <= 4096 (1)
ENDIF		
IF: Asynchronous and control Tx channels		
Set the 11-bit buffer depth	BD2	5 <= (BD2 + 1) <= 4096 (1)
Set the packet start bit if the page contains the start of the packet	PS2	1h
ENDIF		
Clear the page done bit	DNE2	0h
Clear the error bit	ERR2	0h
Set the page ready	RDY2	1h

(1) 4096 is the max packet length.

Synchronizing And Unmuting The Synchronous Channel

The steps to be followed are shown in Table 11-1724

Table 11-1724. Synchronizing And Unmuting The Synchronous Channel

Step	Register/ Bit Field/ Programming Model/ Comments	Value
Check that MediaLB clock is running	MLB_MLBC1[7] CLKMERR	0h
IF: The MediaLB clock is not toggling at the pads	MLB_MLBC1[7] CLKMERR	1h
Clear the register bit	MLB_MLBC1[7] CLKMERR	
Wait one MLB_SPB_OCP_CLK or MLB I/O clock cycle		

Table 11-1724. Synchronizing And Unmuting The Synchronous Channel (continued)

Step	Register/ Bit Field/ Programming Model/ Comments	Value
Check that MediaLB clock is running	MLB_MLBC1[7] CLKMERR	0h
ENDIF		
Poll for MediaLB lock	MLB_MLBC0[7] MLBLK	1h
Wait four frames	-	-
Unmute the synchronous channel/channels	MT	0h

11.11.5.2 MLBSS Operational Modes Configuration

11.11.5.2.1 Channel Servicing

After initialization, each channel requires periodic servicing. Table 11-1725 shows the steps which can be performed concurrently and in any order.

Table 11-1725. Channel Servicing Steps

Step	Crossreference
Servicing the DMA Interrupts	Table 11-1726
Servicing MLB Interrupts	Table 11-1728
Polling for MediaLB System Commands	Table 11-1729

Servicing the DMA Interrupts

The steps to be followed are shown in Table 11-1726

Table 11-1726. Servicing the DMA Interrupts

Step	Register/ Bit Field/ Programming Model/ Comments	Value
1. Enable (unmask) the interrupts from all active DMA channels	MLB_ACMR0 and MLB_ACMR1	1h for each bit associated with an active DMA channel
2. Select the status clear method	MLB_ACTL[0] SCE	-h
3. Select one (MLB_DMA_CH_INT0 only) or two (MLB_DMA_CH_INT0 and MLB_DMA_CH_INT1) interrupt signals	MLB_ACTL[1] SMX	-h
4. Wait for an interrupt from MLB_DMA_CH_INT0 and/or MLB_DMA_CH_INT1 line	-	-
5. Determine which channel or channels are causing an interrupt	Read the MLB_ACSR0 and MLB_ACSR1 registers	-h
6. IF: MLB_ACTL[0] SCE = 1h (software clears the interrupts)		
Write the results of step 5 back to the MLB_ACSR0 and MLB_ACSR1 registers to clear the interrupt	MLB_ACSR0 and MLB_ACSR1	MLB_ACSR0 and MLB_ACSR1
ENDIF		
7. Select a logical channel (N = 0-63) with an interrupt to service	-	-
8. Read the DMA descriptor table entry for channel N	See Table 11-1727	-
9. Reprogram the expired or broken DMA page/pages via steps 3 and 4 of Table 11-1721	-	-
10. Repeat steps 6-9 for all channels with pending interrupts	-	-

Table 11-1726. Servicing the DMA Interrupts (continued)

Step	Register/ Bit Field/ Programming Model/ Comments	Value
11. Repeat steps 4-10 while there are active channels	-	-

NOTE: The MLB_ACSR0, MLB_ACSR1, MLB_ACMR0 and MLB_ACMR1 register bits correspond to the DMA channel address for a particular logical channel. This should not be confused with the MLB channel address.

Channels that receive a DMA error response are disabled (CE = 0) by hardware.

Table 11-1727. Subsequence - Reading The DMA Descriptor Table Entry

Step	Register/ Bit Field/ Programming Model/ Comments	Value
Determine the active page (ping or pong)	Use the PG bit	-h
Determine which page/pages are done	Use the DNE1 and DNE2 bits	1h
Determine which channels encountered a DMA error	Use the ERR1 and ERR2 bits	1h
Determine which asynchronous and control Rx channel pages contain a packet start	Use the PS1 and PS2 bits	1h

Servicing MLB Interrupts

To service the MLB interrupts, the steps desctibed in Table 11-1728 should be followed.

Table 11-1728. Servicing MLB Interrupts

Step	Register/ Bit Field/ Programming Model/ Comments	Value
Clear the appropriate MediaLB channel status bits	MLB_MS0 and MLB_MS1 registers	0h
Enable protocol error interrupts for all active MediaLB channels	MLB_MIEN[28] CTX_PE MLB_MIEN[25] CRX_PE MLB_MIEN[21] ATX_PE MLB_MIEN[18] ARX_PE MLB_MIEN[16] SYNC_PE MLB_MIEN[0] ISOC_PE	1h 1h 1h 1h 1h 1h
Wait for an interrupt on the MLB_SYS_INT line		
Determine which channel/channels are causing the interrupt	Read the MLB_MS0 and MLB_MS1 registers	1h
Determine the interrupt type	Read the RSTS and WSTS bits of the appropriate channel descriptor tables	-h
IF: Synchronous channels		
Clear WSTS errors to resume channel operation	WSTS[3]	0h
ENDIF		
IF: Isochronous channels		
Clear WSTS errors to resume channel operation	WSTS[2-1]	0h
ENDIF		
IF: Asynchronous and control channels		

Table 11-1728. Servicing MLB Interrupts (continued)

Step	Register/ Bit Field/ Programming Model/ Comments	Value
Clear RSTS and WSTS errors to resume channel operation	RSTS[4], WSTS[4], RSTS[2] and WSTS[2]	0h
ENDIF		

Polling For MediaLB System Commands

The MediaLB core supports the MediaLB system commands (MlbScan, MlbReset, MOST_Unlock). The MLB_MSS register is used to detect a system command received from the MediaLB controller. The MLB automatically sends the appropriate system response to the MediaLB Controller. Software should follow the procedure shown in Table 11-1729.

Table 11-1729. Polling For MediaLB System Commands

Step	Register/ Bit Field/ Programming Model/ Comments	Value
Poll periodically for a system command received from the MediaLB controller	MLB_MSS	-h
Clear the appropriate bits in MLB_MSS register after the application finishes the servicing	MLB_MSS	0h
IF: Software system command is detected	MLB_MSS[4] SWSYSCMD	1h
Read the MLB_MSD register to receive the system data sent from MediaLB Controller	MLB_MSD	-h
ENDIF		

11.11.5.2.2 Channel Table RAM Access

The memory interface block allows a direct software access to the channel table RAM. Any write to the MLB_MADR register triggers a single read or write cycle. Reading from the MLB_MADR register does not initiate a read or write access.

Table 11-1730 shows a direct write to the channel table RAM.

Table 11-1730. Direct Channel Table RAM Writes

Step	Register/ Bit Field/ Programming Model/ Comments	Value
Load the 128-bit data entry	MLB_MDAT0, MLB_MDAT1, MLB_MDAT2 and MLB_MDAT3 registers	-h
Enable writing data	MLB_MDWE0, MLB_MDWE1, MLB_MDWE2 and MLB_MDWE3 registers	1h
Write the 8-bit channel table RAM target address	MLB_MADR[7-0] ADDR_7_0	-h
Initiate a write cycle	MLB_MADR[31] WNR	1h
Determine when the transfer is complete	Poll the MLB_MCTL[0] XCMP	1h

Table 11-1731 shows a direct read from the channel table RAM.

Table 11-1731. Direct Channel Table RAM Reads

Step	Register/ Bit Field/ Programming Model/ Comments	Value
Write the 8-bit channel table RAM target address	MLB_MADR[7-0] ADDR_7_0	-h

Table 11-1731. Direct Channel Table RAM Reads (continued)

Step	Register/ Bit Field/ Programming Model/ Comments	Value
Initiate a read cycle	MLB_MADR[31] WNR	0h
Determine when the transfer is complete	Poll the MLB_MCTL[0] XCMP	1h
Read the 128-bit data entry	MLB_MDAT0, MLB_MDAT1, MLB_MDAT2 and MLB_MDAT3 registers	-h

11.11.6 MLB Registers

Table 11-1733 lists the memory-mapped registers for the MLB. All register offset addresses not listed in Table 11-1733 should be considered as reserved locations and the register contents should not be modified.

This section describes the MLB instances registers.

Table 11-1732. MLB Instances

Instance	Base Address
MLB	021C 6000h

Table 11-1733. MLB Registers

Offset	Acronym	Register Name	MLB Physical Address	Section
400h	MLB_MLBC0	MediaLB Control 0 Register	021C 6400h	Section 11.11.6.1
408h	RESERVED		021C 6408h	
40Ch	MLB_MS0	MediaLB Channel Status 0 Register	021C 640Ch	Section 11.11.6.2
414h	MLB_MS1	MediaLB Channel Status 1 Register	021C 6414h	Section 11.11.6.3
420h	MLB_MSS	MediaLB System Status Register	021C 6420h	Section 11.11.6.4
424h	MLB_MSD	MediaLB System Data Register	021C 6424h	Section 11.11.6.5
42Ch	MLB_MIEN	MediaLB Interrupt Enable Register	021C 642Ch	Section 11.11.6.6
434h	RESERVED		021C 6434h	
438h	RESERVED		021C 6438h	
43Ch	MLB_MLBC1	MediaLB Control 1 Register	021C 643Ch	Section 11.11.6.7
480h	MLB_HCTL	HBI Control Register	021C 6480h	Section 11.11.6.8
488h	MLB_HCMR0	HBI Channel Mask 0 Register	021C 6488h	Section 11.11.6.9
48Ch	MLB_HCMR1	HBI Channel Mask 1 Register	021C 648Ch	Section 11.11.6.10
490h	MLB_HCER0	HBI Channel Error 0 Register	021C 6490h	Section 11.11.6.11
494h	MLB_HCER1	HBI Channel Error 1 Register	021C 6494h	Section 11.11.6.12
498h	MLB_HCBR0	HBI Channel Busy 0 Register	021C 6498h	Section 11.11.6.13
49Ch	MLB_HCBR1	HBI Channel Busy 1 Register	021C 649Ch	Section 11.11.6.14
4C0h	MLB_MDAT0	MIF Data 0 Register	021C 64C0h	Section 11.11.6.15
4C4h	MLB_MDAT1	MIF Data 1 Register	021C 64C4h	Section 11.11.6.16
4C8h	MLB_MDAT2	MIF Data 2 Register	021C 64C8h	Section 11.11.6.17
4CCh	MLB_MDAT3	MIF Data 3 Register	021C 64CCh	Section 11.11.6.18
4D0h	MLB_MDWE0	MIF Data Write Enable 0 Register	021C 64D0h	Section 11.11.6.19
4D4h	MLB_MDWE1	MIF Data Write Enable 1 Register	021C 64D4h	Section 11.11.6.20
4D8h	MLB_MDWE2	MIF Data Write Enable 2 Register	021C 64D8h	Section 11.11.6.21
4DCh	MLB_MDWE3	MIF Data Write Enable 3 Register	021C 64DCh	Section 11.11.6.22
4E0h	MLB_MCTL	MIF Control Register	021C 64E0h	Section 11.11.6.23
4E4h	MLB_MADR	MIF Address Register	021C 64E4h	Section 11.11.6.24
7C0h	MLB_ACTL	AHB Control Register	021C 67C0h	Section 11.11.6.25
7D0h	MLB_ACSR0	AHB Channel Status 0 Register	021C 67D0h	Section 11.11.6.26
7D4h	MLB_ACSR1	AHB Channel Status 1 Register	021C 67D4h	Section 11.11.6.27
7D8h	MLB_ACMR0	AHB Channel Mask 0 Register	021C 67D8h	Section 11.11.6.28
7DCh	MLB_ACMR1	AHB Channel Mask 1 Register	021C 67DCh	Section 11.11.6.29

11.11.6.1 MLB_MLBC0 Register (Offset = 400h) [reset = 0h]

MLB_MLBC0 is shown in Figure 11-790 and described in Table 11-1735.

MediaLB Control 0 Register.

Table 11-1734. MLB_MLBC0 Instances

Instance	Physical Address
MLB	021C 6400h

Figure 11-790. MLB_MLBC0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						FCNT	
R-0h						R/W-0h	
15	14	13	12	11	10	9	8
FCNT	CTLRETRY	RESERVED	ASYRETRY	RESERVED			
R/W-0h	R/W-0h	R-0h	R/W-0h	R-0h			
7	6	5	4	3	2	1	0
MLBLK	RESERVED	MLBPEN	MLBCLK			RESERVED	MLBEN
R-0h	R-0h	R/W-0h	R/W-0h			R-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1735. MLB_MLBC0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17-15	FCNT	R/W	0h	Number of frames per sub-buffer (synchronous channels). 000 - 1 frame per sub-buffer (Operation is the same as Standard mode) 001 - 2 frames per sub-buffer 010 - 4 frames per sub-buffer 011 - 8 frames per sub-buffer 100 - 16 frames per sub-buffer 101 - 32 frames per sub-buffer 110 - 64 frames per sub-buffer 111 - Reserved
14	CTLRETRY	R/W	0h	Control Tx packet retry. When set, a control packet that is flagged with a Break or ProtocolError by the receiver is retransmitted. When cleared, a control packet that is flagged with a Break or ProtocolError by the receiver is skipped.
13	RESERVED	R	0h	Reserved
12	ASYRETRY	R/W	0h	Asynchronous Tx packet retry. Asynchronous Tx packet retry. When set, an asynchronous packet that is flagged with a Break or ProtocolError by the receiver is retransmitted. When cleared, an asynchronous packet that is flagged with a Break or ProtocolError by the receiver is skipped.
11-8	RESERVED	R	0h	Reserved

Table 11-1735. MLB_MLBC0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	MLBLK	R	0h	MediaLB lock status. When set, indicates that the MediaLB block is synchronized to the incoming MediaLB frame. If MLBLK is clear (unlocked), MLBLK is set after FRAMESYNC is detected at the same position for three consecutive frames. If MLBLK is set (locked), MLBLK is cleared after not receiving FRAMESYNC at the expected time for two consecutive frames. While MLBLK is set, FRAMESYNC patterns occurring at locations other than the expected one are ignored. (read-only)
6	RESERVED	R	0h	Reserved
5	MLBPEN	R/W	0h	MediaLB 6-pin enable. 0 - MediaLB 3-pin interface enabled 1 - MediaLB 6-pin interface enabled
4-2	MLBCLK	R/W	0h	MediaLB clock speed select. 000 - 256 Fs (for MLBPEN = 0) 001 - 512 Fs (for MLBPEN = 0) 010 - 1024 Fs (for MLBPEN = 0) 011 - 2048 Fs (for MLBPEN = 1) 100 - 3072 Fs (for MLBPEN = 1) 101 - 4096 Fs (for MLBPEN = 1) 110 - 6144 Fs (for MLBPEN = 1) 111 - 8192 Fs (for MLBPEN = 1)
1	RESERVED	R	0h	Reserved
0	MLBEN	R/W	0h	MediaLB enable. When set, MediaLB clock, signal, and data are received and transmitted on the appropriate MediaLB pins.

Table 11-1736. Register Call Summary for MLB_MLBC0

MLB Programming Guide • Channel Initialization: [0][1][2][3]
MLB Registers • MLB Registers: [0] • MLB_MLBC0 Register (Offset = 0h) [reset = 0h]: [0]
MLB Functional Description • MediaLB Core Block: [0] • Data Flow For Receive Channels: [0] • Channel Descriptor Table: [0][1] • Data Flow for Transmit Channels: [0]

11.11.6.2 MLB_MS0 Register (Offset = 40Ch) [reset = 0h]

MLB_MS0 is shown in Figure 11-791 and described in Table 11-1738.

MediaLB Channel Status 0 Register.

Table 11-1737. MLB_MS0 Instances

Instance	Physical Address
MLB	021C 640Ch

Figure 11-791. MLB_MS0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCS_31_0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1738. MLB_MS0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MCS_31_0	R/W	0h	MediaLB channel status (for channels 31 to 0). Channel status bits are set by hardware and cleared by software. Status is only set if the appropriate bits in the MLB_MIEN register are set. Writing a 1 has no effect.

Table 11-1739. Register Call Summary for MLB_MS0

MLB Programming Guide • Channel Servicing: [0][1]
MLB Registers • MLB Registers: [0] • MLB_MS0 Register (Offset = Ch) [reset = 0h]: [0] • MLB_MIEN Register (Offset = 2Ch) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14]

11.11.6.3 MLB_MS1 Register (Offset = 414h) [reset = 0h]

MLB_MS1 is shown in Figure 11-792 and described in Table 11-1741.

MediaLB Channel Status 1 Register.

Table 11-1740. MLB_MS1 Instances

Instance	Physical Address
MLB	021C 6414h

Figure 11-792. MLB_MS1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCS_63_32																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1741. MLB_MS1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MCS_63_32	R/W	0h	MediaLB channel status (for channels 63 to 32). Indicates the channel status for MediaLB channels 63 to 32. Channel status bits are set by hardware and cleared by software. Status is only set if the appropriate bits in the MLB_MIEN register are set. Writing a 1 has no effect.

Table 11-1742. Register Call Summary for MLB_MS1

MLB Programming Guide • Channel Servicing: [0][1]
MLB Registers • MLB Registers: [0] • MLB_MS1 Register (Offset = 14h) [reset = 0h]: [0] • MLB_MIEN Register (Offset = 2Ch) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14]

11.11.6.4 MLB_MSS Register (Offset = 420h) [reset = 0h]

MLB_MSS is shown in Figure 11-793 and described in Table 11-1744.

MediaLB System Status Register.

Table 11-1743. MLB_MSS Instances

Instance	Physical Address
MLB	021C 6420h

Figure 11-793. MLB_MSS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		SERVREQ	SWSYSCMD	CSSYSCMD	ULKSYSCMD	LKSYSCMD	RSTSYSCMD
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1744. MLB_MSS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	SERVREQ	R/W	0h	Service request enabled. When set, the MediaLB block responds with a “device present, request service” system response if a matching channel scan system command is detected. When cleared, the MediaLB block responds with a “device present” system response.
4	SWSYSCMD	R/W	0h	Software system command detected (in the system quadlet). Set by hardware, cleared by software. Data is stored in the MLB_MSD register for this command. Writing a 1 has no effect.
3	CSSYSCMD	R/W	0h	Channel scan system command detected (in the system quadlet). Set by hardware, cleared by software. If the node address specified in Data quadlet matches the value in MLB_MLBC1[15-8] NDA, the device responds either “device present” or “device present, request service” system response in the next system quadlet. Writing a 1 has no effect.
2	ULKSYSCMD	R/W	0h	Network unlock system command detected (in the system quadlet). Set by hardware, cleared by software. Writing a 1 has no effect.
1	LKSYSCMD	R/W	0h	Network lock system command detected (in the system quadlet). Set by hardware, cleared by software. Writing a 1 has no effect.
0	RSTSYSCMD	R/W	0h	Reset system command detected (in the system quadlet). Set by hardware, cleared by software. Writing a 1 has no effect.

Table 11-1745. Register Call Summary for MLB_MSS

MLB Programming Guide • Channel Servicing: [0][1][2][3][4]
MLB Registers • MLB Registers: [0] • MLB_MSS Register (Offset = 20h) [reset = 0h]: [0] • MLB_MSD Register (Offset = 24h) [reset = 0h]: [0][1][2][3]

11.11.6.5 MLB_MSD Register (Offset = 424h) [reset = 0h]

MLB_MSD is shown in Figure 11-794 and described in Table 11-1747.

MediaLB System Data Register.

Table 11-1746. MLB_MSD Instances

Instance	Physical Address
MLB	021C 6424h

Figure 11-794. MLB_MSD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD3								SD2								SD1								SD0							
R-0h								R-0h								R-0h								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1747. MLB_MSD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	SD3	R	0h	System data (byte 3). Updated with MediaLB Data[31-24] when a MediaLB software system command is received in the system quadlet. If MLB_MSS[4] SWSYSCMD is already set, then SD3 is not updated. (read-only).
23-16	SD2	R	0h	System data (byte 2). Updated with MediaLB Data[23:16] when a MediaLB software system command is received in the system quadlet. If MLB_MSS[4] SWSYSCMD is already set, then SD2 is not updated. (readonly)
15-8	SD1	R	0h	System data (byte 1). Updated with MediaLB Data[15:8] when a MediaLB software system command is received in the system quadlet. If MLB_MSS[4] SWSYSCMD is already set, then SD1 is not updated. (read-only)
7-0	SD0	R	0h	System data (byte 0). Updated with MediaLB Data[7:0] when a MediaLB software system command is received in the system quadlet. If MLB_MSS[4] SWSYSCMD is already set, then SD0 is not updated. (read-only)

Table 11-1748. Register Call Summary for MLB_MSD

MLB Programming Guide • Channel Servicing: [0][1]
MLB Registers • MLB Registers: [0] • MLB_MSS Register (Offset = 20h) [reset = 0h]: [0] • MLB_MSD Register (Offset = 24h) [reset = 0h]: [0]

11.11.6.6 MLB_MIEN Register (Offset = 42Ch) [reset = 0h]

MLB_MIEN is shown in Figure 11-795 and described in Table 11-1750.

MediaLB Interrupt Enable Register.

Table 11-1749. MLB_MIEN Instances

Instance	Physical Address
MLB	021C 642Ch

Figure 11-795. MLB_MIEN Register

31	30	29	28	27	26	25	24
RESERVED		CTX_BREAK	CTX_PE	CTX_DONE	CRX_BREAK	CRX_PE	CRX_DONE
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	ATX_BREAK	ATX_PE	ATX_DONE	ARX_BREAK	ARX_PE	ARX_DONE	SYNC_PE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						ISOC_BUFO	ISOC_PE
R/W-0h						R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1750. MLB_MIEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R/W	0h	Reserved
29	CTX_BREAK	R/W	0h	Control Tx break enable. When set, a ReceiverBreak response received from the receiver on a control Tx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.
28	CTX_PE	R/W	0h	Control Tx protocol error enable. When set, a ProtocolError generated by the receiver on a control Tx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.
27	CTX_DONE	R/W	0h	Control Tx packet done enable. When set, a packet transmitted with no errors on a control Tx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.
26	CRX_BREAK	R/W	0h	Control Rx break enable. When set, a ControlBreak command received from the transmitter on a control Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.
25	CRX_PE	R/W	0h	Control Rx protocol error enable. When set, a ProtocolError detected on a control Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.
24	CRX_DONE	R/W	0h	Control Rx packet done enable. When set, a packet received with no errors on a control Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.
23	RESERVED	R/W	0h	Reserved
22	ATX_BREAK	R/W	0h	Asynchronous Tx break enable. When set, a ReceiverBreak response received from the receiver on an asynchronous Tx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.

Table 11-1750. MLB_MIEN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	ATX_PE	R/W	0h	Asynchronous Tx protocol error enable. When set, a ProtocolError generated by the receiver on an asynchronous Tx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.
20	ATX_DONE	R/W	0h	Asynchronous Tx packet done enable. When set, a packet transmitted with no errors on an asynchronous Tx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.
19	ARX_BREAK	R/W	0h	Asynchronous Rx break enable. When set, a AsyncBreak command received from the transmitter on an asynchronous Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.
18	ARX_PE	R/W	0h	Asynchronous Rx protocol error enable. When set, a ProtocolError detected on an asynchronous Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.
17	ARX_DONE	R/W	0h	Asynchronous Rx done enable. When set, a packet received with no errors on an asynchronous Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.
16	SYNC_PE	R/W	0h	Synchronous protocol error enable. When set, a ProtocolError detected on a synchronous Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.
15-2	RESERVED	R/W	0h	Reserved
1	ISOC_BUFO	R/W	0h	Isochronous Rx buffer overflow enable. When set, a buffer overflow on an isochronous Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set. This occurs only when isochronous flow control is disabled.
0	ISOC_PE	R/W	0h	Isochronous Rx protocol error enable. When set, a ProtocolError detected on an isochronous Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.

Table 11-1751. Register Call Summary for MLB_MIEN

MLB Programming Guide • Channel Servicing: [0][1][2][3][4][5] • Channel Initialization: [0]
MLB Registers • MLB Registers: [0] • MLB_MS0 Register (Offset = Ch) [reset = 0h]: [0] • MLB_MIEN Register (Offset = 2Ch) [reset = 0h]: [0]

11.11.6.7 MLB_MLBC1 Register (Offset = 43Ch) [reset = 0h]

MLB_MLBC1 is shown in Figure 11-796 and described in Table 11-1753.

MediaLB Control 1 Register.

Table 11-1752. MLB_MLBC1 Instances

Instance	Physical Address
MLB	021C 643Ch

Figure 11-796. MLB_MLBC1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
NDA							
R/W-0h							
7	6	5	4	3	2	1	0
CLKM	LOCK	RESERVED					
R/W-0h	R/W-0h	R-0h					

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1753. MLB_MLBC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	NDA	R/W	0h	Node device address. Used for system commands directed to individual MediaLB nodes.
7	CLKM	R/W	0h	MediaLB clock missing status. Set when MediaLB clock is not toggling at the pin; cleared by software. Writing a 1 has no effect.
6	LOCK	R/W	0h	MediaLB lock error status. Set when MediaLB is unlocked; cleared by software. Writing a 1 has no effect.
5-0	RESERVED	R	0h	Reserved

Table 11-1754. Register Call Summary for MLB_MLBC1

MLB Programming Guide • Channel Initialization: [0][1][2][3]
MLB Registers • MLB Registers: [0] • MLB_MSS Register (Offset = 20h) [reset = 0h]: [0] • MLB_MLBC1 Register (Offset = 3Ch) [reset = 0h]: [0]

11.11.6.8 MLB_HCTL Register (Offset = 480h) [reset = 0h]

MLB_HCTL is shown in Figure 11-797 and described in Table 11-1756.

HBI Control Register.

Table 11-1755. MLB_HCTL Instances

Instance	Physical Address
MLB	021C 6480h

Figure 11-797. MLB_HCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
EN	RESERVED						
R/W-0h	R/W-0h						
7	6	5	4	3	2	1	0
RESERVED							
R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1756. MLB_HCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Reserved
15	EN	R/W	0h	HBI enable: 1 = enabled, 0 = disabled
14-0	RESERVED	R/W	0h	Reserved (write default value)

Table 11-1757. Register Call Summary for MLB_HCTL

MLB Programming Guide • Channel Initialization: [0]
MLB Registers • MLB Registers: [0] • MLB_HCTL Register (Offset = 80h) [reset = 0h]: [0]

11.11.6.9 MLB_HCMR0 Register (Offset = 488h) [reset = 0h]

MLB_HCMR0 is shown in Figure 11-798 and described in Table 11-1759.

HBI Channel Mask 0 Register.

Table 11-1758. MLB_HCMR0 Instances

Instance	Physical Address
MLB	021C 6488h

Figure 11-798. MLB_HCMR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHM_31_0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1759. MLB_HCMR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHM_31_0	R/W	0h	Bitwise channel mask bit: 0 = masked, 1 = unmasked

Table 11-1760. Register Call Summary for MLB_HCMR0

MLB Programming Guide • Channel Initialization: [0][1]
MLB Registers • MLB Registers: [0] • MLB_HCMR0 Register (Offset = 88h) [reset = 0h]: [0]

11.11.6.10 MLB_HCMR1 Register (Offset = 48Ch) [reset = 0h]

MLB_HCMR1 is shown in Figure 11-799 and described in Table 11-1762.

HBI Channel Mask 1 Register.

Table 11-1761. MLB_HCMR1 Instances

Instance	Physical Address
MLB	021C 648Ch

Figure 11-799. MLB_HCMR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHM_63_32																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1762. MLB_HCMR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHM_63_32	R/W	0h	Bitwise channel mask bit: 0 = masked, 1 = unmasked

Table 11-1763. Register Call Summary for MLB_HCMR1

MLB Programming Guide • Channel Initialization: [0][1]
MLB Registers • MLB Registers: [0] • MLB_HCMR1 Register (Offset = 8Ch) [reset = 0h]: [0]

11.11.6.11 MLB_HCER0 Register (Offset = 490h) [reset = 0h]

MLB_HCER0 is shown in Figure 11-800 and described in Table 11-1765.

HBI Channel Error 0 Register.

Table 11-1764. MLB_HCER0 Instances

Instance	Physical Address
MLB	021C 6490h

Figure 11-800. MLB_HCER0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CERR_31_0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1765. MLB_HCER0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CERR_31_0	R/W	0h	Bitwise channel error bit. Writing a 1 has no effect.

Table 11-1766. Register Call Summary for MLB_HCER0

MLB Registers
- MLB Registers: [0]
- MLB_HCER0 Register (Offset = 90h) [reset = 0h]: [0]

11.11.6.12 MLB_HCER1 Register (Offset = 494h) [reset = 0h]

MLB_HCER1 is shown in Figure 11-801 and described in Table 11-1768.

HBI Channel Error 1 Register.

Table 11-1767. MLB_HCER1 Instances

Instance	Physical Address
MLB	021C 6494h

Figure 11-801. MLB_HCER1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CERR_63_32																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1768. MLB_HCER1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CERR_63_32	R/W	0h	Bitwise channel error bit. Writing a 1 has no effect.

Table 11-1769. Register Call Summary for MLB_HCER1

MLB Registers
- MLB Registers: [0]
- MLB_HCER1 Register (Offset = 94h) [reset = 0h]: [0]

11.11.6.13 MLB_HCBR0 Register (Offset = 498h) [reset = 0h]

MLB_HCBR0 is shown in Figure 11-802 and described in Table 11-1771.

HBI Channel Busy 0 Register.

Table 11-1770. MLB_HCBR0 Instances

Instance	Physical Address
MLB	021C 6498h

Figure 11-802. MLB_HCBR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHB_31_0																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1771. MLB_HCBR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHB_31_0	R	0h	Bitwise channel busy bit: 0 = idle, 1 = busy

Table 11-1772. Register Call Summary for MLB_HCBR0

MLB Registers
- MLB Registers: [0]
- MLB_HCBR0 Register (Offset = 98h) [reset = 0h]: [0]

11.11.6.14 MLB_HCBR1 Register (Offset = 49Ch) [reset = 0h]

MLB_HCBR1 is shown in Figure 11-803 and described in Table 11-1774.

HBI Channel Busy 1 Register.

Table 11-1773. MLB_HCBR1 Instances

Instance	Physical Address
MLB	021C 649Ch

Figure 11-803. MLB_HCBR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHB_63_32																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1774. MLB_HCBR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHB_63_32	R	0h	Bitwise channel busy bit: 0 = idle, 1 = busy

Table 11-1775. Register Call Summary for MLB_HCBR1

MLB Registers
- MLB Registers: [0]
- MLB_HCBR1 Register (Offset = 9Ch) [reset = 0h]: [0]

11.11.6.15 MLB_MDAT0 Register (Offset = 4C0h) [reset = 0h]

MLB_MDAT0 is shown in Figure 11-804 and described in Table 11-1777.

Memory Interface Data 0 Register.

Table 11-1776. MLB_MDAT0 Instances

Instance	Physical Address
MLB	021C 64C0h

Figure 11-804. MLB_MDAT0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_31_0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1777. MLB_MDAT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA_31_0	R/W	0h	CTR data - bits[31-0] of 128-bit entry or DBR data - bits[7-0] of 8-bit entry

Table 11-1778. Register Call Summary for MLB_MDAT0

MLB Programming Guide • Channel Table RAM Access: [0][1]
MLB Registers • MLB Registers: [0] • MLB_MDAT0 Register (Offset = C0h) [reset = 0h]: [0]

11.11.6.16 MLB_MDAT1 Register (Offset = 4C4h) [reset = 0h]

MLB_MDAT1 is shown in Figure 11-805 and described in Table 11-1780.

Memory Interface Data 1 Register.

Table 11-1779. MLB_MDAT1 Instances

Instance	Physical Address
MLB	021C 64C4h

Figure 11-805. MLB_MDAT1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_63_32																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1780. MLB_MDAT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA_63_32	R/W	0h	CTR data - bits[63-32] of 128-bit entry

Table 11-1781. Register Call Summary for MLB_MDAT1

MLB Programming Guide • Channel Table RAM Access: [0][1]
MLB Registers • MLB Registers: [0] • MLB_MDAT1 Register (Offset = C4h) [reset = 0h]: [0]

11.11.6.17 MLB_MDAT2 Register (Offset = 4C8h) [reset = 0h]

MLB_MDAT2 is shown in Figure 11-806 and described in Table 11-1783.

Memory Interface Data 2 Register.

Table 11-1782. MLB_MDAT2 Instances

Instance	Physical Address
MLB	021C 64C8h

Figure 11-806. MLB_MDAT2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_95_64																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1783. MLB_MDAT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA_95_64	R/W	0h	CTR data - bits[95-64] of 128-bit entry

Table 11-1784. Register Call Summary for MLB_MDAT2

MLB Programming Guide • Channel Table RAM Access: [0][1]
MLB Registers • MLB Registers: [0] • MLB_MDAT2 Register (Offset = C8h) [reset = 0h]: [0]

11.11.6.18 MLB_MDAT3 Register (Offset = 4CCh) [reset = 0h]

MLB_MDAT3 is shown in Figure 11-807 and described in Table 11-1786.

Memory Interface Data 3 Register.

Table 11-1785. MLB_MDAT3 Instances

Instance	Physical Address
MLB	021C 64CCh

Figure 11-807. MLB_MDAT3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_127_96																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1786. MLB_MDAT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA_127_96	R/W	0h	CTR data - bits[127-96] of 128-bit entry

Table 11-1787. Register Call Summary for MLB_MDAT3

MLB Programming Guide • Channel Table RAM Access: [0][1]
MLB Registers • MLB Registers: [0] • MLB_MDAT3 Register (Offset = CCh) [reset = 0h]: [0]

11.11.6.19 MLB_MDWE0 Register (Offset = 4D0h) [reset = 0h]

MLB_MDWE0 is shown in Figure 11-808 and described in Table 11-1789.

Memory Interface Data Write Enable 0 Register.

Table 11-1788. MLB_MDWE0 Instances

Instance	Physical Address
MLB	021C 64D0h

Figure 11-808. MLB_MDWE0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASK_31_0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1789. MLB_MDWE0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MASK_31_0	R/W	0h	Bitwise write enable for CTR data - bits[31-0] (0 = disabled, 1 = enabled)

Table 11-1790. Register Call Summary for MLB_MDWE0

MLB Programming Guide • Channel Initialization: [0][1] • Channel Table RAM Access: [0]
MLB Registers • MLB Registers: [0] • MLB_MDWE0 Register (Offset = D0h) [reset = 0h]: [0]

11.11.6.20 MLB_MDWE1 Register (Offset = 4D4h) [reset = 0h]

MLB_MDWE1 is shown in Figure 11-809 and described in Table 11-1792.

Memory Interface Data Write Enable 1 Register.

Table 11-1791. MLB_MDWE1 Instances

Instance	Physical Address
MLB	021C 64D4h

Figure 11-809. MLB_MDWE1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASK_63_32																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1792. MLB_MDWE1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MASK_63_32	R/W	0h	Bitwise write enable for CTR data - bits[63-32] (0 = disabled, 1 = enabled)

Table 11-1793. Register Call Summary for MLB_MDWE1

MLB Programming Guide • Channel Initialization: [0] • Channel Table RAM Access: [0]
MLB Registers • MLB Registers: [0] • MLB_MDWE1 Register (Offset = D4h) [reset = 0h]: [0]

11.11.6.21 MLB_MDWE2 Register (Offset = 498h) [reset = 0h]

MLB_MDWE2 is shown in Figure 11-810 and described in Table 11-1795.

Memory Interface Data Write Enable 2 Register.

Table 11-1794. MLB_MDWE2 Instances

Instance	Physical Address
MLB	021C 6498h

Figure 11-810. MLB_MDWE2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASK_95_64																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1795. MLB_MDWE2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MASK_95_64	R/W	0h	Bitwise write enable for CTR data - bits[95-64] (0 = disabled, 1 = enabled)

Table 11-1796. Register Call Summary for MLB_MDWE2

MLB Programming Guide • Channel Initialization: [0] • Channel Table RAM Access: [0]
MLB Registers • MLB Registers: [0] • MLB_MDWE2 Register (Offset = 98h) [reset = 0h]: [0]

11.11.6.22 MLB_MDWE3 Register (Offset = 49Ch) [reset = 0h]

MLB_MDWE3 is shown in Figure 11-811 and described in Table 11-1798.

Memory Interface Data Write Enable 3 Register.

Table 11-1797. MLB_MDWE3 Instances

Instance	Physical Address
MLB	021C 649Ch

Figure 11-811. MLB_MDWE3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASK_127_96																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1798. MLB_MDWE3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MASK_127_96	R/W	0h	Bitwise write enable for CTR data - bits[127-96] (0 = disabled, 1 = enabled)

Table 11-1799. Register Call Summary for MLB_MDWE3

MLB Programming Guide • Channel Initialization: [0] • Channel Table RAM Access: [0]
MLB Registers • MLB Registers: [0] • MLB_MDWE3 Register (Offset = 9Ch) [reset = 0h]: [0]

11.11.6.23 MLB_MCTL Register (Offset = 4E0h) [reset = 0h]

MLB_MCTL is shown in Figure 11-812 and described in Table 11-1801.

Memory Interface Control Register.

Table 11-1800. MLB_MCTL Instances

Instance	Physical Address
MLB	021C 64E0h

Figure 11-812. MLB_MCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							XCMP
R/W-0h							R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1801. MLB_MCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	Reserved (write default value)
0	XCMP	R/W	0h	Transfer complete (write 0 to clear). Writing a 1 has no effect.

Table 11-1802. Register Call Summary for MLB_MCTL

MLB Programming Guide • Channel Table RAM Access: [0][1]
MLB Registers • MLB Registers: [0] • MLB_MCTL Register (Offset = E0h) [reset = 0h]: [0]

11.11.6.24 MLB_MADR Register (Offset = 4E4h) [reset = 0h]

MLB_MADR is shown in Figure 11-813 and described in Table 11-1804.

Memory Interface Address Register.

Table 11-1803. MLB_MADR Instances

Instance	Physical Address
MLB	021C 64E4h

Figure 11-813. MLB_MADR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WNR	TB	RESERVED													
R/W-0h	R/W-0h	R/W-0h													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		ADDR_13_8						ADDR_7_0							
R/W-0h		R/W-0h						R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1804. MLB_MADR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	WNR	R/W	0h	Write-Not-Read selection (0 = read, 1 = write)
30	TB	R/W	0h	Target location bit (0 = selects CTR, 1 = selects DBR)
29-14	RESERVED	R/W	0h	Reserved (write default value)
13-8	ADDR_13_8	R/W	0h	DBR address of 8-bit entry - bits[13-8]
7-0	ADDR_7_0	R/W	0h	CTR address of 128-bit entry or DBR address of 8-bit entry - bits[7-0]

Table 11-1805. Register Call Summary for MLB_MADR

MLB Programming Guide • Channel Table RAM Access: [0][1][2][3][4][5]
MLB Registers • MLB Registers: [0] • MLB_MADR Register (Offset = E4h) [reset = 0h]: [0]

11.11.6.25 MLB_ACTL Register (Offset = 7C0h) [reset = 0h]

MLB_ACTL is shown in Figure 11-814 and described in Table 11-1807.

AHB Control Register.

Table 11-1806. MLB_ACTL Instances

Instance	Physical Address
MLB	021C 67C0h

Figure 11-814. MLB_ACTL Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED			MPB	RESERVED	DMA_MODE	SMX	SCE
R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1807. MLB_ACTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R/W	0h	Reserved (write default value)
4	MPB	R/W	0h	Packet buffering mode: 0 = single-packet mode 1 = multiple-packet mode
3	RESERVED	R/W	0h	Reserved (write default value)
2	DMA_MODE	R/W	0h	DMA Mode: 0 = DMA Mode 0 1 = DMA Mode 1
1	SMX	R/W	0h	AHB interrupt mux enable: 0 = MLB_ACSR0 generates an interrupt on MLB_DMA_CH_INT0; MLB_ACSR1 generates an interrupt on MLB_DMA_CH_INT1 1 = MLB_ACSR0 and MLB_ACSR1 generate an interrupts on MLB_DMA_CH_INT0 only
0	SCE	R/W	0h	Software clear enable: 0 = Hardware clears interrupt after a MLB_ACSR0 and MLB_ACSR0 register read 1 = Software clears interrupt

Table 11-1808. Register Call Summary for MLB_ACTL

MLB Programming Guide
- Channel Servicing: [0][1][2]
- Channel Initialization: [0][1]

Table 11-1808. Register Call Summary for MLB_ACTL (continued)

MLB Registers
• MLB Registers: [0] • MLB_ACTL Register (Offset = C0h) [reset = 0h]: [0]

11.11.6.26 MLB_ACSR0 Register (Offset = 7D0h) [reset = 0h]

MLB_ACSR0 is shown in Figure 11-815 and described in Table 11-1810.

AHB Channel Status 0 Register.

Table 11-1809. MLB_ACSR0 Instances

Instance	Physical Address
MLB	021C 67D0h

Figure 11-815. MLB_ACSR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHS_31_0																															
R/W1toCl-0h																															

LEGEND: R/W1toCl = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-1810. MLB_ACSR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHS_31_0	R/W1toCl	0h	Interrupt status for logical channels 31 to 0: 0 = None 1 = Interrupt

Table 11-1811. Register Call Summary for MLB_ACSR0

MLB Programming Guide • Channel Servicing: [0][1][2][3][4] • Channel Initialization: [0]
MLB Registers • MLB Registers: [0] • MLB_ACTL Register (Offset = C0h) [reset = 0h]: [0][1][2][3] • MLB_ACSR0 Register (Offset = D0h) [reset = 0h]: [0]

11.11.6.27 MLB_ACSR1 Register (Offset = 7D4h) [reset = 0h]

MLB_ACSR1 is shown in Figure 11-816 and described in Table 11-1813.

AHB Channel Status 1 Register.

Table 11-1812. MLB_ACSR1 Instances

Instance	Physical Address
MLB	021C 67D4h

Figure 11-816. MLB_ACSR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHS_63_32																															
R/W1toCl-0h																															

LEGEND: R/W1toCl = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-1813. MLB_ACSR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHS_63_32	R/W1toCl	0h	Interrupt status for logical channels 32 to 63: 0 = None 1 = Interrupt

Table 11-1814. Register Call Summary for MLB_ACSR1

MLB Programming Guide • Channel Servicing: [0][1][2][3][4] • Channel Initialization: [0]
MLB Registers • MLB Registers: [0] • MLB_ACTL Register (Offset = C0h) [reset = 0h]: [0][1] • MLB_ACSR1 Register (Offset = D4h) [reset = 0h]: [0]

11.11.6.28 MLB_ACMR0 Register (Offset = 7D8h) [reset = 0h]

MLB_ACMR0 is shown in Figure 11-817 and described in Table 11-1816.

AHB Channel Mask 0 Register.

Table 11-1815. MLB_ACMR0 Instances

Instance	Physical Address
MLB	021C 67D8h

Figure 11-817. MLB_ACMR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHM_31_0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1816. MLB_ACMR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHM_31_0	R/W	0h	Bitwise channel mask bit: 0 = Masked 1 = Unmasked

Table 11-1817. Register Call Summary for MLB_ACMR0

MLB Programming Guide • Channel Servicing: [0][1] • Channel Initialization: [0]
MLB Registers • MLB Registers: [0] • MLB_ACMR0 Register (Offset = D8h) [reset = 0h]: [0]

11.11.6.29 MLB_ACMR1 Register (Offset = 7DCh) [reset = 0h]

MLB_ACMR1 is shown in Figure 11-818 and described in Table 11-1819.

AHB Channel Mask 1 Register.

Table 11-1818. MLB_ACMR1 Instances

Instance	Physical Address
MLB	021C 67DCh

Figure 11-818. MLB_ACMR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHM_63_32																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1819. MLB_ACMR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHM_63_32	R/W	0h	Bitwise channel mask bit: 0 = Masked 1 = Unmasked

Table 11-1820. Register Call Summary for MLB_ACMR1

MLB Programming Guide • Channel Servicing: [0][1] • Channel Initialization: [0]
MLB Registers • MLB Registers: [0] • MLB_ACMR1 Register (Offset = DCh) [reset = 0h]: [0]

11.12 MMC/SD

11.12.1 MMC/SD Overview

The Multimedia Card (MMC), Secure Digital (SD), and Secure Digital I/O (SDIO) high speed controller (MMC/SD) provides an interface between local host (LH) such as microprocessor unit (MPU) or digital signal processor (DSP) and either MMC, SD memory card, or SDIO card and handles MMC/SD/SDIO transactions with minimal LH intervention.

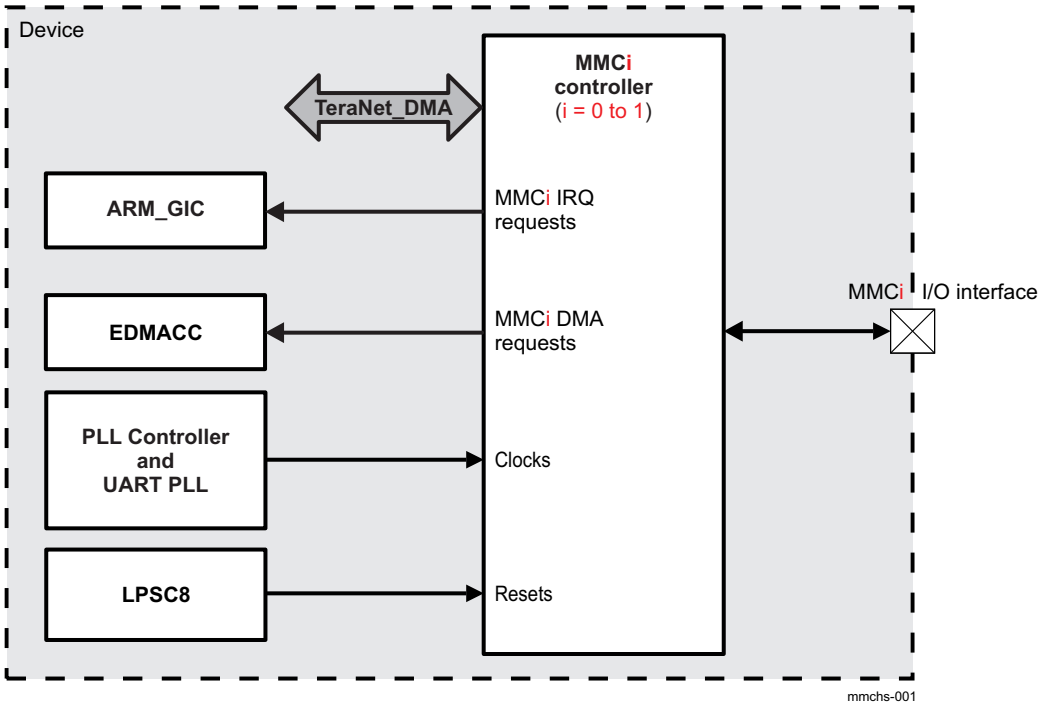
The MMC/SD host controller deals with MMC/SD protocol at transmission level, data packing, adding cyclic redundancy checks (CRCs), start/end bit insertion, and checking for syntactical correctness.

The application interface can send every MMC/SD command and either poll for the status of the adapter or wait for an interrupt request, which is sent back in case of exceptions or to warn of the end of operation.

The application interface can read card responses or card flag registers. It can also mask individual interrupt sources. All these operations can be performed by reading and writing control registers. The MMC/SD host controller also supports either two DMA slave channels or DMA master access (in this case, slave DMA channels are deactivated) depending on its integration generic parameter setting MMCHS_HL_HWINFO[0] MADMA_EN bit and MMCHS_CON[20] DMA_MNS bit.

There are two MMC/SD host controllers inside the device. Each controller has 8-bit wide data bus. The MMC/SDi controller is also referred to as MMCi. Figure 11-819 gives an overview of the MMCi (i = 0 to 1) controller.

Figure 11-819. MMCi Overview (i = 0 to 1)



11.12.1.1 MMC Features

This section describes the features supplied by the MMC host controllers.

Compliance with standards:

- Full compliance with MMC/eMMC command/response sets as defined in the JC64 MMC/eMMC standard specification, v4.5.
- Full compliance with SD command/response sets as defined in the SD Physical Layer specification v3.01.
- Full compliance with SDIO command/response sets and interrupt/read-wait suspend-resume operations as defined in the SD part E1 specification v3.00.
- Full compliance with SD Host Controller Standard Specification sets as defined in the SD card specification Part A2 v3.00.

Main features of the MMC host controllers:

- Flexible architecture allowing support for new command structure
- 32-bit wide access bus to maximize bus throughput
- Designed for low power (Local Power Management)
- Programmable clock generation
- Card insertion/removal detection and write protect detection
- The slave interface supports:
  - 32-bit wide data bus
  - Streaming burst supported only with burst length up to 7
  - WNP supported
- The master interface supports:
  - 32-bit wide data bus
  - Burst supported
- Built-in 1024-byte buffer for read or write
- Two DMA channels, one interrupt line
- Support JC 64 v4.4.1 boot mode operations
- Support SDA 3.00 Part A2 programming model
- Support SDA 3.00 Part A2 DMA feature (ADMA2)
- Supported data transfer rates:
  - MMC0 supports the following data transfer rates (eMMC/SD):
    - SDR12 (3.3 V IOs): up to 12 MBps (24 MHz clock)
    - SDR25 (3.3 V IOs): up to 24 MBps (48 MHz clock)
    - HS mode (3.3 V IOs): up to 24 MBps (48 MHz clock)
    - DS mode (3.3 V IOs): up to 12 MBps (24 MHz clock)
    - Default SD mode 1-bit data transfer up to 24 Mbps (3 MBps)
  - MMC1 supports the following data transfer rates (eMMC):
    - SDR12 (1.8 V IOs): up to 12 MBps (24 MHz clock)
    - SDR25 (1.8 V IOs): up to 24 MBps (48 MHz clock)
    - DDR50 (1.8 V IOs): up to 48 MBps (48 MHz clock)
    - 1.8 V legacy modes with 1/4/8-bit single data rate at up to 26 MHz bus clock
- MMC0 Supports 3.3-V IO modes only
- MMC1 Supports 1.8-V IO modes only

The differences between the MMC host controller and a standard SD host controller defined by the *SD Card Specification, Part A2, SD Host Controller Standard Specification*, v3.00 are:

- The clock divider in the MMC host controller supports a wider range of frequency than specified in the *SD Memory Card Specifications*, v3.0. The MMC host controller supports odd and even clock ratio.
- The MMC host controller supports configurable busy time-out.
- ADMA2 64-bit mode is not supported.
- There is no external LED control.

The following features are not supported:

- Byte or half-word accesses. Only word accesses to the slave port are supported.
- MMC Out-of-band interrupt.
- Dual voltage I/O (MMC0 supports 3.3 V only. MMC1 supports 1.8 V only).
- No built-in hardware support for error correction codes (ECC).
- SPI transfers are not supported.
- Module doesn't support card insertion/removal sensing with pull up resistor on MMCi_DAT[3] data bus line as specified in the SD physical layer specification.

11.12.2 MMC/SD Environment

Each MMC host controller can support the following combination of external devices:

- One or more low/high speed MMC sharing the same bus, with respect to load and timings defined by standards on MMC bus depending on bus frequency.
- One single SD memory card or SDIO card.

NOTE: When a bus is shared between two devices and one device is inserted or removed while data transfer is occurring to the other device, it may create some protocol errors that the software will need to resolve.

11.12.2.1 MMC/SD Functional Modes

11.12.2.1.1 MMC Connected to a MMC/SD/SDIO Card or eMMC Device

Figure 11-820 shows the MMCi host controller (where i = 0 to 1) connected to a MMC/SD/SDIO card or eMMC device and its related external connections.

Figure 11-820. MMCi Host Controller Connected to a MMC/SD/SDIO Card or eMMC Device (where i = 0 to 1)

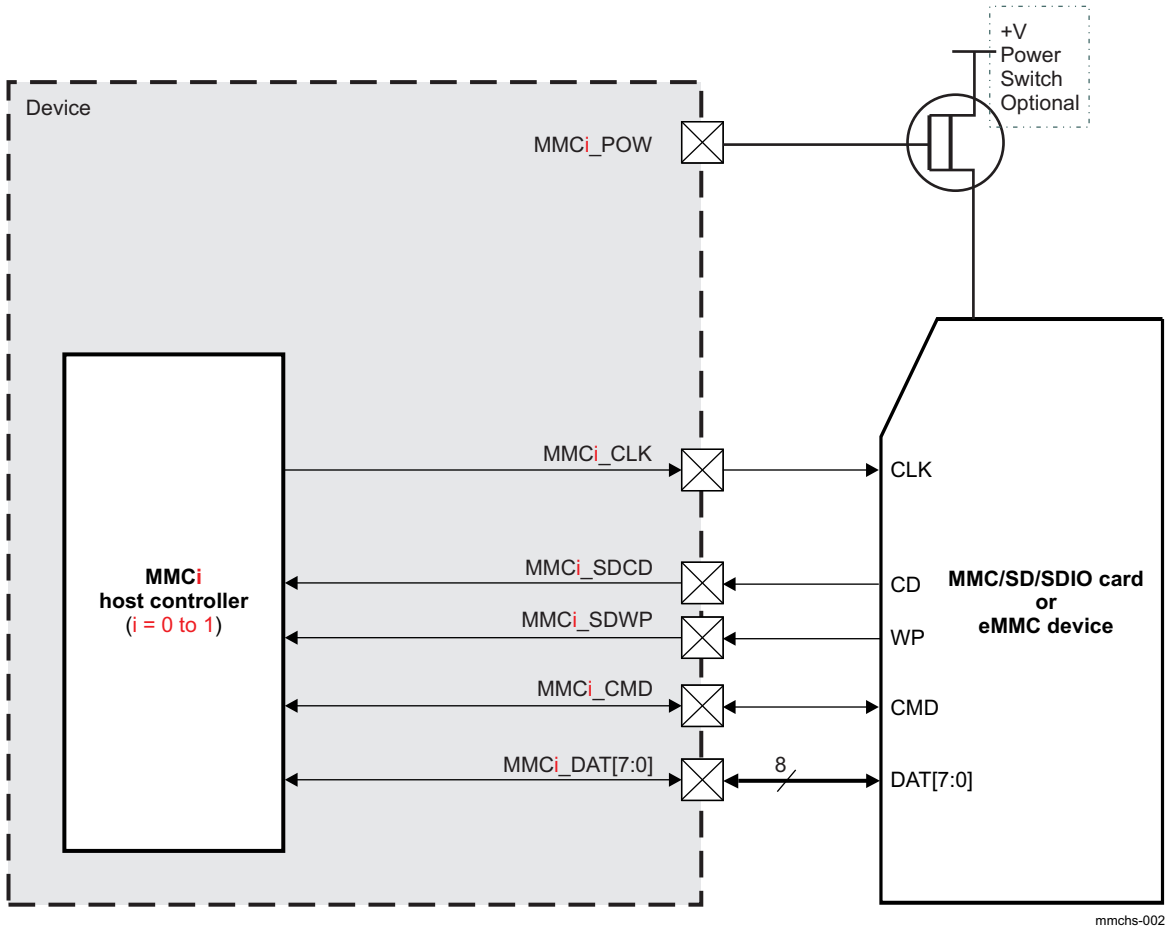


Table 11-1821 describes the MMCi host controller I/Os (where i = 0 to 1).

Table 11-1821. Description of MMCi host controller I/Os (where i = 0 to 1)

Instance	Signal name	I/O[1]	Description	Reset Value[2]
MMC0	MMC0_CLK	I/O	External clock for MMC/SD/SDIO card	0
	MMC0_CMD	I/O	Command line	HiZ[3]
	MMC0_DAT[7:0]	I/O	Data signals	HiZ[3]
	MMC0_SDCD	I	Card insertion/removal detection signal	HiZ
	MMC0_SDWP	I	Write protect detection signal	HiZ
	MMC0_POW	O	Card on/off power supply control	0

(1) I = Input; O = Output; I/O = Bidirectional
(2) HiZ = High Impedance
(3) Initialized as input upon reset

Table 11-1821. Description of MMCi host controller I/Os (where i = 0 to 1) (continued)

Instance	Signal name	I/O[1]	Description	Reset Value[2]
MMC1	MMC1_CLK	I/O	External clock for eMMC device	0
	MMC1_CMD	I/O	Command line	HiZ[3]
	MMC1_DAT[7:0]	I/O	Data signals	HiZ[3]
	MMC1_SDCD[4]	I	Card insertion/removal detection signal	HiZ
	MMC1_SDWP	I	Write protect detection signal	HiZ
	MMC1_POW[5]	O	Card on/off power supply control	0

(4) The MMC1_SDCD terminal must be pulled to a valid logic low state when booting from an on-board eMMC device connected to MMC1.

(5) The MMC1_POW signal is supported but not used because MMC1 host controller is intended for on-board eMMC devices which are always powered on.

11.12.2.2 Protocol and Data Format

The bus protocol between the MMCi host controller and the card is message-based. Each message is represented by one of the following parts:

- Command: A command starts an operation. The command is transferred serially from the MMC host controller to the card on the CMD line.
- Response: A response is an answer to a command. The response is sent from the card to the MMC host controller. It is transferred serially on the CMD line.
- Data: Data are transferred from the MMC host controller to the card or from a card to the MMC host controller using the data lines.
- Busy: The DAT[0] signal is maintained low by the card as far as it is programming the data received.
- CRC status: The CRC result is sent by the card through the DAT[0] line when executing a write transfer. In the case of transmission error, occurring on any of the active data lines, the card sends a negative CRC status on DAT[0]. In the case of successful transmission, over all active data lines, the card sends a positive CRC status on DAT[0] and starts the data programming procedure.

11.12.2.2.1 Protocol

There are two types of data transfer:

- Sequential operation
- Block-oriented operation

There are specific commands for each type of operation (sequential or block-oriented).

For information about commands and programming sequences supported by the MMC, SD, and SDIO cards, see the *Multimedia Card System Specification*, *SD Memory Card Specifications*, and *SDIO Card Specification (Part E1*).

Figure 11-821 and Figure 11-822 show how sequential operations are defined. Sequential operation is only for 1-bit transfer and initiates a continuous data stream. The transfer terminates when a stop command follows on the CMD line.

NOTE: Stream commands are supported only by MMCs.

Figure 11-821. Sequential Read Operation (MMCs Only)

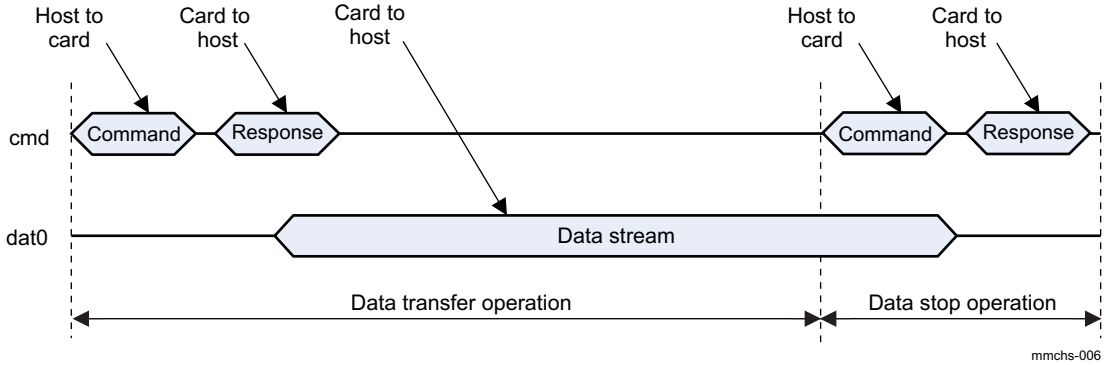


Figure 11-822. Sequential Write Operation (MMCs Only)

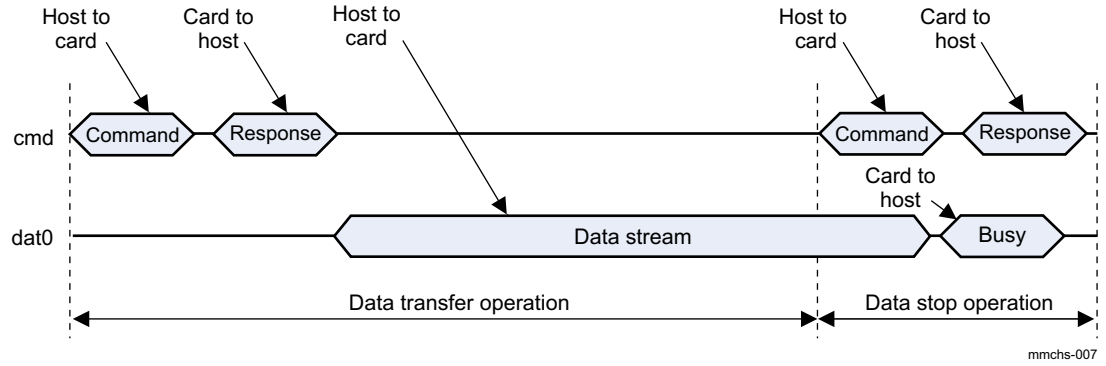


Figure 11-823 and Figure 11-824 show how multiple block-oriented operations are defined. A multiple block-oriented operation sends a data block plus CRC bits. The transfer terminates when a stop command follows on the CMD line. These operations are available for all kinds of cards.

Figure 11-823. Multiple Block Read Operation

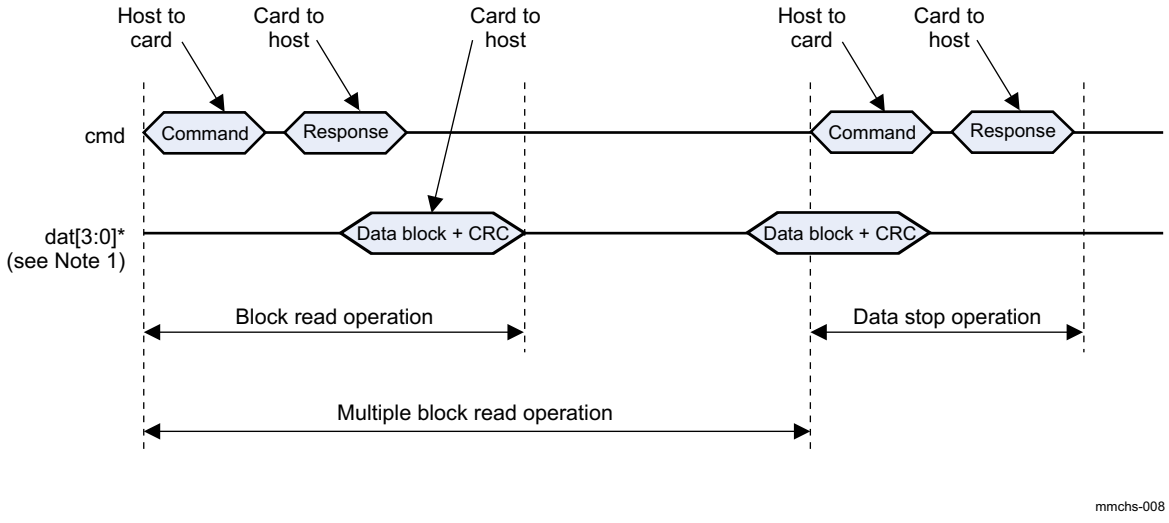
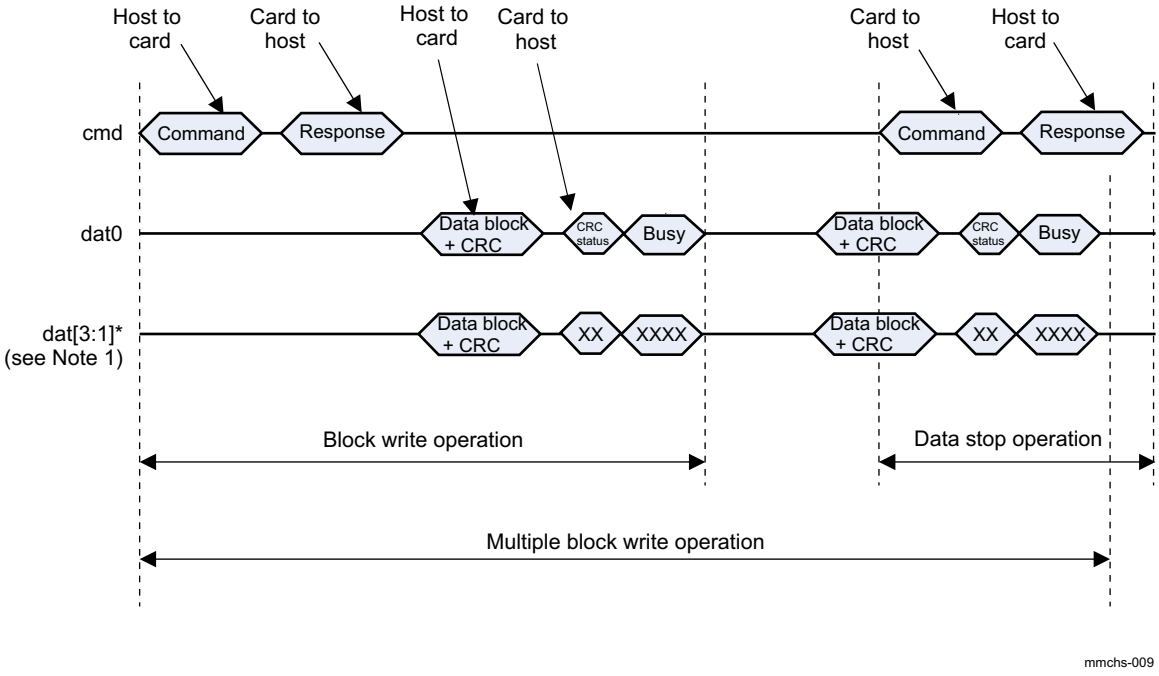


Figure 11-824. Multiple Block Write Operation With Card Busy Signal



NOTE:

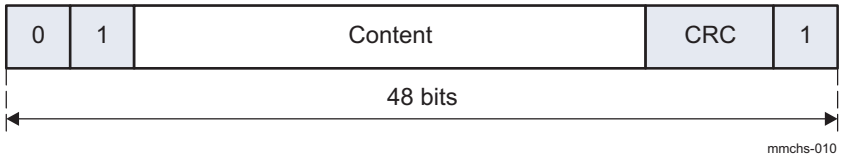
- The card busy signal is not always generated by the card; refer to Figure 11-823 and Figure 11-824, that show a particular case.
- Software must perform a software reset (set the MMCi.MMCHS_SYSCTL[26] SRD bit to 1h) after a data time-out to ensure that MMCi_CLK is stopped.
- For multiblock transfer, and especially for MMC cards, a transfer can be aborted without using a stop command. If a CMD23 is used before data transfer to define the number of blocks that will be transferred, then the transfer stops automatically after the last block (if the MMCs supports this feature).

11.12.2.2.2 Data Format

Coding Scheme for Command Token

Command tokens always start with 0 and end with 1. The second bit is a transmitter bit: 1 for a host command. The content is the command index (coded by 6 bits) and an argument (for example, an address), coded by 32 bits. The content is protected by 7-bit CRC checksum (see Figure 11-825).

Figure 11-825. Command Token Format



Coding Scheme for Response Token

Response tokens always start with 0 and end with 1. The second bit is a transmitter bit: 0 for a card response. The content is different for each type of response (R1, R2, R3, R4, and R5, R6, R7 [for SD]) and the content is protected by 7-bit CRC checksum (see Figure 11-826 and Figure 11-827). Depending on the type of commands sent to the card, the MMCi.MMCHS_CMD register must be configured differently to avoid false CRC or index errors to be flagged on command response (see Table 11-1822). For more information about response types, see the *Multimedia Card System Specification*, *SD Memory Card Specifications*, and *SDIO Card Specification*.

Figure 11-826. Response Token Format (R1, R3, R4, R5, R6, R7)

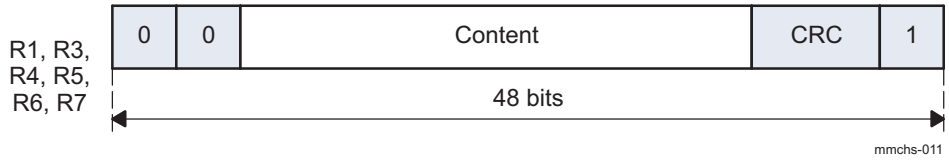


Figure 11-827. Response Token Format (R2)

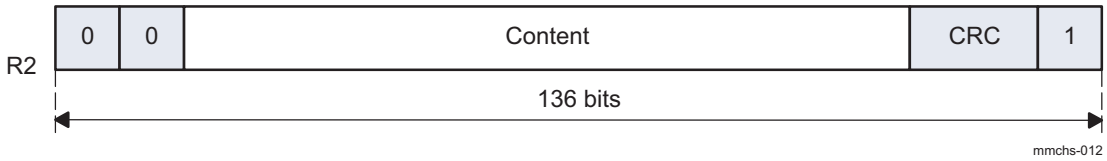


Table 11-1822. Relationship Between Configuration and Name of Response Type

Response Type MMCi.MMCHS_CMD[17-16] RSP_TYPE	Index Check Enable MMCi.MMCHS_CMD[20] CICE	CRC Check Enable MMCi.MMCHS_CMD[19] CCCE	Name of Response Type
00	0	0	No response
01	0	1	R2
10	0	0	R3 (R4 for SD cards)
10	1	1	R1, R6, R5, (R7 for SD cards)
11	1	1	R1b, R5b

Coding Scheme for Data Token

Data tokens always start with 0 and end with 1 (see Figure 11-828 through Figure 11-830).

Figure 11-828. Data Token Format for 1-Bit Transfers

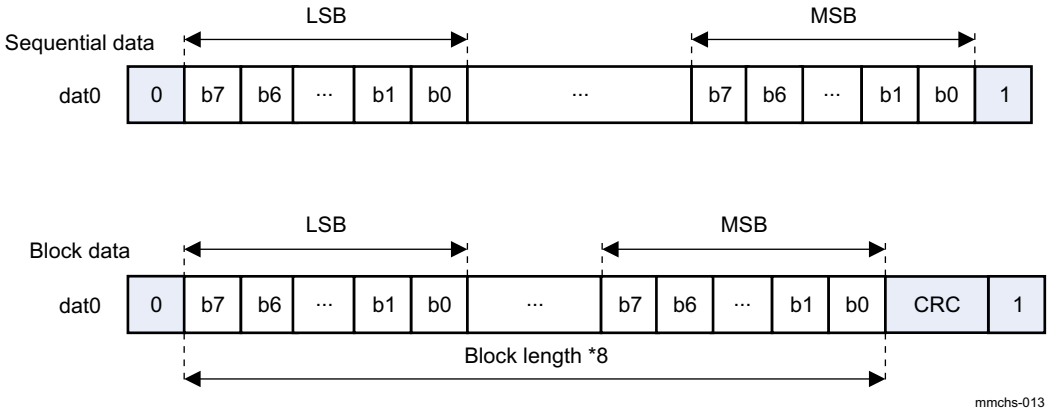


Figure 11-829. Data Token Format for 4-Bit Transfers

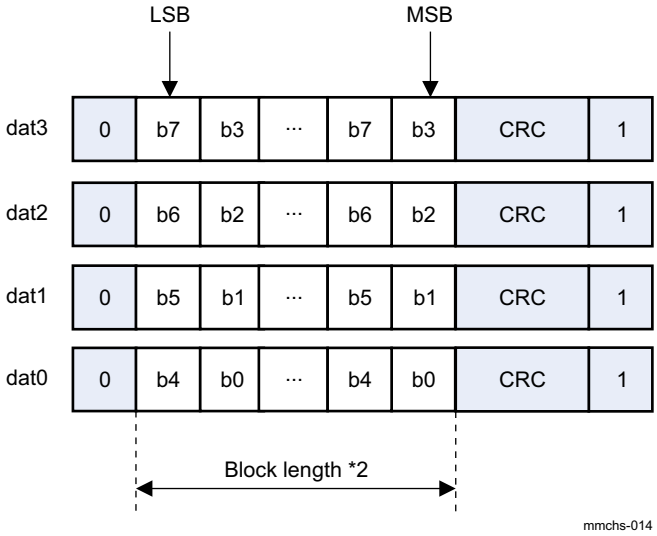
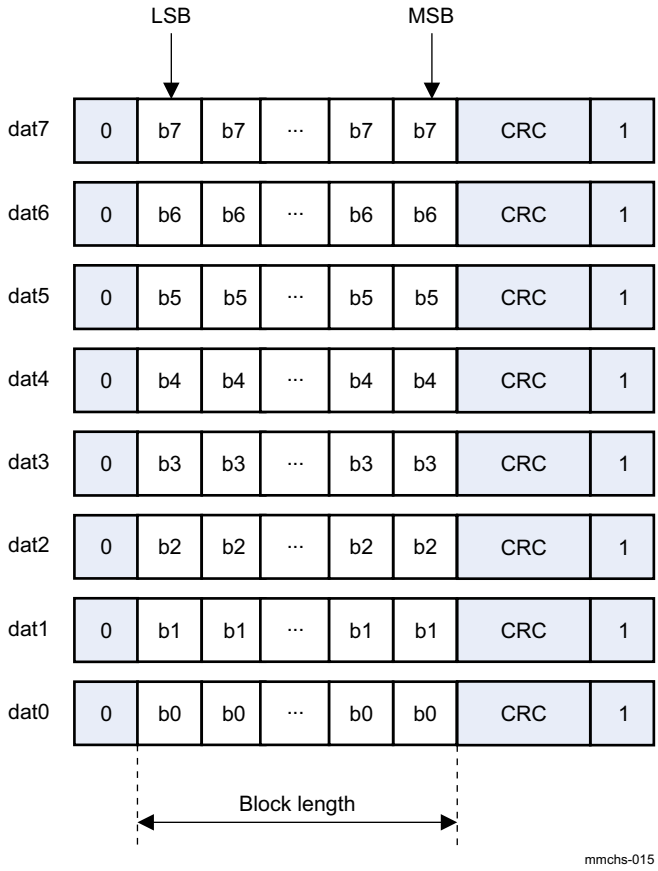


Figure 11-830. Data Token Format for 8-Bit Transfers



11.12.3 MMC/SD Integration

This section describes the module integration in the device, including information about clocks, resets, and hardware requests.

Figure 11-831 shows the MMC module integration.

Figure 11-831. MMC Integration

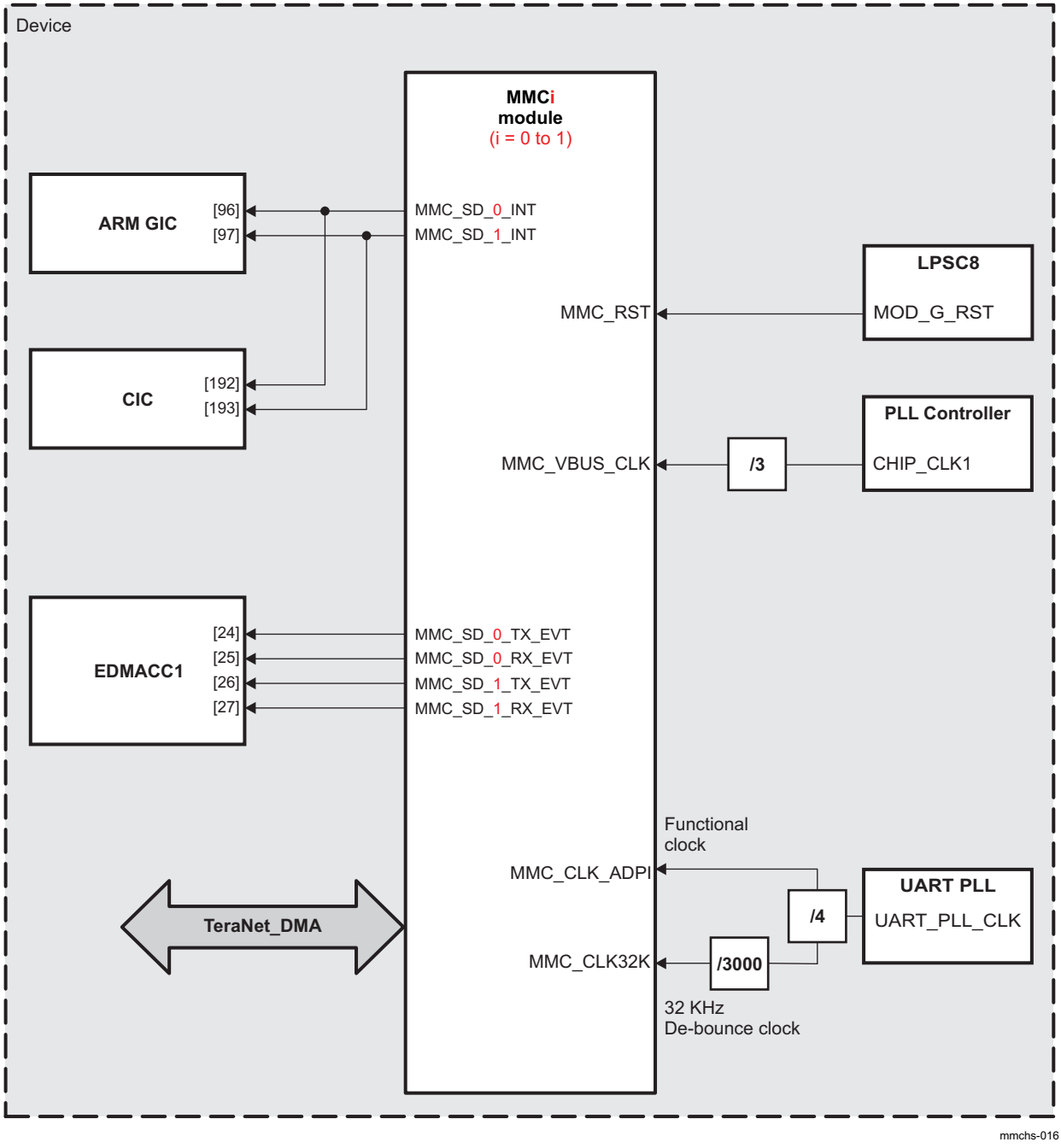


Table 11-1823 through Table 11-1825 summarize the integration of the module in the device.

Table 11-1823. MMC Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
MMC0	PD5	LPSC8	TeraNet_DMA
MMC1	PD5	LPSC8	TeraNet_DMA

Table 11-1824. MMC Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
MMC0	MMC_VBUS_CLK	CHIP_CLK1 / 3	PLL Controller	Interface Clock
	MMC_CLK_ADPI	UART_PLL_CLK / 4	UART PLL	Functional Clock
	MMC_CLK32K	UART_PLL_CLK / 12000	UART PLL	32 kHz Debounce Clock
MMC1	MMC_VBUS_CLK	CHIP_CLK1 / 3	PLL Controller	Interface Clock
	MMC_CLK_ADPI	UART_PLL_CLK / 4	UART PLL	Functional Clock
	MMC_CLK32K	UART_PLL_CLK / 12000	UART PLL	32 kHz Debounce Clock
Resets				
Module Instance	**Destination Signal Name**	**Source Signal**	**Source**	**Description**
MMC0	MMC_RST	MOD_G_RST	LPSC8	Asynchronous System Reset
MMC1	MMC_RST	MOD_G_RST	LPSC8	Asynchronous System Reset

Table 11-1825. MMC Hardware Requests

Interrupt Requests								
Module Instance	**Event Name**	**Mapped To Input Event [Number]**						**Description**
		ARM GIC	**CIC**	**ICSS0 INTC**	**ICSS1 INTC**	**DSP INTC**	**PMMC INTC**	
MMC0	MMC_SD_0_INT	[96]	[192]	–	–	–	–	MMC0 Interrupt Request
MMC1	MMC_SD_1_INT	[97]	[193]	–	–	–	–	MMC1 Interrupt Request
DMA Requests								
Module Instance	**Event Name**	**Mapped To Input Event [Number]**						**Description**
		EDMACC0			**EDMACC1**			
MMC0	MMC_SD_0_TX_EVT	–			[24]			MMC0 Transmit Event
	MMC_SD_0_RX_EVT	–			[25]			MMC0 Receive Event
MMC1	MMC_SD_1_TX_EVT	–			[26]			MMC1 Transmit Event
	MMC_SD_1_RX_EVT	–			[27]			MMC1 Receive Event

NOTE:

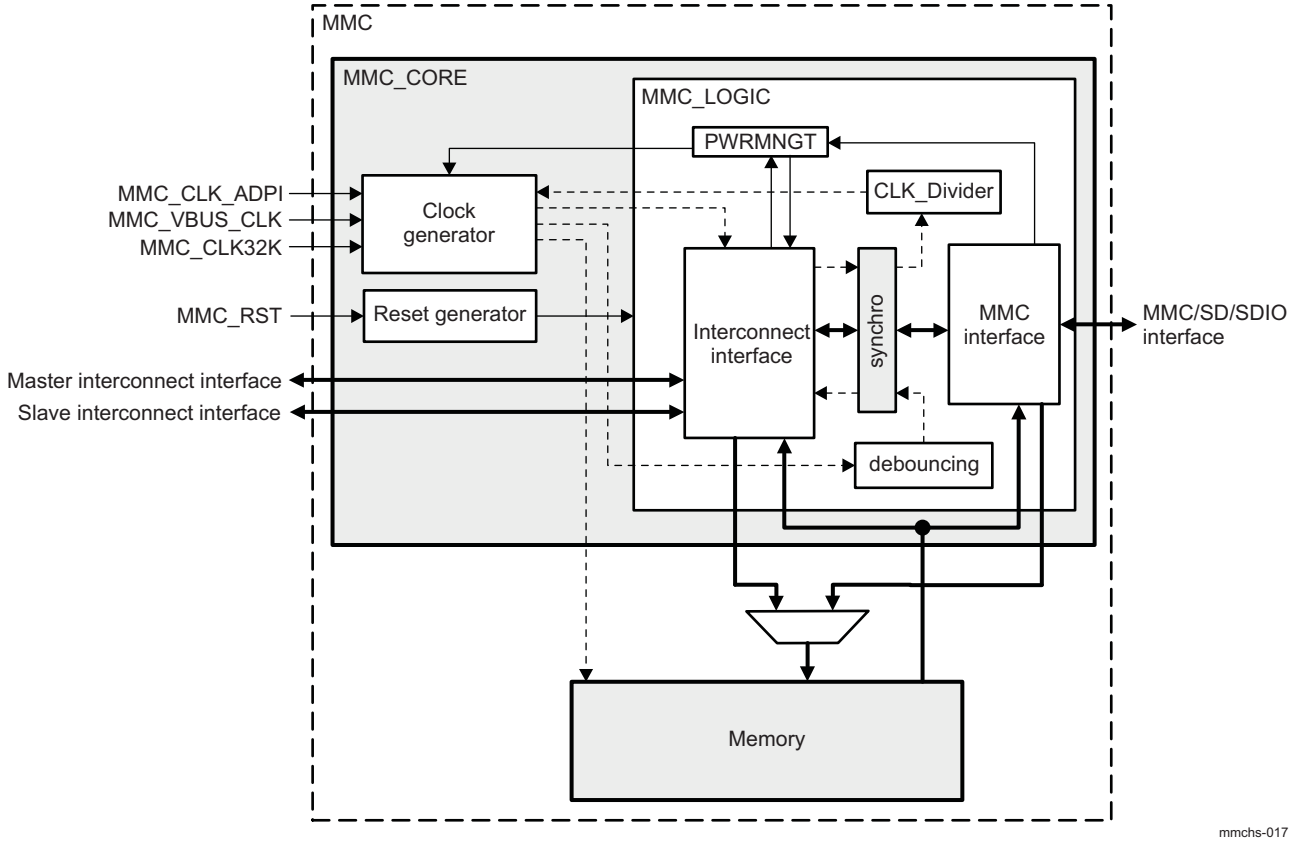
- For a description of the interrupt source, see Section 11.12.4.4, *Interrupt Requests*.
- For a description of the DMA source, see Section 11.12.4.5, *DMA Modes*.

11.12.4 MMC/SD Functional Description

11.12.4.1 Block Diagram

Figure 11-832 is a block diagram of the MMC host controller.

Figure 11-832. MMC Diagram



11.12.4.2 Resets

11.12.4.2.1 Hardware Reset

The module is reinitialized by the hardware (see Table 11-1824 for more information about reset signals).

The MMCi.MMCHS_SYSSTATUS[0] RESETDONE bit can be monitored by software to check whether the module is ready to use after a hardware reset.

NOTE: The functional clock (MMC_CLK_ADPI) and interface clock (MMC_VBUS_CLK) must be provided to the module to allow the MMCi.MMCHS_SYSSTATUS[0] RESETDONE status bit to be set.

The debounce clock (MMC_CLK32K) must be active to reset the module correctly.

This hardware reset signal has a global reset action on the module. All configuration registers and all state-machines are reset in all clock domains.

11.12.4.2.2 Software Reset

The module is reinitialized by software through the MMCi.MMCHS_SYSCONFIG[1] SOFTRESET bit. This bit has the same effect on the module logic as the hardware signal (MMC_RST), with the following exceptions:

- Debounce logic.
- MMCi.MMCHS_PSTATE, MMCi.MMCHS_CAPA, and MMCi.MMCHS_CUR_CAPA registers (see the corresponding register description).

The SOFTRESET bit is active high. The bit is automatically reinitialized to 0 by hardware. The MMCi.MMCHS_SYSCTL[24] SRA bit has the same action on the design as the MMCi.MMCHS_SYSCONFIG[1] SOFTRESET bit.

The MMCi.MMCHS_SYSSTATUS[0] RESETDONE bit can be monitored by software to check whether the module is ready to use after a software reset.

Moreover, two partial software reset bits are provided:

- MMCi.MMCHS_SYSCTL[26] SRD.
- MMCi.MMCHS_SYSCTL[25] SRC.

These two reset bits are useful to reinitialize data or command processes, respectively, in case of line conflict. When these bits are set to 1, a reset process is automatically released when the reset completes:

- The MMCi.MMCHS_SYSCTL[26] SRD bit resets all finite state-machines (FSMs) and status management that handle data transfers on the interface and functional sides.
- The MMCi.MMCHS_SYSCTL[25] SRC bit resets all FSMs and status management that handle command transfers on the interface and functional sides.

11.12.4.3 Power Management

The MMC host controller can save power by autogating of interface and functional clocks.

The autogating of interface and functional clocks occurs when the following conditions are met:

- The MMCi.MMCHS_SYSCONFIG[0] AUTOIDLE bit is set to 1.
- There is no transaction on the MMC interface.

The autogating of interface and functional clocks stops when the following conditions are met:

- A register access occurs through the slave interface interconnect.
- A transaction on the MMC interface starts.

Local Power Management

describes the power management features available for the MMCi controllers.

Table 11-1826. Local Power Management Features

Feature	Registers	Description
Clock auto gating	MMCHS_SYSCONFIG[0] AUTOIDLE	This bit allows a local power optimization inside the module by gating the MMC_VBUS_CLK clock upon the interface activity, or gating the MMC_CLK_ADPI clock upon the internal activity.
Clock activity	MMCHS_SYSCONFIG[8-9] CLOCKACTIVITY	For configuration details, see Table 11-1827.

Table 11-1827. Clock Activity Settings

CLOCKACTIVITY Values	Clock State When Module is in IDLE State		Features Available When Module is in IDLE State
	MMC_VBUS_CLK	MMC_CLK_ADPI	
00	OFF	OFF	None
10	OFF	ON	None
01	ON	OFF	None
11	ON	ON	All

11.12.4.4 Interrupt Requests

Several internal module events can generate an interrupt. Each interrupt has a status bit, an interrupt enable bit, and a signal status enable:

- The status of each type of interrupt is automatically updated in the MMCi.MMCHS_STAT register; it indicates which service is required.
- The interrupt status enable bits of the MMCi.MMCHS_IE register enable or disable the automatic update of the MMCi.MMCHS_STAT register on an event-by-event basis.
- The interrupt signal enable bits of the MMCi.MMCHS_ISE register enable or disable the transmission of an interrupt request on the interrupt line MMC_SD_i_INT on an event-by-event basis.

If an interrupt status is disabled in the MMCi.MMCHS_IE register, then the corresponding interrupt request is not transmitted, and the value of the corresponding interrupt signal enable in the MMCi.MMCHS_ISE register is ignored.

When an interrupt event occurs, the corresponding status bit is automatically set to 1h (the MMCi host controller updates the status bit) in the MMCi.MMCHS_STAT register. If a mask is later applied on the interrupt in the MMCi.MMCHS_ISE register, the interrupt request is deactivated.

When the interrupt source has not been serviced, if the interrupt status is cleared in the MMCi.MMCHS_STAT register and the corresponding mask is removed from the MMCi.MMCHS_ISE register, the interrupt status is not asserted again in the MMCi.MMCHS_STAT register and the MMCi host controller does not transmit an interrupt request.

NOTE: If the buffer write ready (BWR) interrupt or the buffer read ready (BRR) only interrupt are not serviced and are cleared in the MMCi.MMCHS_STAT register, and the corresponding mask is removed, then the MMCi host controller waits for the service of the interrupt without updating the status MMCi.MMCHS_STAT register or transmitting an interrupt request.

Table 11-1828 lists the event flags, and their mask, that can cause module interrupts.

Table 11-1828. Events

Event Flag	Event Mask	Map to	Description
MMCHS_STAT[29] BADA	MMCHS_IE[29] BADA_ENABLE	MMC_SD_i_INT	Bad access to data space. This bit is set automatically to indicate a bad access to buffer when not allowed: This bit is set during a read access to the data register (MMCHS_DATA) while buffer reads are not allowed (MMCHS_PSTATE[11] BRE = 0). This bit is set during a write access to the data register (MMCHS_DATA) while buffer writes are not allowed (MMCHS_PSTATE[10] BWE = 0).
MMCHS_STAT[28] CERR	MMCHS_IE[28] CERR_ENABLE	MMC_SD_i_INT	Card error. This bit is set automatically when there is at least one error in a response of type R1, R1b, R6, R5, or R5b. Only bits referenced as type E (error) in the status field in the response can set a card status error. An error bit in the response is flagged only if the corresponding bit in the card status response error MMCHS_CSRE is set. There is no card error detection for the auto CMD12 command.
MMCHS_STAT[26] TE	MMCHS_IE[26] TE_ENABLE	MMC_SD_i_INT	Tuning Error. This bit is set when an unrecoverable error is detected in a tuning circuit except during tuning procedure. The Tuning Error is with higher priority than the other error interrupts generated during data transfer.

Table 11-1828. Events (continued)

Event Flag	Event Mask	Map to	Description
MMCHS_STAT[25] ADMAE	MMCHS_IE[25] ADMAE_ENABLE	MMC_SD_i_INT	ADMA error. This bit is set when the host controller detects errors during an ADMA-based data transfer. The state of the ADMA at an error occurrence is saved in the ADMA error status register. In addition, the host controller generates this interrupt when it detects invalid descriptor data (Valid=0) at the ST_FDS state.
MMCHS_STAT[24] ACE	MMCHS_IE[24] ACE_ENABLE	MMC_SD_i_INT	Auto CMD12 error and Auto CMD23 error. This bit is set automatically when one of the bits MMCHS_AC12[4-0] changes from 0 to 1.
MMCHS_STAT[22] DEB	MMCHS_IE[22] DEB_ENABLE	MMC_SD_i_INT	Data end bit error. This bit is set automatically when detecting a 0 at the end bit position of read data on the DAT line or at the end position of the CRC status in write mode.
MMCHS_STAT[21] DCRC	MMCHS_IE[21] DCRC_ENABLE	MMC_SD_i_INT	Data CRC error. This bit is set automatically when there is a CRC16 error in the data phase response following a block read command or if there is a 3-bit CRC status difference of a position 010 token during a block write command.
MMCHS_STAT[20] DTO	MMCHS_IE[20] DTO_ENABLE	MMC_SD_i_INT	Data time-out error. This bit is set automatically according to the following conditions: • Busy time-out for R1b, R5b response type • Busy time-out after write CRC status • Write CRC status time-out • Read data time-out
MMCHS_STAT[19] CIE	MMCHS_IE[19] CIE_ENABLE	MMC_SD_i_INT	Command index error. This bit is set automatically when the response index differs from the corresponding command index previously emitted. The check is enabled through the MMCHS_CMD[20] CICE bit.
MMCHS_STAT[18] CEB	MMCHS_IE[18] CEB_ENABLE	MMC_SD_i_INT	Command end bit error. This bit is set automatically when detecting a 0 at the end bit position of a command response.
MMCHS_STAT[17] CCRC	MMCHS_IE[17] CCRC_ENABLE	MMC_SD_i_INT	Command CRC error. This bit is set automatically when a CRC7 error occurs in the command response. CRC check is enabled through the MMCHS_CMD[19] CCCE bit.
MMCHS_STAT[16] CTO	MMCHS_IE[16] CTO_ENABLE	MMC_SD_i_INT	Command time-out error. This bit is set automatically when no response is received within 64 clock cycles from the end bit of the command. For commands that reply within five clock cycles, the time-out is still detected at 64 clock cycles.
MMCHS_STAT[15] ERRI	MMCHS_IE[15] ERRI_ENABLE	MMC_SD_i_INT	Error interrupt. If any of the bits in the error interrupt status register (MMCHS_STAT[31-16]) are set, this bit is set to 1.

Table 11-1828. Events (continued)

Event Flag	Event Mask	Map to	Description
MMCHS_STAT[10] BSR	MMCHS_IE[10] BSR_ENABLE	MMC_SD_i_INT	Boot status received interrupt. This bit is set automatically when the MMCHS_CON[18] BOOT_CF0 bit is set to 1h or 2h and a boot status is received on the dat0 line. This interrupt is useful only for the MMC card.
MMCHS_STAT[8] CIRQ	MMCHS_IE[8] CIRQ_ENABLE	MMC_SD_i_INT	Card interrupt. This bit is used only for SD and CE-ATA cards. In 1-bit mode, the interrupt source is asynchronous . In 4-bit mode, the interrupt source is sampled during the interrupt cycle. In CE-ATA mode, the interrupt source is detected when the card drives the CMD line to 0 during one cycle after data transmission end.
MMCHS_STAT[7] CREM	MMCHS_IE[7] CREM_ENABLE	MMC_SD_i_INT	Card removal. This bit is set automatically when MMCHS_PSTATE[16] CINS changes from 1 to 0.
MMCHS_STAT[6] CINS	MMCHS_IE[6] CINS_ENABLE	MMC_SD_i_INT	Card insertion. This bit is set automatically when MMCHS_PSTATE[16] CINS changes from 0 to 1.
MMCHS_STAT[5] BRR	MMCHS_IE[5] BRR_ENABLE	MMC_SD_i_INT	Buffer read ready. This bit is set automatically during a read operation to the card (see class 2 block-oriented read commands) when one block specified by the MMCHS_BLK[10-0] BLEN bit field is completely written in the buffer. It indicates that the memory card has filled out the buffer and that the LH must empty the buffer by reading it.
MMCHS_STAT[4] BWR	MMCHS_IE[4] BWR_ENABLE	MMC_SD_i_INT	Buffer write ready. This bit is set automatically during a write operation to the card (see class 4 block-oriented write command) when the host can write a complete block as specified by the MMCHS_BLK[10-0] BLEN bit field. It indicates that the memory card has emptied one block from the buffer and that the LH can write one block of data into the buffer.
MMCHS_STAT[3] DMA	MMCHS_IE[3] DMA_ENABLE	MMC_SD_i_INT	DMA interrupt. This status is set when an interrupt is required in the ADMA instruction and after the data transfer completes.
MMCHS_STAT[2] BGE	MMCHS_IE[2] BGE_ENABLE	MMC_SD_i_INT	Block gap event. When a stop at the block gap is requested (MMCHS_HCTL[16] SBGR), this bit is automatically set when transaction is stopped at the block gap during a read or write operation.

Table 11-1828. Events (continued)

Event Flag	Event Mask	Map to	Description
MMCHS_STAT[1] TC	MMCHS_IE[1] TC_ENABLE	MMC_SD_i_INT	Transfer completed. This bit is always set when a read/write transfer is complete or between two blocks when the transfer is stopped because of a stop at block gap request (MMCHS_HCTL[16] SBGR). • In read mode: This bit is automatically set when a read transfer completes (MMCHS_PSTATE[9] RTA). • In write mode: This bit is automatically set when the DAT line use completes (MMCHS_PSTATE[2] DLA).
MMCHS_STAT[0] CC	MMCHS_IE[0] CC_ENABLE	MMC_SD_i_INT	Command complete. This bit is set when a 1-to-0 transition occurs in the register command inhibit (MMCHS_PSTATE[0] CMDI). If the command is a type for which no response is expected, then the command complete interrupt is generated at the end of the command. A command time-out error (MMCHS_STAT[16] CTO) has higher priority than command complete (MMCHS_STAT[0] CC). If a response is expected but none is received, then a command time-out error is detected and signaled, instead of the command complete interrupt.

NOTE: To send an interrupt request to the MMC_SD_i_INT line, the mask/unmask bit must be set in the MMCi.MMCHS_IE and MMCi.MMCHS_ISE registers.

The MMCi host controller supports interrupt-driven operation and polling.

11.12.4.4.1 Interrupt-Driven Operation

An interrupt-enable bit must be set in the MMCi.MMCHS_IE register to enable the module internal source of interrupt.

When an interrupt event occurs, the single interrupt line is asserted and the LH must:

1. Read the MMCi.MMCHS_STAT register to identify which event occurred.
2. Write 1 into the corresponding bit of the MMCi.MMCHS_STAT register to clear the interrupt status and release the interrupt line (if a read is done after this write, this returns 0).

NOTE: In the MMCi.MMCHS_STAT register, the card interrupt (CIRQ) and error interrupt (ERRI) bits cannot be cleared.

The MMCi.MMCHS_STAT[8] CIRQ status bit must be masked by disabling the MMCi.MMCHS_IE[8] CIRQ_ENABLE bit (set to 0h), and then the interrupt routine must clear the SDIO interrupt source in the SDIO card common control register (CCCR).

The MMCi.MMCHS_STAT[15] ERRI bit is automatically cleared when all status bits in the MMCi.MMCHS_STAT register (bits 31 through 16) are cleared.

11.12.4.4.2 Polling

When the interrupt capability of an event is disabled in the MMCi.MMCHS_ISE register, the interrupt line is not asserted:

- Software can poll the status bit in the MMCi.MMCHS_STAT register to detect when the corresponding event occurs.
- Writing 1 into the corresponding bit of the MMCi.MMCHS_STAT register clears the interrupt status and does not affect the interrupt line state.

NOTE: See the previous note concerning clearing of the CIRQ and ERRI bits.

11.12.4.4.3 Asynchronous Interrupt

Asynchronous interrupt is defined in *SDIO Card Specification version 3.00, part E*. This interrupt is effective in 4-bit mode and is generated without SD clock. Asynchronous interrupt period is defined in the synchronous interrupt period after the last data block and until a next command is received.

Asynchronous interrupt period in a multiple block write operation.

If MMCHS_CAPA[29] AIS is set to 0, writing to MMCHS_AC12[30] AI_ENABLE is ignored. This bit is set to 0. A synchronous interrupt period starts two clocks after the last data block. If MMCHS_AC12[30] AI_ENABLE is set to 1, the asynchronous interrupt period starts four clocks after the start of the synchronous interrupt period. Four clocks after the start bit of the next command, the asynchronous interrupt period ends and goes back to the synchronous interrupt period.

11.12.4.5 DMA Modes

Two DMA management modes can be used to load data from memory to the internal buffer of the controller (or vice versa). These modes are exclusive and depend on the module integration.

- DMA master mode:

  DMA master mode is selected by setting the MMCHS_CON[20] DMA_MNS bit to 1. In this case, the controller has direct access to data using a specific algorithm called ADMA2 (prevents the system from being interrupted). Data are exchanged using the master interface interconnect, which supports burst accesses to maximize throughput.

  NOTE: This mode is supported only by modules connected to the master interface interconnect. For more information and/or to check the value of the MMCHS_HL_HWINFO[0] MADMA_EN bit. Refer to this register for a more detailed description.

  This mode is available for modules MMC0 and MMC1.

- DMA slave mode:

  DMA slave mode is selected by setting the MMCHS_CON[20] DMA_MNS bit to 0. In this case, the controller is slave on the DMA transaction managed by two separated requests (MMC_SD_i_TX_EVT and MMC_SD_i_RX_EVT).

  NOTE: This mode is the only mode supported by modules that are not connected to the master interface interconnect (regardless of the value of the MMCHS_CON[20] DMA_MNS bit). For more information and/or to check the value of the MMCHS_HL_HWINFO[0] MADMA_EN bit. Refer to this register for a more detailed description.

  This mode is available for modules MMC0 and MMC1.

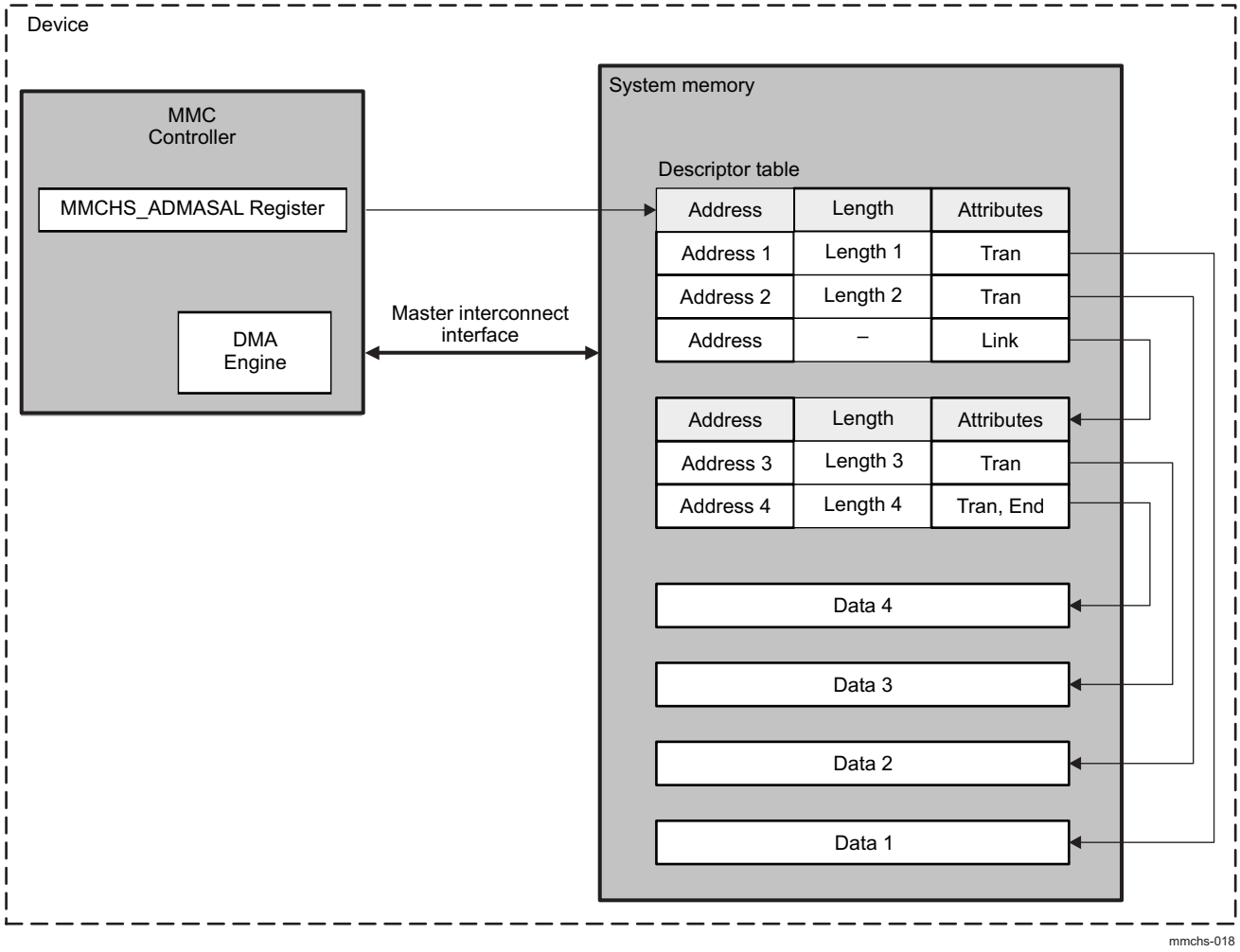
11.12.4.5.1 Master DMA Operations

The MMCi host controller has direct access to the internal data. This feature is called advanced DMA (ADMA). It follows a specific algorithm (ADMA2) defined by an instruction in memory that starts at an address previously loaded in the MMCHS_ADMASAL register before any data command issued to the MMC card. Only 32-bit address spacing is supported by the controller for data storage.

NOTE: This mode is supported only by modules connected to the master interface interconnect. For more information and/or to check the value of the MMCHS_HL_HWINFO[0] MADMA_EN bit. Refer to this register for a more detailed description.

These instructions must be loaded by software in a 32-bit-addressed descriptor table in system memory, as shown in Figure 11-833. In this case the MMCHS_ADMASAL register is used as the program address pointer.

Figure 11-833. ADMA Block Diagram Overview



11.12.4.5.1.1 Descriptor Table Description

Each descriptor line contains an address, a length, and attributes fields. The attributes define which operation will be processed. Every descriptor line is a 64-bit-wide register that is fetched in the controller using the master interface interconnect, and requires two 32-bit accesses to memory.

Table 11-1829 shows the structure of a descriptor line.

Table 11-1829. Descriptor Line Overview

Address Field		Length		Reserved		Attributes					
63	32	31	16	15	6	5	4	3	2	1	0
32-bit address		16-bit length		0h		Act2	Act1	0	Int	Ent	Valid

The attribute of the descriptor line is divided into two parts:

- Attributes[5:4]: The action to be processed by the ADMA engine
- Attributes[3:0]: Additional parameters characterizing the behavior of the ADMA engine

Table 11-1830 describes the available actions of a descriptor line.

Table 11-1830. Available Actions of a Descriptor Line

Act2	Act1	Symbol	Comment	Operation
0	0	Nop	No operation	Do not execute the current line and go to the next line.
0	1	Rsv	Reserved	Reserved action. Behaves the same as the Nop command.
1	0	Tran	Transfer data	Transfer data of one descriptor line.
1	1	Link	Link descriptor	Link to another descriptor.

Table 11-1831 describes the additional parameters of a descriptor line.

Table 11-1831. Additional Parameters of a Descriptor Line

Bit	Description
Valid	Valid = 1 indicates that this descriptor line is effective. If Valid = 0, an ADMA error interrupt is generated and the ADMA is stopped. This prevents runaways.
End	End = 1 indicates the end of a descriptor. The transfer-complete interrupt is generated when the operation of the descriptor line is complete.
Int	Int = 1 generates a DMA interrupt when the operation of the descriptor line is complete.

11.12.4.5.1.2 Requirements for Descriptors

The following sections discuss restrictions and tips on how to correctly configure the descriptors to be used by the ADMA engine.

11.12.4.5.1.2.1 Data Length

There are three requirements to program descriptors:

- The minimum unit of address is 4 bytes.
- The maximum data length of each descriptor line is less than 64 KB.
- Total length = $Length_1$ + $Length_2$+ $Length_3$ + ... + $Length_n$ must be a multiple of the block size.

If the total length of a descriptor is not a multiple of the block size, data transfer with the ADMA engine may not have been terminated. In this case, the controller returns a data time-out event and the transfer is aborted.

The block count register (the MMCHS_BLK[31-16] NBLK bit field) is defined as 16 bits and limits data transfers to a maximum of 65,535 blocks. If the ADMA data transfer size is less than or equal to the 65,535-block transfer, the block count register can be used. In this case, the total length of the descriptor table must be equivalent to "block size" by "block count." If the ADMA data transfer is greater than 65,535 blocks, the block count register must be disabled by setting the block count enable bit (MMCHS_CMD[1] BCE bit) to 0. In this case, the length of the data transfer is not designated by the block count but by the descriptor table.

NOTE: The timing for detecting the last block on the SD bus may differ, which affects control of the read transfer active (MMCHS_PSTATE[9] RTA), write transfer active (MMCHS_PSTATE[8] WTA), and DAT line active (MMCHS_PSTATE[2] DLA) bits. In case of a read operation, more blocks than required may be read from the card. The host driver must ignore an out-of-range error if the read operation is for the last block of the memory area.

11.12.4.5.1.2.2 Supported Features

The ADMA engine does not support the suspend/resume function. However, the stop and continue functions are available.

When the stop-at-block-gap request (the MMCHS_HCTL[16] SBGR bit) is set during the ADMA operation, the block gap event interrupt is generated when the ADMA is stopped at block gap (the MMCHS_STAT[2] BGE bit). The host controller must stop the ADMA read operation by using read wait or by stopping the SD clock. While stopping ADMA, SD commands cannot be issued.

11.12.4.5.1.2.3 Error Generation

When an error occurs during an ADMA transfer, the ADMA operation is stopped and the ADMA error interrupt is generated. The ADMA error state field (the MMCHS_ADMAES[1-0] AES bit field) holds the state of the ADMA when it is stopped. Software can identify the erroneous descriptor line by using the following method:

- If the ADMA stopped at ST_FDS state, the ADMA system address register (MMCHS_ADMASAL) points to the erroneous descriptor line.
- If the ADMA stopped at ST_TFR or ST_STOP state, the ADMA system address register points to the descriptor line following the erroneous one.

11.12.4.5.1.3 Advanced DMA Description

The ADMA is a DMA controller embedded in each MMC controller. It can be seen as a small sequencer that fetches a descriptor line and executes the corresponding action. The base address of the descriptor table is stored in the MMCHS_ADMASAL register.

NOTE: Software must write the base address of the descriptor table in the MMCHS_ADMASAL register before the first use of the ADMA engine.

The ADMA program is executed according to descriptor attributes (see Section 11.12.4.5.1.1, *Descriptor Table Description*) and an FSM, as shown in Figure 11-834.

Figure 11-834. ADMA Finite State-Machine

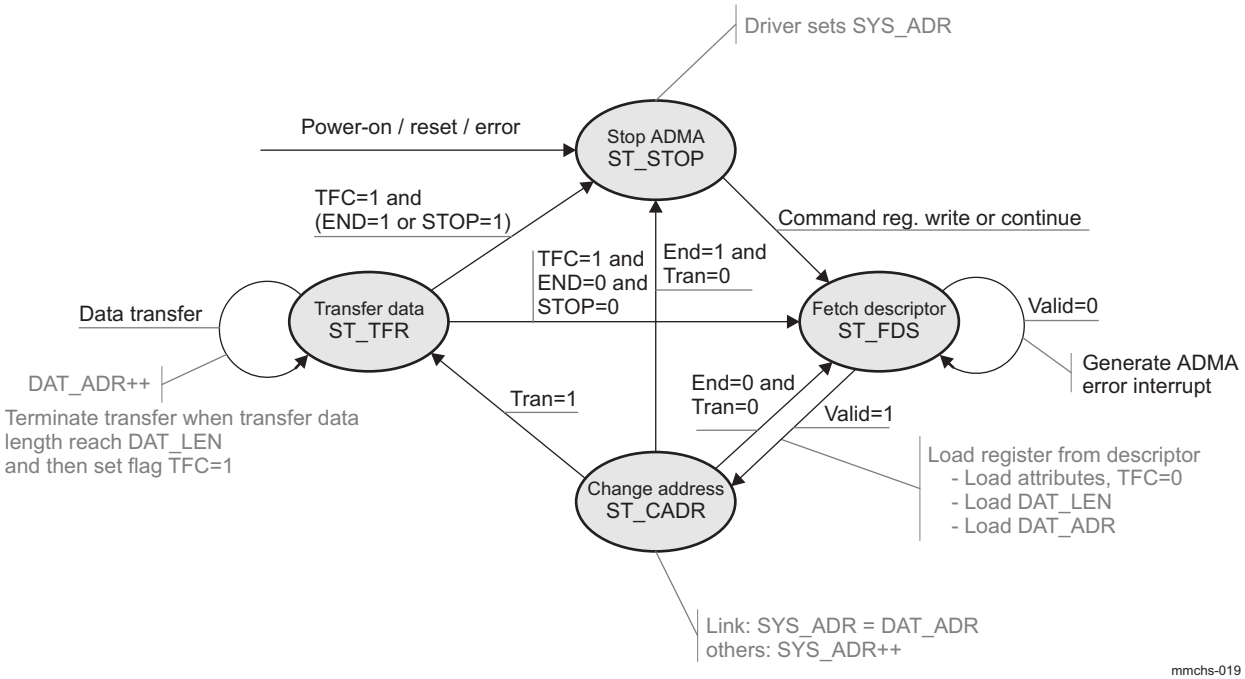


Table 11-1832 describes each state of the ADMA FSM.

Table 11-1832. ADMA2 States Description

State Name	Operation
ST_FDS (fetch descriptor)	ADMA2 fetches a descriptor line and sets parameters in internal registers. It then goes to ST_CADR state.
ST_CADR (change address)	Link operation loads another descriptor address to the ADMA system address register (MMCHS_ADMASAL). In other operations, the ADMA system address register is incremented to point to the next descriptor line. If End = 0, go to ST_FDS state. **NOTE**: ADMA2 does not stop at this state if some errors occur.
ST_TFR (transfer data)	Data transfer of one descriptor line is executed between system memory and the SD card: • If data transfer continues (End = 0), go to ST_FDS state. • If data transfer completes, go to ST_STOP state.
ST_STOP (stop DMA)	ADMA2 stays in this state in the following cases: • After power-on reset (POR) or software reset • All descriptor data transfers are complete. If a new ADMA operation is stated by writing the command register, go to ST_FDS state.

Table 11-1833 gives the description of each symbol used in the ADMA FSM (see Figure 11-834).

Table 11-1833. ADMA FSM Symbol Definition

Symbol	Definition
SYS_ADR	ADMA system address register
SYS_ADR++	Point to next descriptor line
DAT_ADR	Data address register (internal)
DAT_LEN	Data length register (internal)
TFC	Transfer complete flag (internal)
STOP	Stop-at-block-gap request

11.12.4.5.2 Slave DMA Operations

The MMCi host controller can be interfaced with a DMA controller. At the system level, the advantage is to discharge the LH of the data transfers. The module does not support wide DMA access (more than 1024 bytes) for SD cards, as specified in the *SD Card Specification* and *SD Host Controller Standard Specification*.

NOTE: This mode is implied by modules that are not connected to the master interface interconnect (regardless of the value of the MMCHS_CON[20] DMA_MNS bit). For more information and/or to check the value of the MMCHS_HL_HWINFO[0] MADMA_EN bit, see Section 11.12.4.5.1, *Master DMA Operations*.

The DMA request is issued if the following conditions are met:

- The MMCi.MMCHS_CMD[0] DE bit is set to 1 to trigger the initial DMA request (the write must be done when running the data transfer command).
- A command was emitted on the CMD line.
- There is enough space in the buffer of the MMCi host controller to write an entire block (BLEN writes).

11.12.4.5.2.1 DMA Receive Mode

In a DMA block read operation (single or multiple), the request signal MMC_SD_i_RX_EVT is asserted to its active level when a complete block is written in the buffer. The block size transfer is specified in the MMCi.MMCHS_BLK[10-0] BLEN bit field.

MMC_SD_i_RX_EVT is deasserted to its inactive level when the a certain device DMA module reads one word from the buffer.

Only one request is sent per block; the DMA controller can make a 1-shot read access or several DMA bursts, in which case the DMA controller must manage the number of burst accesses, according to the BLEN bit field block size.

New DMA requests are internally masked if the DMA has not read exactly BLEN bytes and a new complete block is not ready. Because DMA accesses are 32-bit accesses, the number of DMA reads is Integer (BLEN/4) + 1.

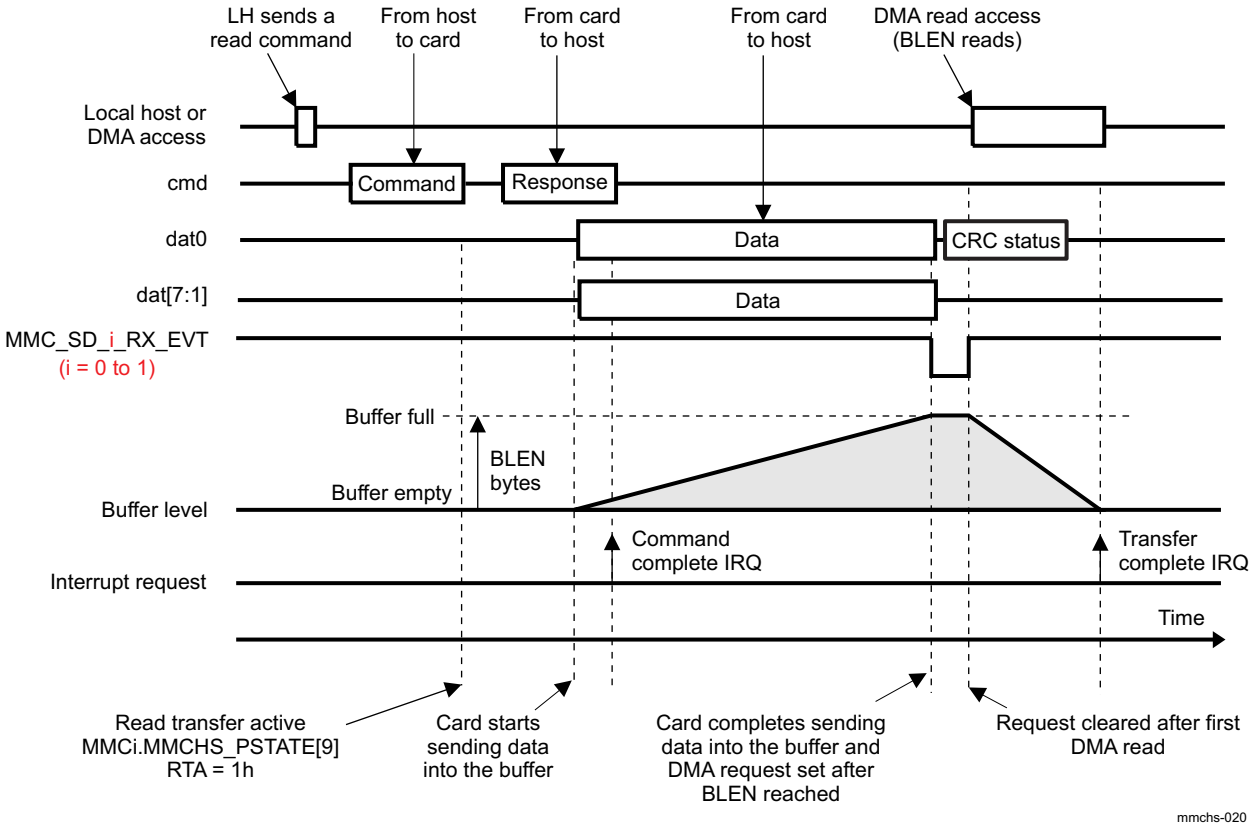
The receive buffer never overflows. In multiple block transfers for block sizes larger than 512 bytes, when the buffer becomes full, the MMCi_CLK clock signal (provided to the card) is momentarily stopped until a certain device DMA or the MPU performs a read access, which reads a complete block in the buffer.

To summarize:

- DMA transfer size = BLEN buffer size in one shot or by burst
- One DMA request per block

Figure 11-835 shows DMA receive mode.

Figure 11-835. DMA Receive Mode



11.12.4.5.2.2 DMA Transmit Mode

In a DMA block write operation (single or multiple), the request signal MMC_SD_i_TX_EVT is asserted to its active level when a complete block is to be written to the buffer. The block size transfer is specified in the MMCi.MMCHS_BLK[10-0] BLEN bit field.

MMC_SD_i_TX_EVT is deasserted to its inactive level when a certain device DMA writes one word to the buffer.

Only one request is sent per block; the DMA controller can make a 1-shot write access or multiple write DMA bursts, in which case the DMA controller must manage the number of burst accesses, according to the BLEN bit field block size.

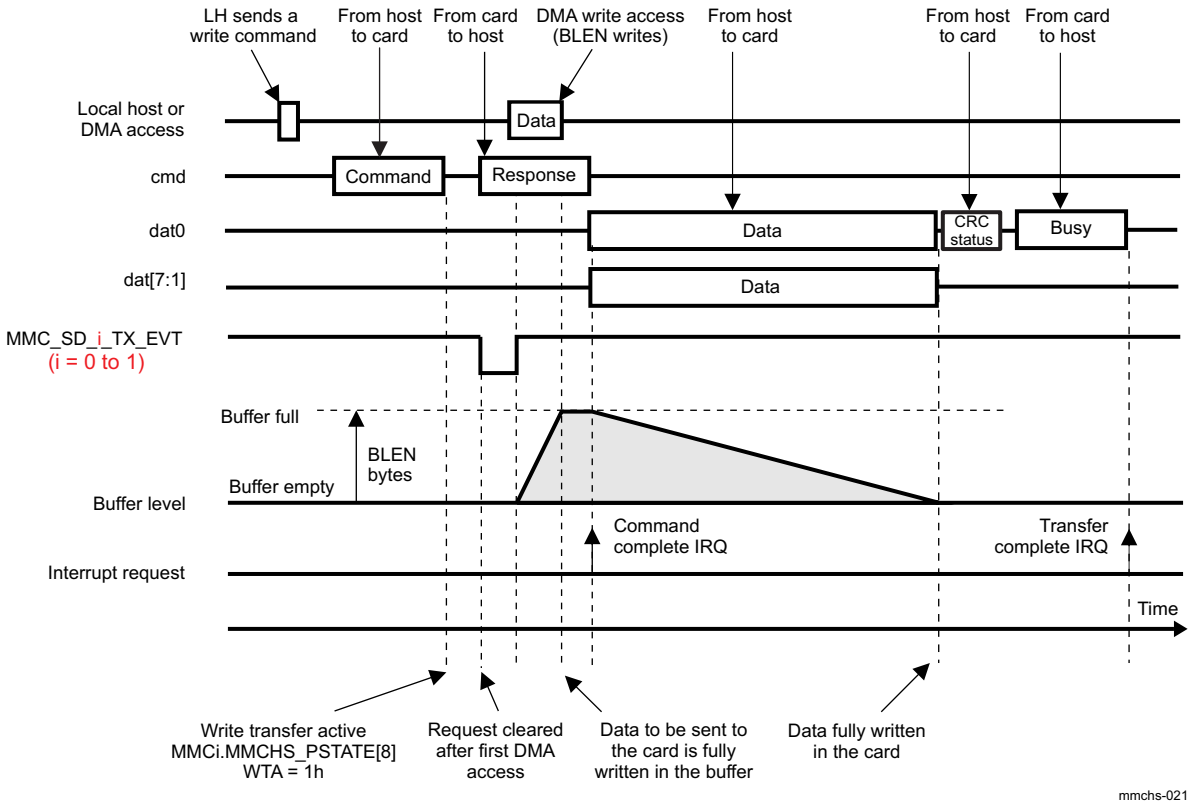
New DMA requests are internally masked if the DMA has not written exactly BLEN bytes (because DMA accesses are 32-bit accesses, the number of DMA reads is Integer (BLEN/4) + 1) and if there is not enough memory space to write a complete block in the buffer.

To summarize:

- DMA transfer size = BLEN buffer size in one shot or by burst
- One DMA request per block

Figure 11-836 shows DMA transmit mode.

Figure 11-836. DMA Transmit Mode



11.12.4.6 Mode Selection

The MMC host controller can be used in two modes: MMC and SD/SDIO (see Section 11.12.2, *MMC/SD Environment*). It has been designed to be the most transparent with the type of card.

The type of the card connected is differentiated by the software initialization procedure. Software identifies the type of card connected during software initialization. For each card type, there are corresponding commands. Some commands are not supported by all cards. For more information, see the *Multimedia Card System Specification*, *SD Memory Card Specifications*, and *SDIO Card Specification, Part E1*.

The purpose of the module is to transfer commands and data to whatever card is connected, respecting the protocol of the connected card.

Writes and reads to the card must respect the appropriate protocol of that card.

11.12.4.7 Buffer Management

11.12.4.7.1 Data Buffer

The MMCi host controller uses a data buffer. This buffer transfers data from one data bus (interconnect) to another data bus (SD or MMC card bus) and vice versa.

The buffer is the heart of the interface and ensures the transfer between the two interfaces (interconnect and the card).

To enhance performance, the data buffer is completed by a prefetch register and a post-write buffer that are not accessible by the host controller.

The read access time of the prefetch register is faster than that of the data buffer. The prefetch register allows data to be read from the data buffer at an increased speed by preloading data into the prefetch register.

The entry point of the prefetch buffer and the post-write buffer is the 32-bit MMCi.MMCHS_DATA register. A write access to the MMCi.MMCHS_DATA register followed by a read access from the MMCi.MMCHS_DATA register corresponds to a write access to the post-write buffer followed by a read access to the prefetch buffer. As a consequence, it is normal that the data of the write access to the MMCi.MMCHS_DATA register and the data of the read access to the MMCi.MMCHS_DATA register are different.

The number of 32-bit accesses to the MMCi.MMCHS_DATA register that are needed to read (or write) a data block with a size of the MMCi.MMCHS_BLK[11-0] BLEN bit field is equal to the rounded up result of BLEN divided by 4.

The maximum block size supported by the host controller is hard-coded in the MMCi.MMCHS_CAPA[17-16] MBL bit field and cannot be changed.

A read access to the MMCi.MMCHS_DATA register is allowed only when the buffer read-enable status is set to 1 (the MMCi.MMCHS_PSTATE[11] BRE bit); otherwise, a bad access (the MMCi.MMCHS_STAT[29] BADA bit) is signaled.

A write access to the MMCi.MMCHS_DATA register is allowed only when the buffer write-enable status is set to 1 (the MMCi.MMCHS_PSTATE[10] BWE bit); otherwise, a bad access (the MMCi.MMCHS_STAT[29] BADA bit) is signaled and the data are not written.

The data buffer has two modes of operation to store and read of the first and second portions of the data buffer:

- When the size of the data block to transfer is less than or equal to MEM_SIZE/2 (in double-buffering), two data transfers can occur at the same time from one data bus to the other data bus, and vice versa. The MMCi host controller uses the two portions of the data buffer in a ping-pong manner so that storing and reading the first and second portions of the data buffer are automatically interchanged from time to time. In this way, data can be read from one portion (for instance, through a DMA read access on the interconnect bus) while data (for instance, from the card) are being stored into the other portion, and vice versa. When BLEN is less than or equal to 200h (that is, less than or equal to 512 bytes), each of the two portions of the buffer that can be used have a size of BLEN (that is, 32 bits × BLEN divided by 4). No more than this total size of 2 × 32 bits × BLEN divided by 4 can be used.

CAUTION

The MMCi.MMCHS_CMD[4] DDIR bit must be configured before a transfer to indicate the direction of the transfer.

Figure 11-837 and Figure 11-838 show the buffer management for a write and a read, respectively.

Figure 11-837. Buffer Management for a Write

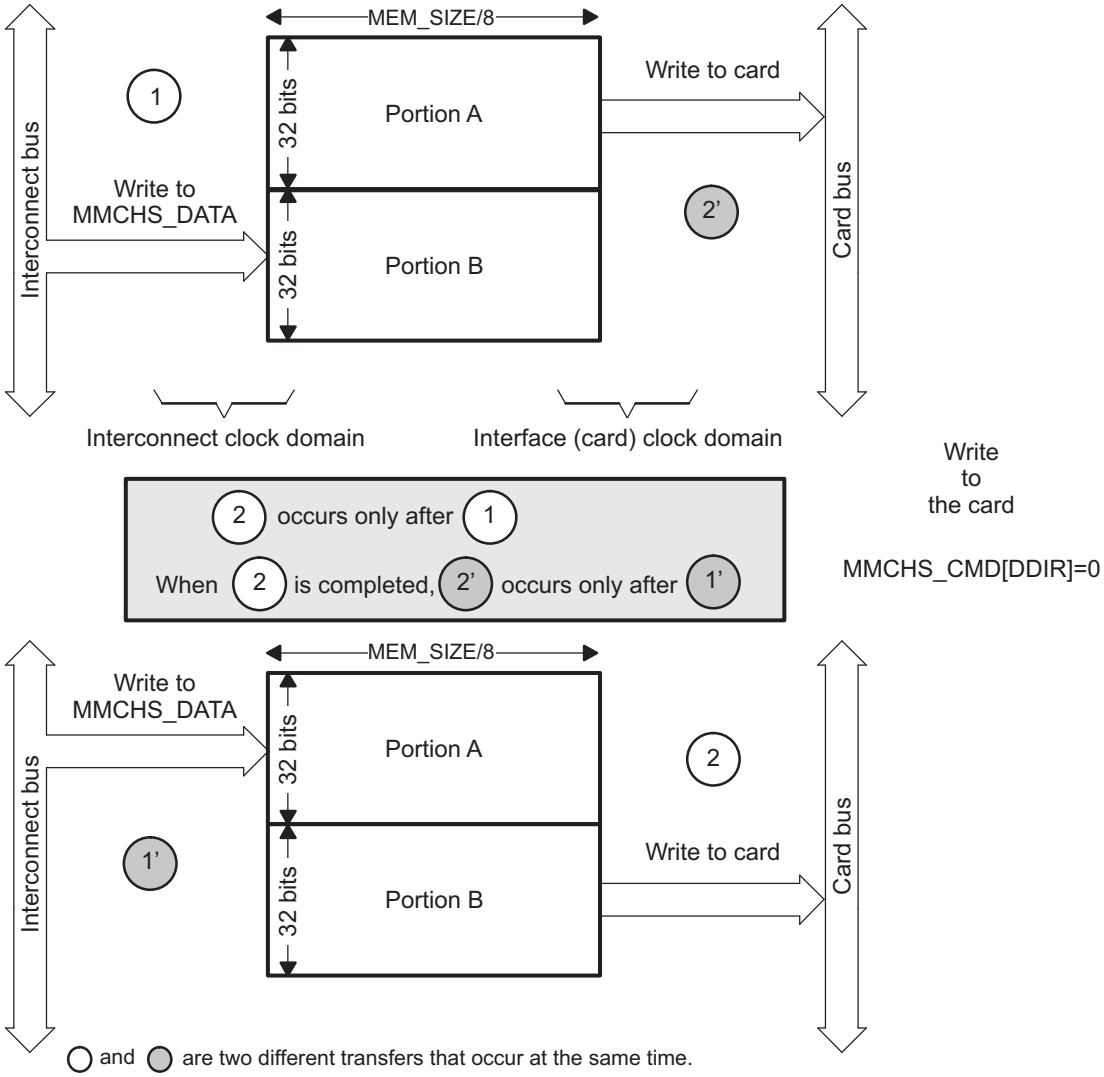
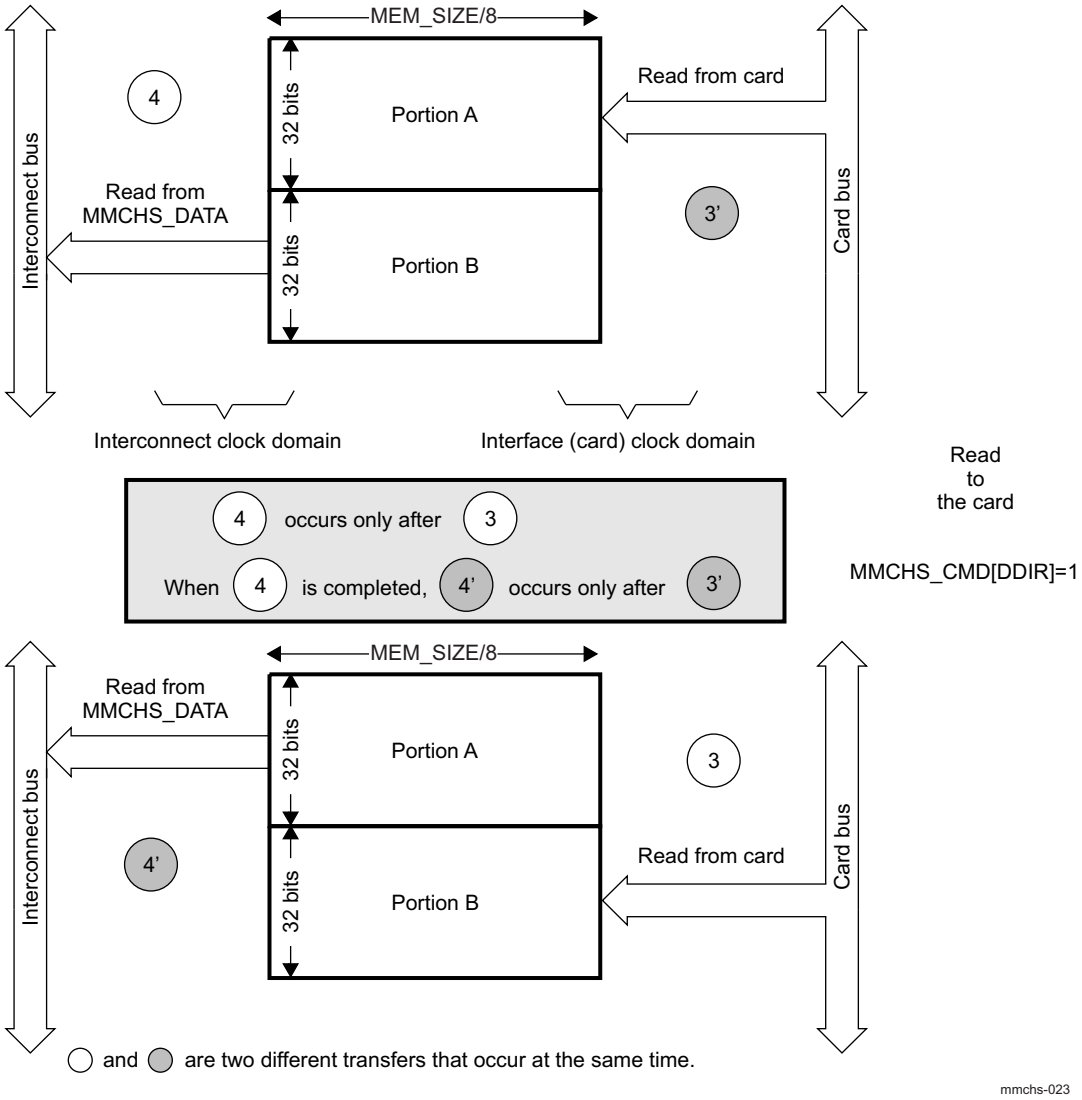


Figure 11-838. Buffer Management for a Read



- When the size of the data block to transfer is larger than MEM_SIZE/2, only one data transfer at a time can occur from one data bus to the other data bus. The MMCi host controller uses the entire data buffer as a single portion.

  In this mode, a bad access (the MMCi.MMCHS_STAT[29] BADA bit) is signaled when two data transfers occur at the same time from one data bus to the other data bus, and vice versa.

11.12.4.7.1.1 Memory Size, Block Length, and Buffer-Management Relationship

The maximum block length and buffer management that can be targeted by the system depend on the memory depth setting (see Table 11-1834).

NOTE: Double-buffering is always the buffer management for large memory depth.

Table 11-1834. Memory Size, BLEN, and Buffer Relationship

Memory Size (MMCHS_HL_HWINFO[5-2] MEM_SIZE in bytes)	512	1024
Maximum block length supported	512	1024

Table 11-1834. Memory Size, BLEN, and Buffer Relationship (continued)

Memory Size (MMCHS_HL_HWINFO[5-2] MEM_SIZE in bytes)	512	1024
Double-buffering for maximum block length	N/A	BLEN <= 512
Single-buffering for block length	BLEN <= 512	512 < BLEN <= 1024

NOTE: For single-buffering management, throughput on the MMC bus interface deteriorates in multiblock transfers, because the controller must wait for the filling or emptying of the buffer between each block transfer on the MMC bus. The clock is maintained on write MMC transfers (the MMCHS_CMD[4] DDIR bit is 0) and halted on read MMC transfers (the MMCHS_CMD[4] DDIR bit is 1).

11.12.4.7.1.2 Data Buffer Status

The data buffer status is defined in the following interrupt status register and status register:

- Interrupt status registers:
  - MMCi.MMCHS_STAT[29] BADA: Bad access to data space
  - MMCi.MMCHS_STAT[5] BRR: Buffer read ready
  - MMCi.MMCHS_STAT[4] BWR: Buffer write ready
- Status registers:
  - MMCi.MMCHS_PSTATE[11] BRE: Buffer read enable
  - MMCi.MMCHS_PSTATE[10] BWE: Buffer write enable

11.12.4.8 Transfer Process

The process of a transfer depends on the type of command. It can be with or without a response, and with or without data.

11.12.4.8.1 Different Types of Commands

Different types of commands are specific to the MMC, SD, and SDIO cards. For more information, see the *Multimedia Card System Specification*, *SD Memory Card Specifications*, *SDIO Card Specification, Part E1*; or *the SD Card Specification, Part A2, SD Host Controller Standard Specification*.

11.12.4.8.2 Different Types of Responses

Different types of responses are specific to the eMMC, SD, and SDIO cards. For more information, see the *Multimedia Card System Specification*; *SD Memory Card Specifications*; *SDIO Card Specification, Part E1*, or the *SD Card Specification, Part A2, SD Host Controller Standard Specification*.

Table 11-1835 describes how the eMMC, SD, and SDIO responses are stored in the MMCHS_RSPxx registers.

Table 11-1835. MMC, SD, SDIO Responses in the MMCHS_RSPxx Registers

Type of Response	Response Field	Response Register
R1, R1b (normal response), R3, R4, R5, R5b, R6, R7	RESP[39:8](1)	MMCHS_RSP10[31-0]
R1b (Auto CMD12 response), R1(Auto CMD23 response)	RESP[39:8](1)	MMCHS_RSP76[31-0]
R2	RESP[127:0](1)	MMCHS_RSP76[31-0] MMCHS_RSP54[31-0] MMCHS_RSP32[31-0] MMCHS_RSP10[31-0]

(1) RESP refers to the command response format described in the specifications mentioned.

When the host controller modifies part of the MMCHS_RSPxx registers, it preserves the unmodified bits.

The host controller stores the Auto CMD12 response in the MMCHS_RSP76[31-0] register because the host controller may execute multiple block data transfers on the DATA line concurrently with a command. This allows the host controller to avoid overwriting the response of Auto CMD12 with the command response stored in the MMCHS_RSP10 register, and vice versa.

While executing Auto CMD23 the response of CMD23 is saved in the MMCHS_RSP76[31-0] register and the response of multiple-block read and write command is saved in the MMCHS_RSP10 register. The response error of CMD23 is indicated in the MMCHS_AC12 register, bits [7-0].

11.12.4.9 Transfer or Command Status and Errors Reporting

Flags in the MMCi host controller show the status of communication with the card:

- A time-out (of a command, data, or response)
- A CRC error

Error conditions generate interrupts. For more information, see Table 11-1836 and the register description.

Table 11-1836. CC and TC Values Upon Error Detected

Error Hold in MMCi.MMCHS_STAT		CC	TC	Comments
29	BADA			No dependency with CC or TC BADA is related to the MMCHS_DATA register accesses. Its assertion does not depend on the ongoing transfer.
28	CERR	1		CC is set upon CERR.
22	DEB		1	TC is set upon DEB.
21	DCRC		1	TC is set upon DCRC.
20	DTO			DTO and TC are mutually exclusive. DCRC and DEB cannot occur with DTO.
19	CIE	1		CC is set upon CIE.
18	CEB	1		CC is set upon CEB.
17	CCRC	1		CC can be set upon CCRC. See CTO comment.
16	CTO			CTO and CC are mutually exclusive. CIE, CEB, and CERR cannot occur with CTO. CTO can occur at the same time as CCRC: It indicates a command abort due to contention on the CMD line. In this case no CC appears.

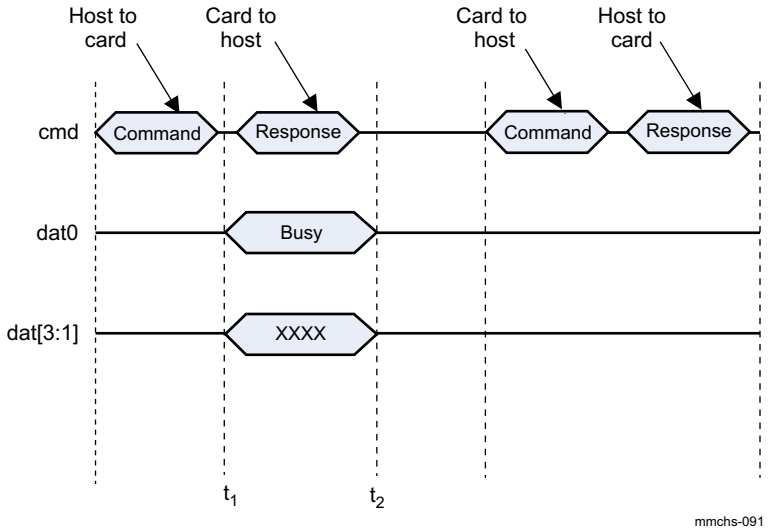
A MMCHS_STAT[20] DTO event can be asserted in the following conditions:

- Busy time-out for R1b, R5b response type
- Busy time-out after write CRC status
- Write CRC status time-out
- Read data time-out
- Boot acknowledge time-out

11.12.4.9.1 Busy Time-Out for R1b, R5b Response Type

Figure 11-839 shows the DTO event condition asserted when there is a busy time-out for Rb1, R5b response.

Figure 11-839. Busy Time-Out for R1b, R5b Response Type



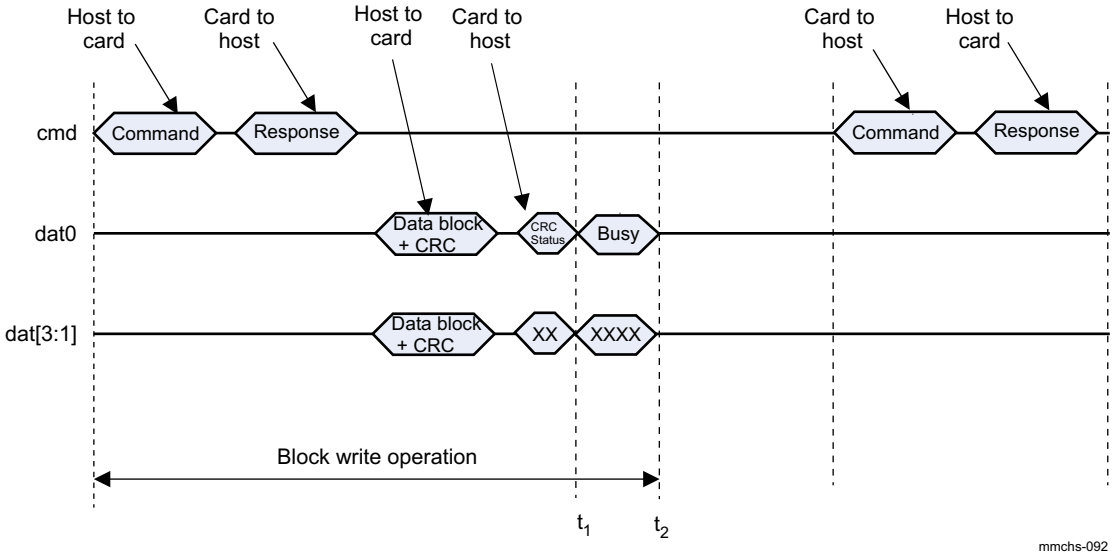
t_1 – Data time-out counter is loaded and starts after R1b, R5b response type.

t_2 – Data time-out counter stops and if it is 0, the MMCHS_STAT[20] DTO bit is generated.

11.12.4.9.2 Busy Time-Out After Write CRC Status

Figure 11-840 shows the DTO event condition asserted when there is a busy time-out after write CRC status.

Figure 11-840. Busy Time-Out After Write CRC Status



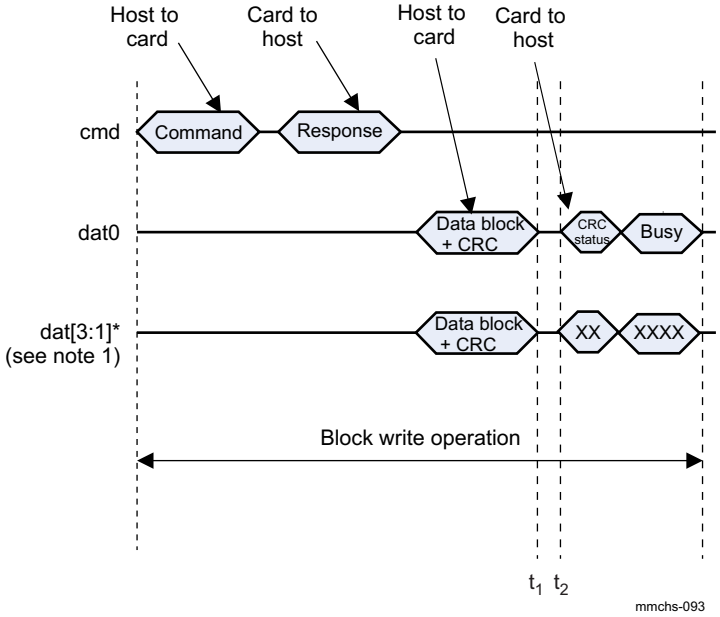
t_1 – Data time-out counter is loaded and starts after CRC status.

t_2 – Data time-out counter stops and if it is 0, the MMCHS_STAT[20] DTO bit is generated.

11.12.4.9.3 Write CRC Status Time-Out

Figure 11-841 shows the DTO event condition asserted when there is a write CRC status time-out.

Figure 11-841. Write CRC Status Time-Out



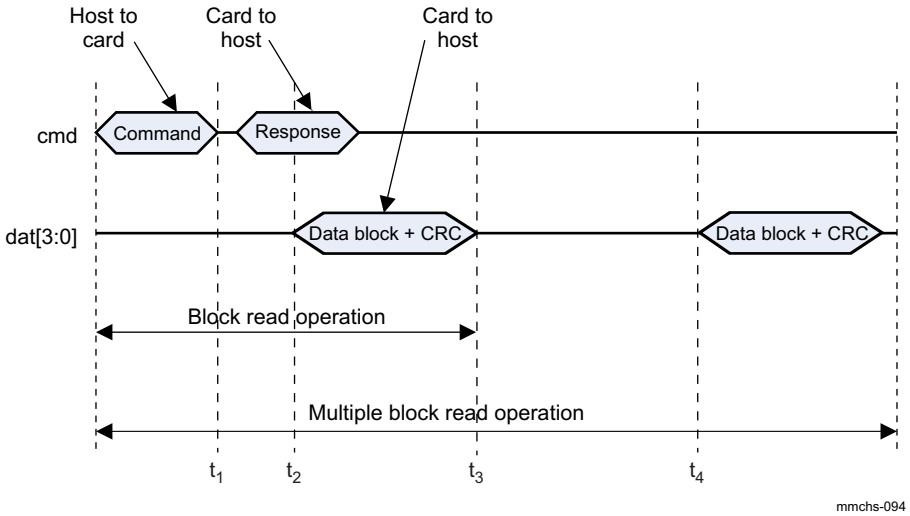
t_1 – Data time-out counter is loaded and starts after data block + CRC.

t_2 – Data time-out counter stops and if it is 0, the MMCHS_STAT[20] DTO bit is generated.

11.12.4.9.4 Read Data Time-Out

Figure 11-842 shows the DTO event condition asserted when there is a read data time-out.

Figure 11-842. Read Data Time-Out



t_1 – Data time-out counter is loaded and starts after command transmission.

t_2 – Data time-out counter stops and if it is 0, the MMCHS_STAT[20] DTO bit is generated.

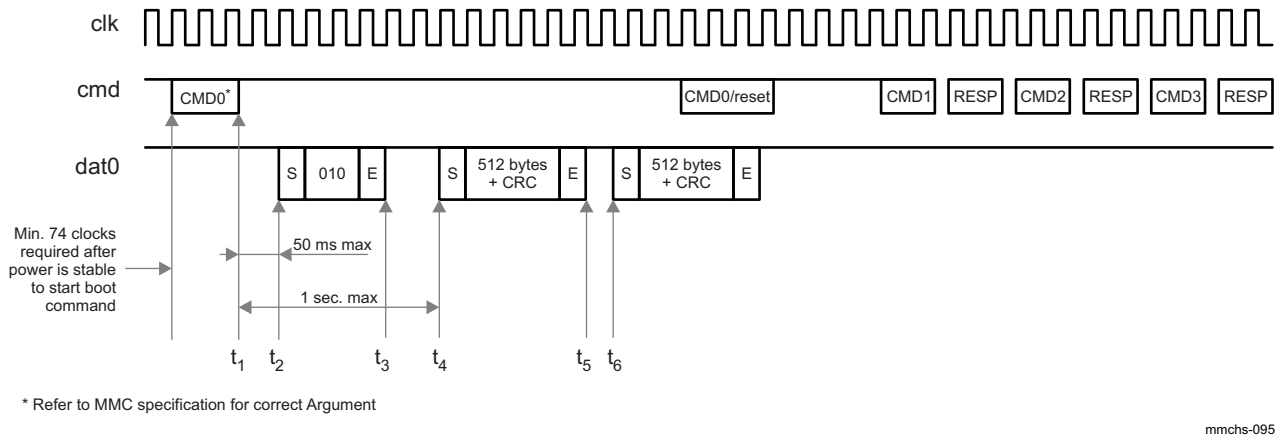
t_3 – Data time-out counter is loaded and starts after data block + CRC transmission.

t_4 – Data time-out counter stops and if it is 0, the MMCHS_STAT[20] DTO bit is generated.

11.12.4.9.5 Boot Acknowledge Time-Out

Figure 11-843 shows the DTO event condition asserted when there is a boot acknowledge time-out and CMD0 is used.

Figure 11-843. Boot Acknowledge Time-Out When Using CMD0



t_1 – Data time-out counter is loaded and starts after CMD0.

t_2 – Data time-out counter stops and if it is 0, the MMCHS_STAT[20] DTO bit is generated.

t_3 – Data time-out counter is loaded and starts.

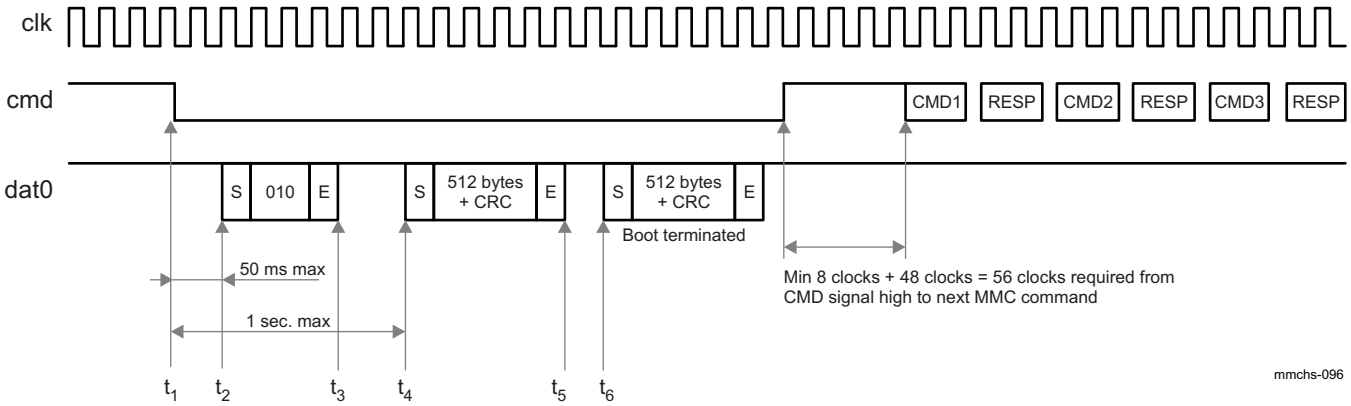
t_4 – Data time-out counter stops and if it is 0, the MMCHS_STAT[20] DTO bit is generated.

t_5 – Data time-out counter is loaded and starts after data + CRC transmission.

t_6 – Data time-out counter stops and if it is 0, the MMCHS_STAT[20] DTO bit is generated.

Figure 11-844 shows the DTO event condition asserted when there is a boot acknowledge time-out when the CMD line is tied to 0.

Figure 11-844. Boot Acknowledge Time-Out When CMD Line Tied to 0



t_1 – Data time-out counter is loaded and starts after the CMD line is tied to 0.

t_2 – Data time-out counter stops and if it is 0, the MMCHS_STAT[20] DTO bit is generated.

t_3 – Data time-out counter is loaded and starts.

t_4 – Data time-out counter stops and if it is 0, the MMCHS_STAT[20] DTO bit is generated.

t_5 – Data time-out counter is loaded and starts after data + CRC transmission.

t_6 – Data time-out counter stops and if it is 0, the MMCHS_STAT[20] DTO bit is generated.

11.12.4.10 Transfer Stop

Whenever a transfer is initiated, the transmission can be stopped before it finishes. Several cases are possible, depending on the transfer type:

- Multiple-block-oriented transfers (transfer length is known)
- Continuous stream transfers (transfer has an infinite length)

NOTE: Because the MMCi controller manages transfers based on a block granularity, the buffer accepts a block only if there is enough space to store it completely. Consequently, if a block is pending in the buffer, no command is sent to the card because the card clock will be shut off by the controller.

The MMCi controller includes three features that make a transfer stop more convenient and easier to manage:

- Auto CMD12/Auto CMD23 (for eMMC and SD only):

  Auto CMD12/Auto CMD23 feature is enabled by setting the MMCi.MMCHS_CMD[3-2] ACEN bit field to 1h or to 2h respectively (this setting is relevant for a MMC/SD transfer with a known number of blocks to transfer). When the Auto CMD12/Auto CMD23 feature is enabled, the MMCi controller automatically issues a CMD12/CMD23 command when the expected number of blocks is exchanged.
- Stop at block gap:

  This feature is enabled by setting the MMCi.MMCHS_HCTL[16] SBGR bit to 1h. When enabled, this capability holds the transfer on until the end of a block boundary. If a stop transmission is needed, software can use this pause to send a CMD12 to the card.
- ADMA mode:

  For ADMA-capable modules (MMC0 and MMC1) (for more information, see Section 11.12.4.5, *DMA Modes*), the last instruction can stop the transfer (the END bit is enabled in the descriptor line).

NOTE: For eMMC and SD cards, the stop-at-block-gap feature is not supported in read mode.

For SDIO cards, this setting can be supported in read mode if the card has read-wait capability.

Table 11-1837 shows the common way to stop a transfer, indicating the command to send and the features to enable.

Table 11-1837. MMCi Controller Transfer Stop Command Summary

		Write Transfer	Read Transfer
		SD/MMC	SD/MMC
Single block		Transfer ends automatically. Wait TC.	Transfer ends automatically. Wait TC.
Multiblocks (finite or infinite)	Before the programmed block boundary	Send CMD12/CMD23. Wait TC.	Send CMD12/CMD23. Wait TC.
	Stop at the end of the transfer (finite transfer only)	Auto CMD12/Auto CMD23 active. Transfer ends automatically. Wait TC.	Auto CMD12/Auto CMD23 active. Transfer ends automatically. Wait TC.

NOTE: The MMCi controller sends the stop command to the card on a block boundary, regardless of when the command was written to the controller registers.

11.12.4.11 Output Signals Generation

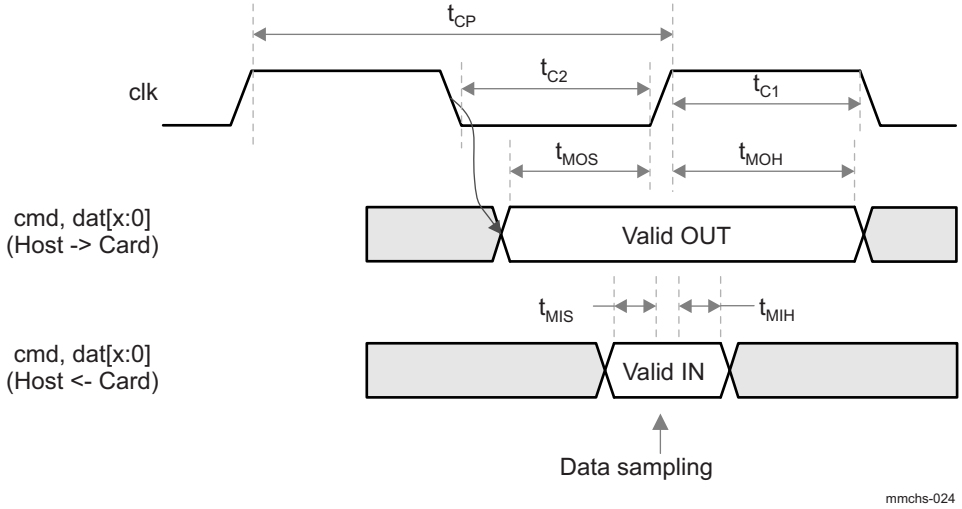
The MMC output signals can be driven on clock falling edge only (see MMCHS_HCTL[2] HSPE bit).

11.12.4.11.1 Generation on Falling Edge of MMC Clock

The controller defaults to this mode to maximize hold timings. In this case, the MMCHS_HCTL[2] HSPE bit is set to 0.

Figure 11-845 shows the output signals of the module when generating from the falling edge of the MMC clock.

Figure 11-845. Output Driven on Falling Edge



11.12.4.12 Card Boot Mode Management

Boot operation mode lets the MMCi host controller read boot data from the connected slave (MMC device) by keeping the CMD line low after power on (or sending CMD0 with a specific argument) before issuing CMD1. The data can be read from the boot area or user area, depending on the register setting.

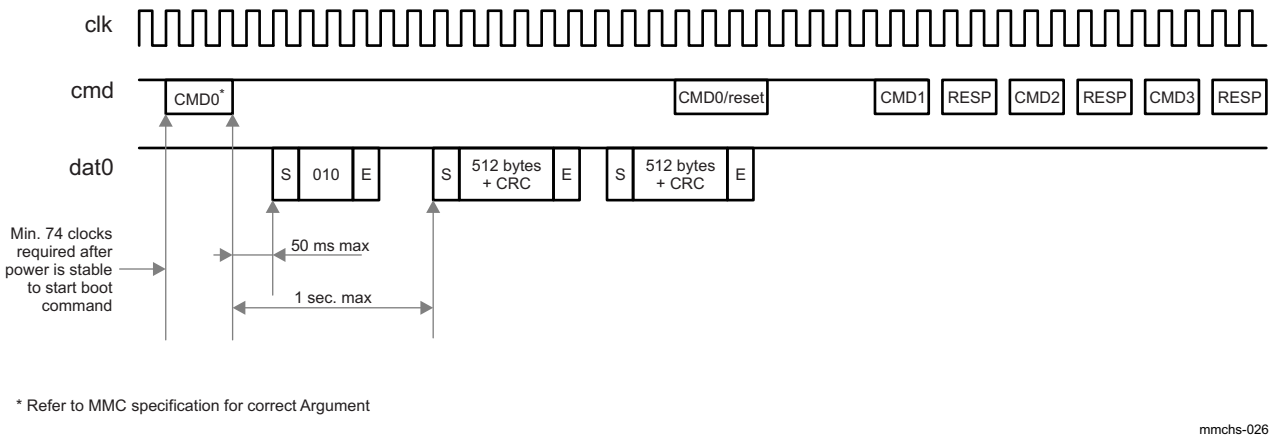
Power-on boot defines a way for the boot code to be accessed by the MMCi host controller without an upper-level software driver, thus speeding the time it takes for a controller to access the boot code.

The two possible ways to issue a boot command (issuing a CMD0 or driving the CMD line to 0 during the whole boot phase) are described in the following sections.

11.12.4.12.1 Boot Mode Using CMD0

Figure 11-846 shows the timing diagram of a boot sequence using CMD0.

Figure 11-846. Boot Mode Using the CMD0 Timing Diagram



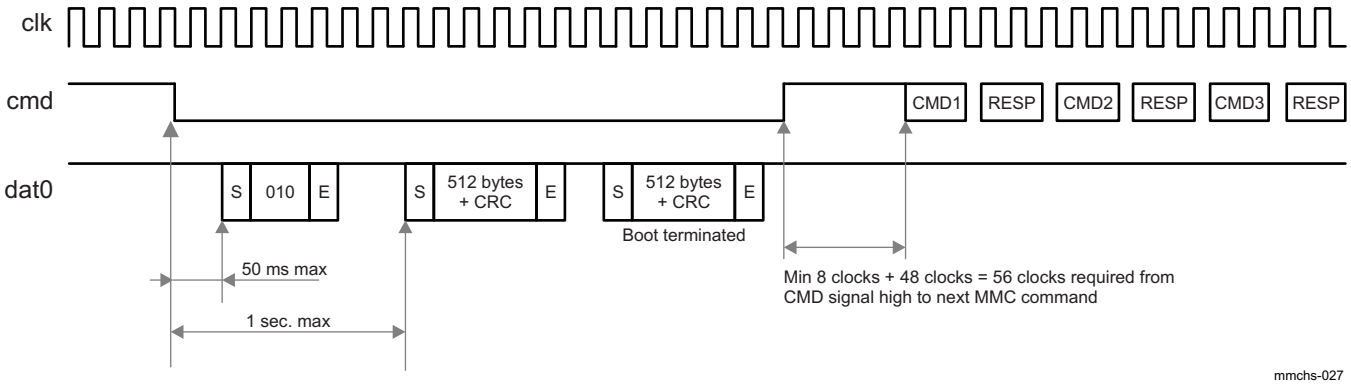
NOTE: Refer to MMC specification for correct Argument.

For more information about how to configure the MMC host controller, see Section 11.12.5.1.2.2.1, *Boot Using the CMD0*.

11.12.4.12.2 Boot Mode With CMD Line Tied to 0

Figure 11-847 shows the timing diagram of a boot sequence with CMD line tied to 0.

Figure 11-847. Boot Mode With CMD Line Tied to 0 Timing Diagram



For more information about how to configure the MMC host controller, see Section 11.12.5.1.2.2.2, *Boot With CMD Line Tied to 0*.

11.12.4.13 MMC CE-ATA Command Completion Disable Management

The MMCi host controller supports CE-ATA features, in particular the detection of the command completion token. When a command that requires a CCS (the MMCHS_CON[12] CEATA bit is set to 1 and the MMCHS_CMD[3-2] ACEN bit field is set to 1h) is launched, the host system is no longer allowed to emit a new command in parallel to the data transfer unless it is a command completion disable token.

The settings to emit a command completion disable token are:

- Set the MMCHS_CON[12] CEATA bit to 1.
- Set the MMCHS_CON[2] HR bit to 1.
- Clear the MMCHS_ARG register.
- Write into the MMCHS_CMD register with the value 0000 0000h.

When a command completion disable token was emitted (that is, the MMCHS_STAT[0] CC bit is received), the host system is again allowed to emit another type of command (for example, a CMD12 to abort transfer).

A critical case can be encountered when CCSD is emitted during the last data block transfer, and the sequence on the command line is sent close to the CCS token sent by the card.

Three possible cases are:

- CCS is received immediately before CCSD is emitted:

  An interrupt CIRQ is generated when CCS is detected, CCSD is transmitted to the card, and then an interrupt CC is generated when CCSD ends. In this case, the card considers the CCSD sequence.
- CCS is not generated or is generated during the CCSD transfer:

  The CCS bit cannot be detected (conflict is not possible because they drive the same level on the command line, and no CIRQ interrupt is generated; a CC interrupt is generated when CCSD ends).
- CCS is generated without CCSD token required:

  Only the interrupt CIRQ is generated when CCS is detected.

11.12.4.14 Test Registers

Test registers are available to comply with the *SD Host Controller Specification*. This feature is useful to generate interrupts manually for driver debugging.

The force event register (MMCHS_FE) is used to control the error status and error interrupt status for Auto CMD12 and Auto CMD23.

The system test register (MMCHS_SYSTEST) is used to control the signals that connect to I/O pins when the module is configured in the system test mode (the MMCHS_CON[4] MODE bit = 1) for boundary connectivity verification.

The MMCHS_HCTL[7] CDSS and MMCHS_HCTL[6] CDTL bits enable manual control of MMCHS_PSTATE[16] CINS and interrupt generating indicated in MMCHS_STAT[7] CREM and MMCHS_STAT[6] CINS.

11.12.4.15 MMC Hardware Status Features

Table 11-1838 describes the MMC hardware status features.

Table 11-1838. MMC Hardware Status Features

Feature	Type	Register/Bit Field	Description
Interrupt flags		See Section 11.12.4.4, *Interrupt Requests.*	
CMD line signal level	Status	MMCHS_PSTATE[24] CLEV	Indicates the level of the command line
DAT lines signal level	Status	MMCHS_PSTATE[23-20] DLEV	Indicates the level of the data lines
Write protect switch pin level	Status	MMCHS_PSTATE[19] WP	Indicates whether the SD card is write protected or not.
Card detect pin level	Status	MMCHS_PSTATE[18] CDPL	Indicates the level of the MMCi_SDCD signal/pad
Card State Stable	Status	MMCHS_PSTATE[17] CSS	Used for testing. Indicates MMCi_SDCD stable state
Card inserted	Status	MMCHS_PSTATE[16] CINS	Indicates whether the SD card is inserted
Buffer read enable	Status	MMCHS_PSTATE[11] BRE	Readable data exists in the buffer.
Buffer write enable	Status	MMCHS_PSTATE[10] BWE	Indicates whether there is enough space in the buffer to write BLEN bytes of data
Read transfer active	Status	MMCHS_PSTATE[9] RTA	Used to detect completion of a read transfer.
Write transfer active	Status	MMCHS_PSTATE[8] WTA	Indicates a write transfer active
Re - Tuning Request	Status	MMCHS_PSTATE[3] RTR	Indicates whether the sampling clock needs re - tuning or not.
Data line active	Status	MMCHS_PSTATE[2] DLA	Indicates whether the data lines are active

Table 11-1838. MMC Hardware Status Features (continued)

Feature	Type	Register/Bit Field	Description
Command Inhibit (DAT lines)	Status	MMCHS_PSTATE[1] DATI	Indicates whether issuing of command using data lines is allowed
Command inhibit (CMD line)	Status	MMCHS_PSTATE[0] CMDI	Indicates whether issuing of command using command line is allowed

Table 11-1839 describes the MMC preset value features.

Table 11-1839. MMC Preset Value Registers

Feature	Type	Register	Description
Preset value register	Status	MMCHS_PVINITSD	Preset Values for Initialization and Default Speed modes
Preset value register	Status	MMCHS_PVHSSDR12	Preset Values for High Speed and SDR12 speed modes
Preset value register	Status	MMCHS_PVSDR25SDR50	Preset Values for SDR25 and SDR50 speed modes(1)
Preset value register	Status	MMCHS_PVSDR104DDR50	Preset Values for SDR104 and DDR50 speed modes(2)

(1) SDR50 mode is not supported.
(2) SDR104 mode is not supported.

11.12.5 MMC/SD Programming Guide

11.12.5.1 Low-Level Programming Models

11.12.5.1.1 Global Initialization

11.12.5.1.1.1 Surrounding Modules Global Initialization

This section identifies the requirements for initializing the surrounding modules when the MMCi module must be used for the first time after a device reset. This initialization of surrounding modules is based on the integration and environment of the MMC modules. For more information, see Section 11.12.3, *MMC/SD Integration*, and Section 11.12.2, *MMC/SD Environment*. Table 11-1840 shows the global initialization of surrounding modules.

Table 11-1840. Global Initialization of Surrounding Modules

Surrounding Modules	Comments
Clock Management	Module interface and functional clocks must be enabled. See Section 5.4, *Clock Management*.
BOOT_CFG	Module-specific pad muxing and configuration must be set in the BOOT_CFG. See Section 5.1, *Control Module (BOOT_CFG)*.
Interrupt Controller	Device INTCs must be configured to enable the interrupt request generation. See Chapter 9, *Interrupts*.

11.12.5.1.1.2 MMC Host Controller Initialization Flow

Table 11-1841 shows the general boot process.

Table 11-1841. MMC Controller Meta Initialization Steps

Step	Access Type	Register/Bit Field/Programming Model	Value
Initialize clocks.		See Section 11.12.5.1.1.2.1, *Enable Interface and Functional Clock for MMC controller*.	
Software reset of the controller.		See Section 11.12.5.1.1.2.2, *MMC Soft Reset Flow*.	
Set module hardware capabilities.		See Section 11.12.5.1.1.2.3, *Set MMC Default Capabilities*.	

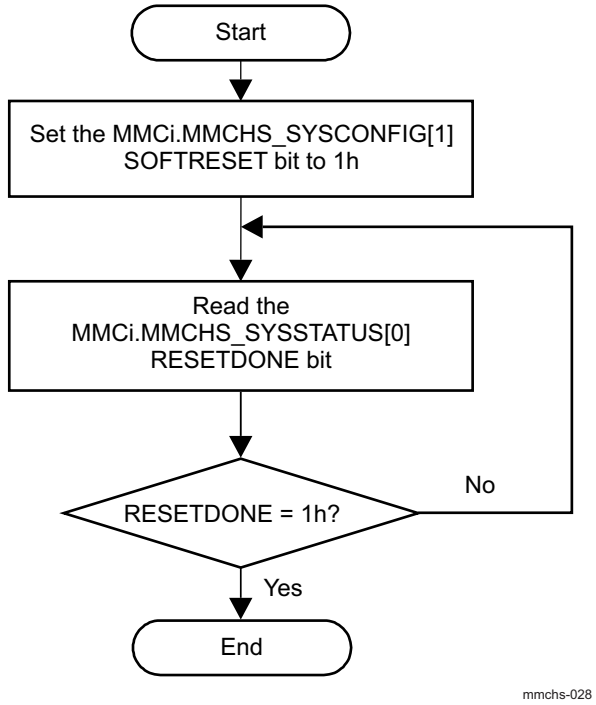
11.12.5.1.1.2.1 Enable Interface and Functional Clock for MMC Controller

Before any MMC register access, the MMC interface and functional clock in the Clock Management registers must be enabled. See Section 5.4, *Clock Management*.

11.12.5.1.1.2.2 MMC Soft Reset Flow

Figure 11-848 shows the soft reset process of the MMC controller.

Figure 11-848. MMC Controller Software Reset Flow



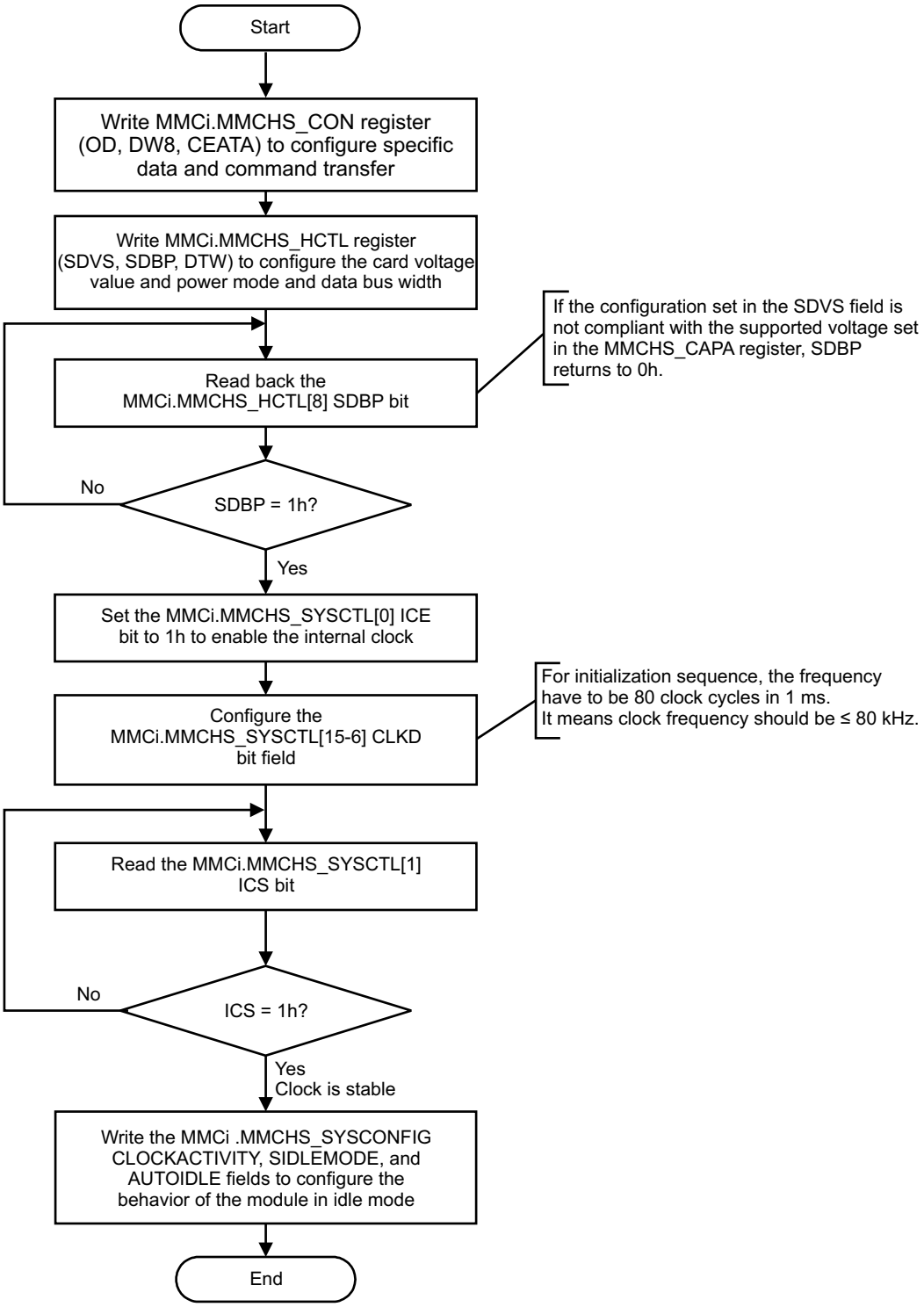
11.12.5.1.1.2.3 Set MMC Default Capabilities

Software must read capabilities (in boot ROM, for example) and is allowed to set (write) the MMCi.MMCHS_CAPA[26-24] and MMCi.MMCHS_CUR_CAPA[23-0] bit fields before the MMC host driver is started.

11.12.5.1.1.2.4 MMC Host and Bus Configuration

Figure 11-849 shows the MMC bus configuration process.

Figure 11-849. MMC Controller Bus Configuration



11.12.5.1.2 Operational Modes Configuration

11.12.5.1.2.1 Basic Operations for MMC Host Controller

The MMC host controller performs data transfers: data to card (referred to as write transfers) and data from card (referred to as read transfers).

The host controller requires transfers to run on a block-by-block basis rather than on a DMA burst size basis. A single DMA request (or block request interrupt) is signaled for each block. Pipelining is supported as long as the block size is less than one half of the memory buffer size.

11.12.5.1.2.1.1 Card Detection, Identification, and Selection

Figure 11-850 and Figure 11-851 show the card detection, identification and selection process.

Figure 11-850. MMC Controller Card Identification and Selection – Part 1

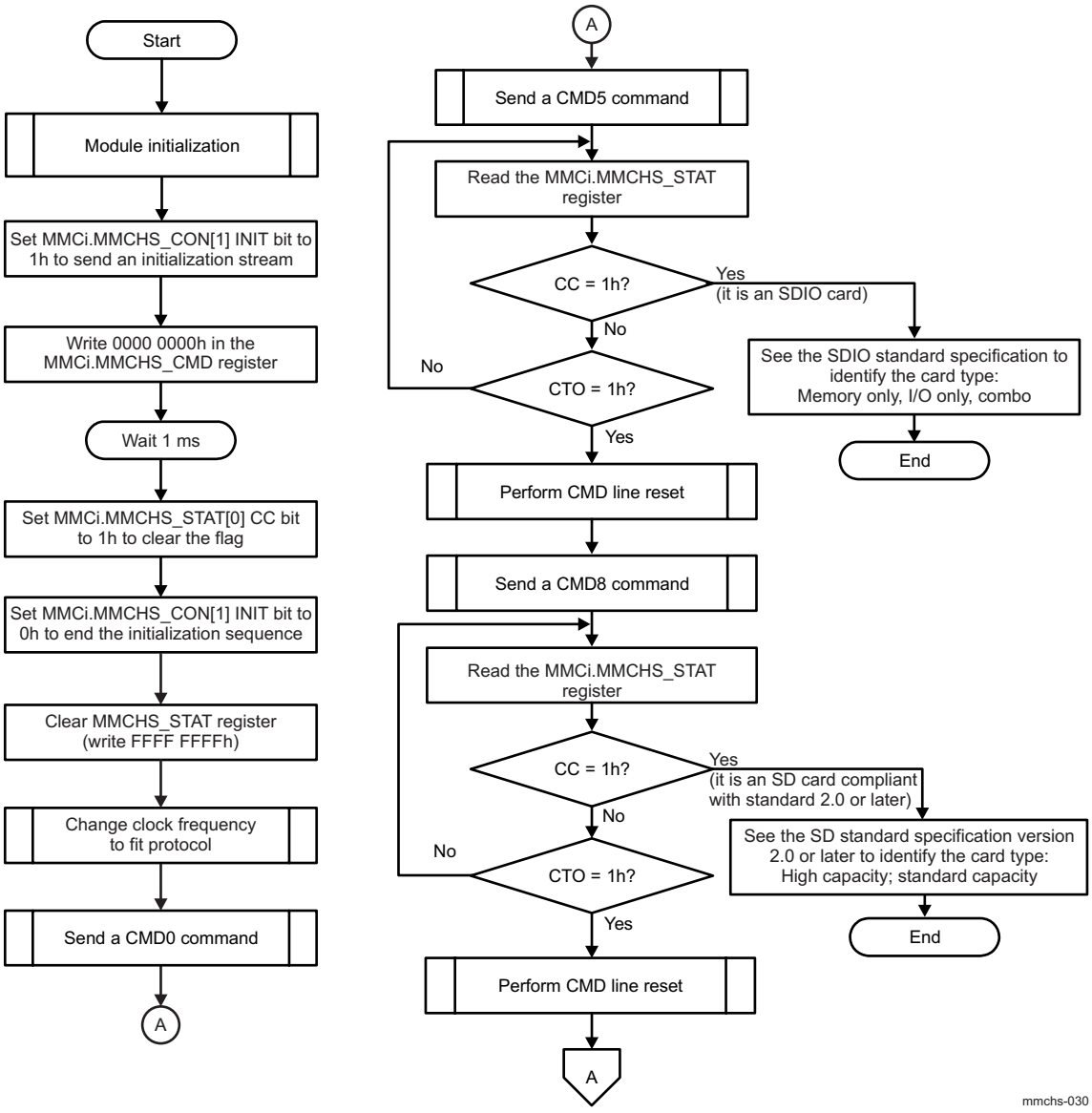


Figure 11-851. MMC Controller Card Identification and Selection – Part 2

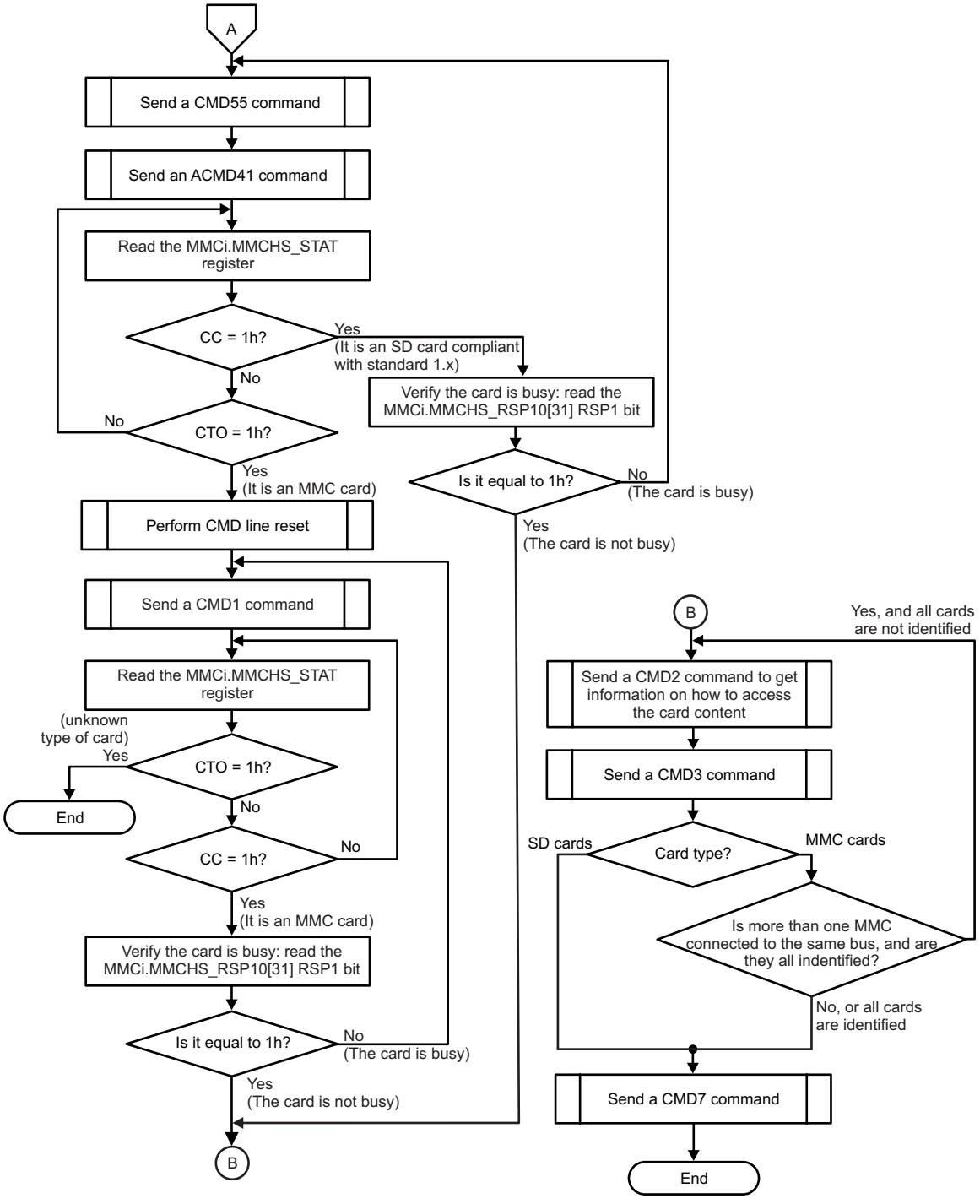


Table 11-1842 lists the subprocess call summary.

Table 11-1842. Subprocess Call Summary for Main Sequence – Card Identification and Selection

Subprocess Name	Cross-Reference
Initialize module.	See Section 11.12.5.1.1.2, *MMC Host Controller Initialization Flow*.
Change clock frequency to fit protocol.	See Section 11.12.5.1.2.1.7.2, *MMC Clock Frequency Change*.

Table 11-1842. Subprocess Call Summary for Main Sequence – Card Identification and Selection (continued)

Subprocess Name	Cross-Reference
Send a command.	See Section 11.12.5.1.2.1.7.1, *Command Transfer Flow*.
Perform CMD line reset.	See Section 11.12.5.1.2.1.1.1, *CMD Line Reset Procedure*.

11.12.5.1.2.1.1.1 CMD Line Reset Procedure

Table 11-1843lists the CML line reset.

Table 11-1843. CMD Line Reset

Step	Access Type	Register/Bit Field/Programming Model	Value
Initiate CMD line reset.	W	MMCi.MMCHS_SYSCTL[25] SRC	1h
Poll the SRC bit until it is set to 1h.	R	MMCi.MMCHS_SYSCTL[25] SRC	= 1h
Wait until the SRC bit returns to 0h (reset procedure is completed).	R	MMCi.MMCHS_SYSCTL[25] SRC	= 0h

11.12.5.1.2.1.2 Read/Write Transfer Flow in DMA Mode With Interrupt

Figure 11-852 shows the read and write protocol in DMA slave mode with interrupt signaling.

Figure 11-852. MMC Controller Read/Write Transfer Flow in DMA Slave Mode With interrupt

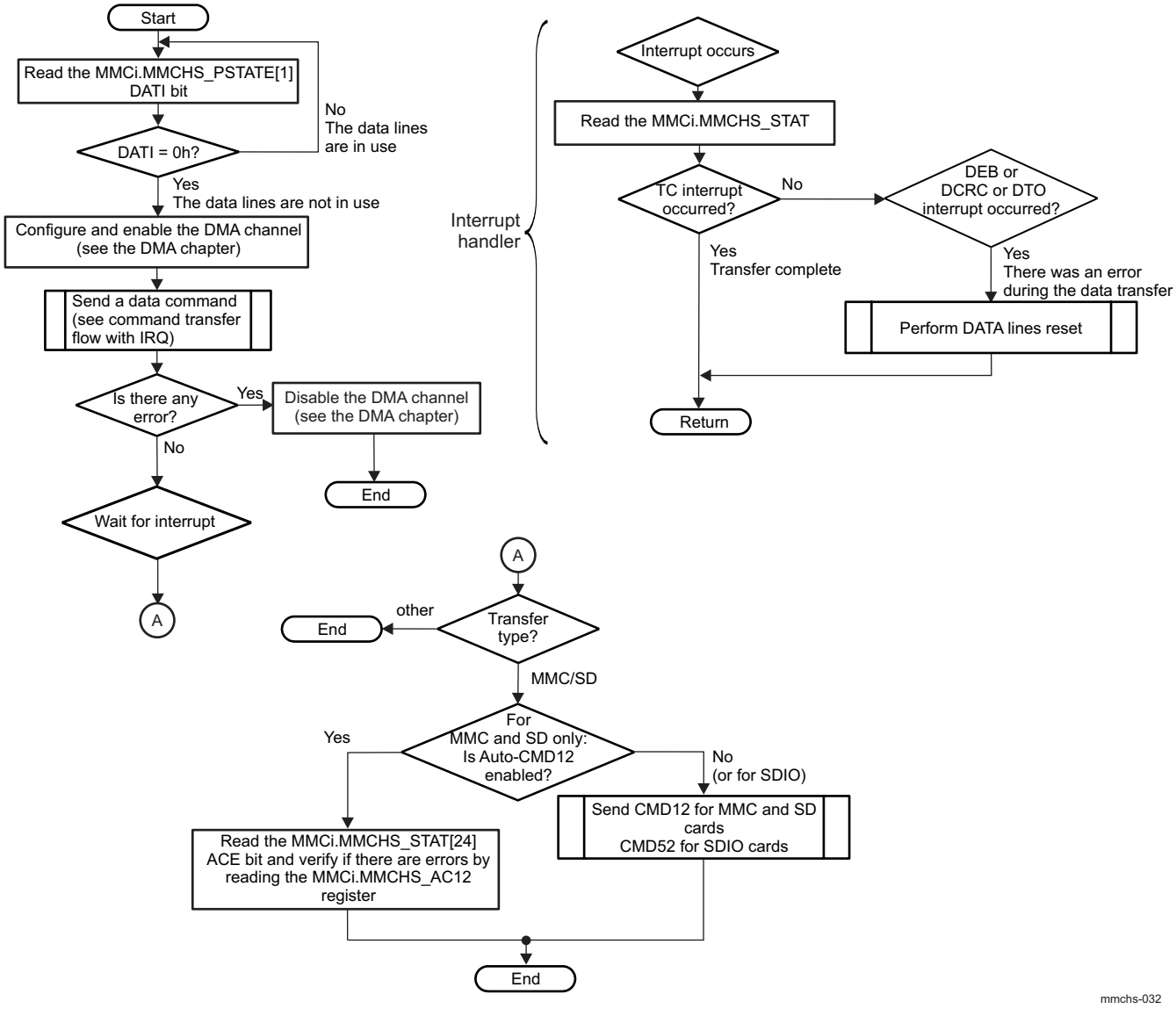


Table 11-1844. Subprocess Call Summary for Main Sequence – MMC Controller Read/Write Transfer Flow in DMA Mode With Interrupt

Subprocess Name	Cross-Reference
Send a data command.	See Figure 11-859.
Perform DATA lines reset.	See Section 11.12.5.1.2.1.2.1.1, *DATA Lines Reset Procedure*.

11.12.5.1.2.1.2.1 DATA Lines Reset Procedure

Table 11-1845 describes the DATA lines reset.

Table 11-1845. DATA Lines Reset

Step	Access Type	Register/Bit Field/Programming Model	Value
Initiate DATA lines reset.	W	MMCi.MMCHS_SYSCTL[26] SRD	1h
Poll the SRD bit until it is set to 1h.	R	MMCi.MMCHS_SYSCTL[26] SRD	= 1h
Wait until the SRD bit returns to 0h (reset procedure is complete).	R	MMCi.MMCHS_SYSCTL[26] SRD	= 0h

11.12.5.1.2.1.3 Read/Write Transfer Flow in DMA Mode With Polling

Figure 11-853 shows the read and write protocol in DMA mode.

Figure 11-853. MMC Controller Read/Write Transfer Flow in DMA Mode With Polling

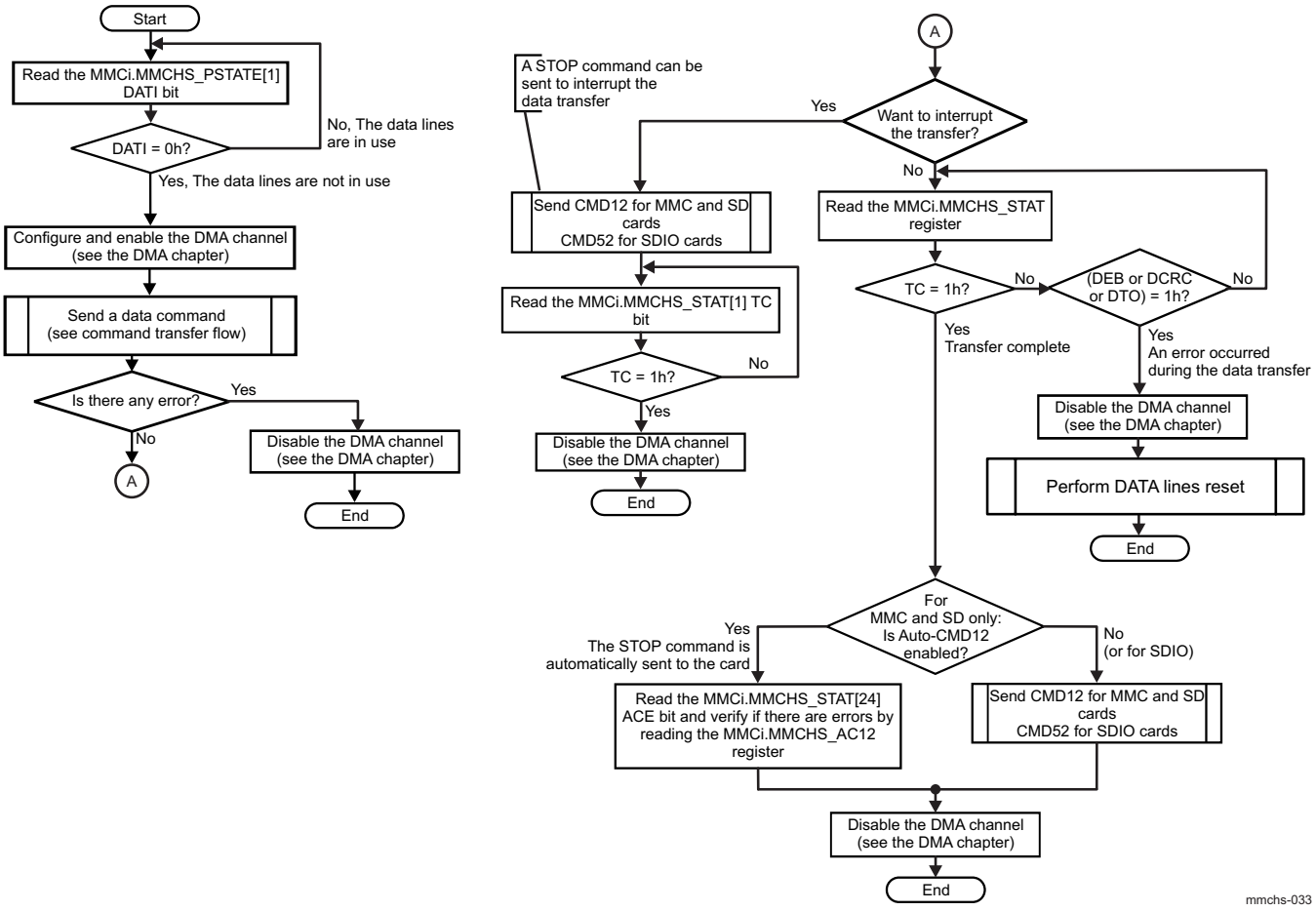


Table 11-1846. Subprocess Call Summary for Main Sequence – Read/Write Transfer Flow in DMA Mode With Polling

Subprocess Name	Cross-Reference
Send command.	See Section 11.12.5.1.2.1.7.1, *Command Transfer Flow*.
Perform DATA lines reset.	See Section 11.12.5.1.2.1.2.1, *DATA Lines Reset Procedure*.

11.12.5.1.2.1.4 Read/Write Transfer Flow Without DMA With Polling

Figure 11-854 shows a read/write transfer without using the DMA and with polling.

Figure 11-854. MMC Controller Read/Write Transfer Flow Without DMA and With Polling

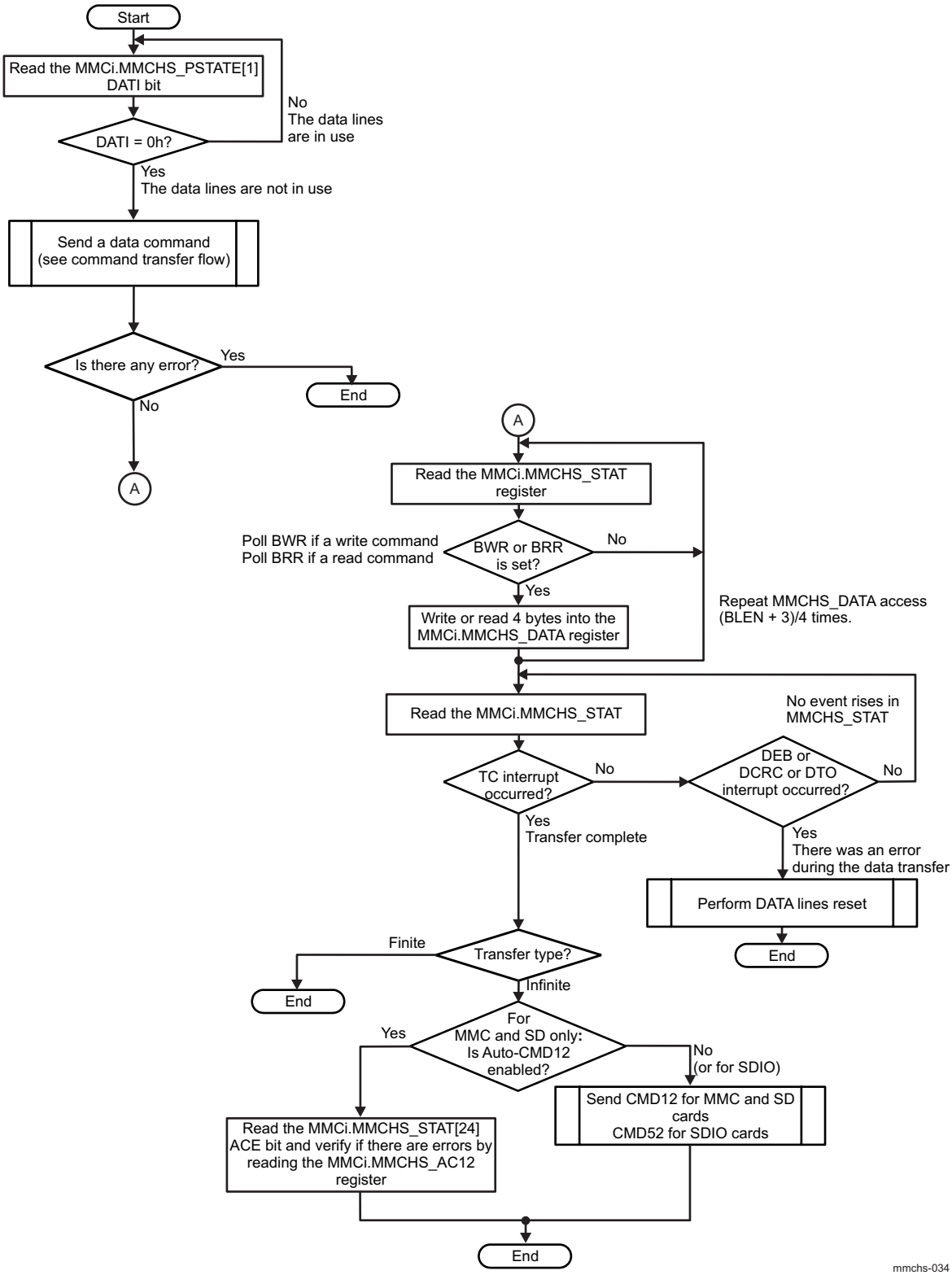


Table 11-1847. Subprocess Call Summary for Main Sequence – Read/Write Transfer Flow Without DMA With Polling

Subprocess Name	Cross-Reference
Send data command.	See Section 11.12.5.1.2.1.7.1, *Command Transfer Flow*.
Perform DATA lines reset.	See Section 11.12.5.1.2.1.2.1, *DATA Lines Reset Procedure*.

11.12.5.1.2.1.5 Read/Write Transfer Flow in CE-ATA Mode

Figure 11-855 shows the read and write CE-ATA protocol when in polling mode.

Figure 11-855. MMC Controller Read/Write in CE-ATA Mode

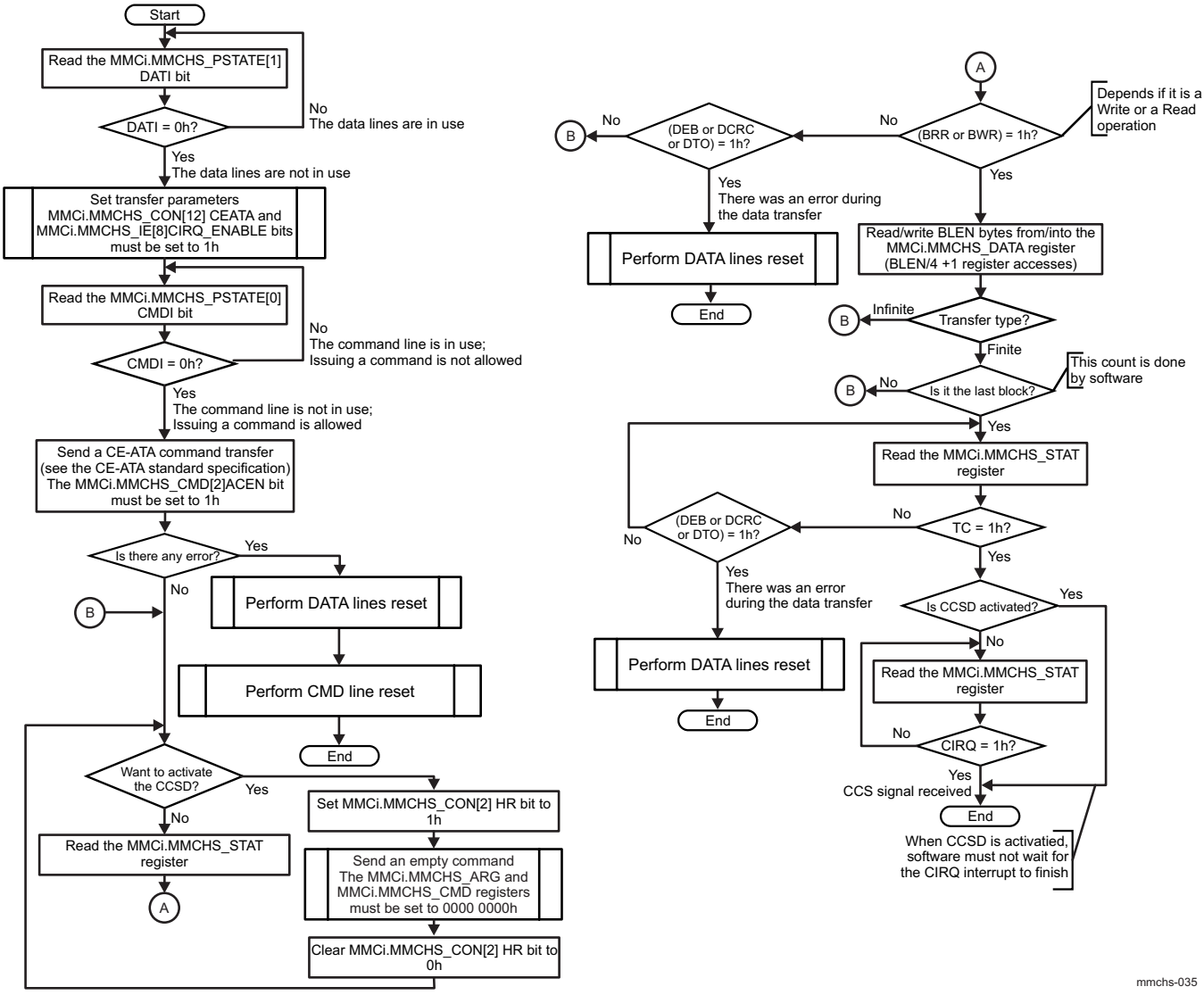


Table 11-1848. Subprocess Call Summary for Main Sequence – Read/Write in CE-ATA Mode

Subprocess Name	Cross-Reference
Perform CMD line reset.	See Section 11.12.5.1.2.1.1.1, *CMD Line Reset Procedure*.
Perform DATA lines reset.	See Section 11.12.5.1.2.1.2.1, *DATA Lines Reset Procedure*.

CAUTION

CE-ATA protocol is supported only by MMC cards.

In CE-ATA mode, issuing a command during the transfer (except a CCSD command) is not allowed.

In CE-ATA mode, infinite transfers are not allowed; only finite transfers are permitted.

11.12.5.1.2.1.6 Suspend-Resume Flow

The suspend-and-resume feature is supported only by SDIO cards.

11.12.5.1.2.1.6.1 Suspend Flow

Figure 11-856 shows the suspend flow for SDIO cards.

Figure 11-856. MMC Controller Suspend Flow

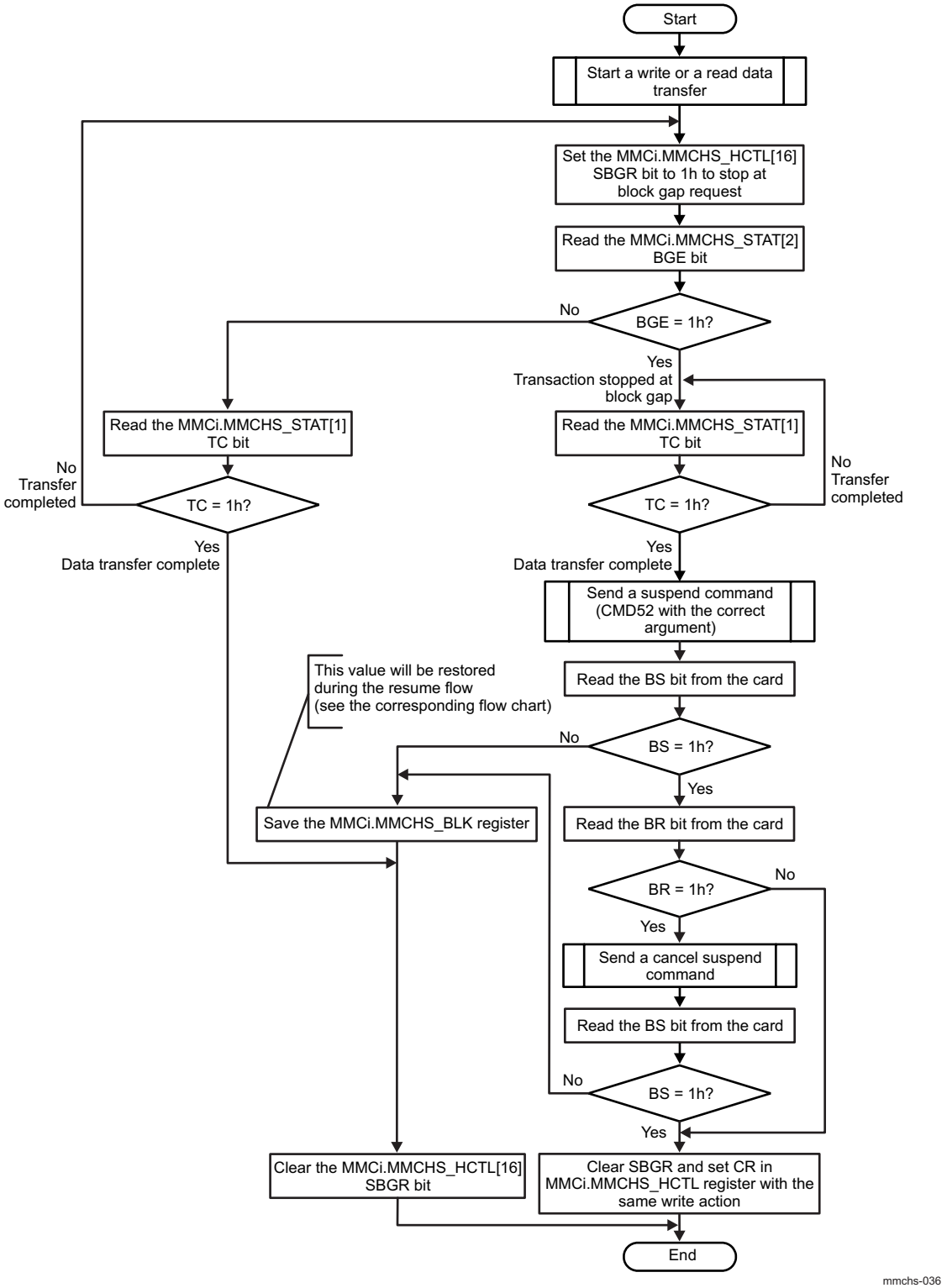


Table 11-1849. Subprocess Call Summary for Main Sequence – Suspend Flow

Subprocess Name	Cross-Reference
Start a write or a read data transfer.	See Section 11.12.5.1.2.1, *Basic Operations for MMC Host Controller*.

Table 11-1849. Subprocess Call Summary for Main Sequence – Suspend Flow (continued)

Subprocess Name	Cross-Reference
Send a suspend command (CMD52 with the correct argument).	See Section 11.12.5.1.2.1.7.1, *Command Transfer Flow*.
Send a cancel suspend command.	See Section 11.12.5.1.2.1.7.1, *Command Transfer Flow*.

11.12.5.1.2.1.6.2 Resume Flow

Figure 11-857 shows the resume flow for SDIO cards.

Figure 11-857. MMC Controller Resume Flow

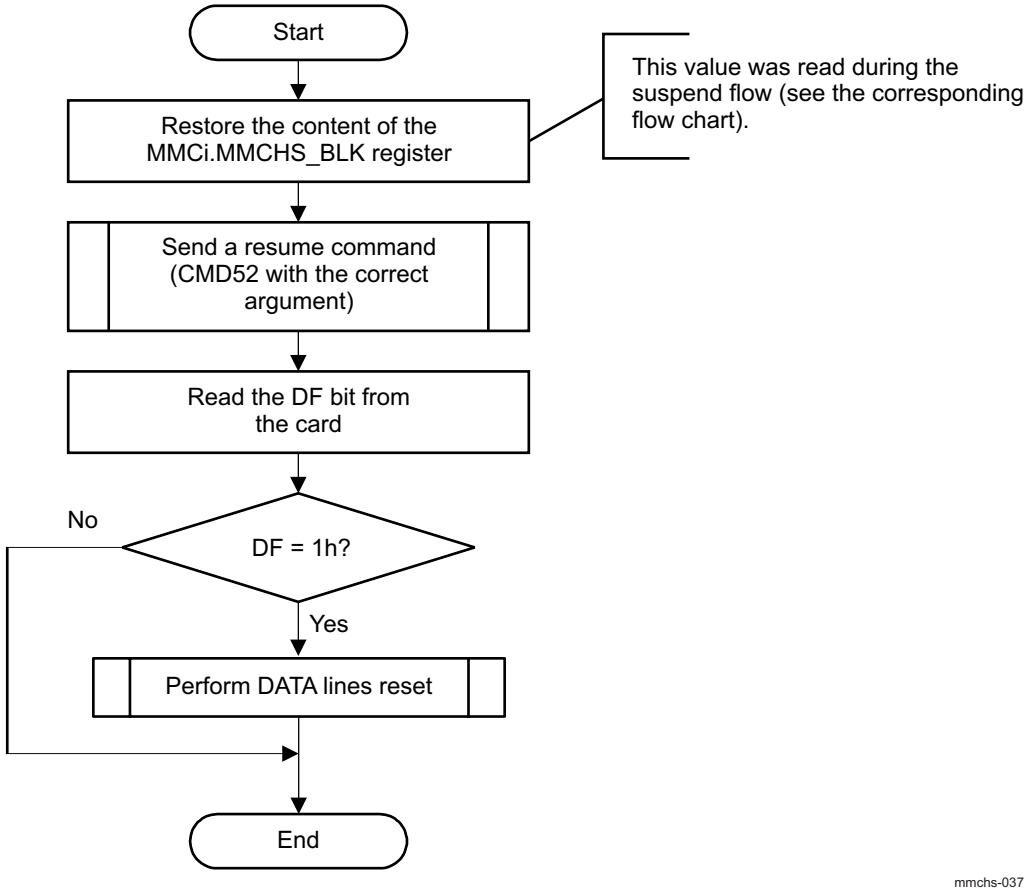


Table 11-1850. Subprocess Call Summary for Main Sequence - Resume Flow

Subprocess Name	Cross-Reference
Send a resume command (CMD52 with the correct argument).	See Section 11.12.5.1.2.1.7.1, *Command Transfer Flow*.
Perform DATA lines reset.	See Section 11.12.5.1.2.1.2.1, *DATA Lines Reset Procedure*.

11.12.5.1.2.1.7 Basic Operations – Steps Detailed

11.12.5.1.2.1.7.1 Command Transfer Flow

Figure 11-858 shows how to send a command to the card using polling instead of interrupts for event signaling.

Figure 11-858. MMC Controller Command Transfer Flow With Polling

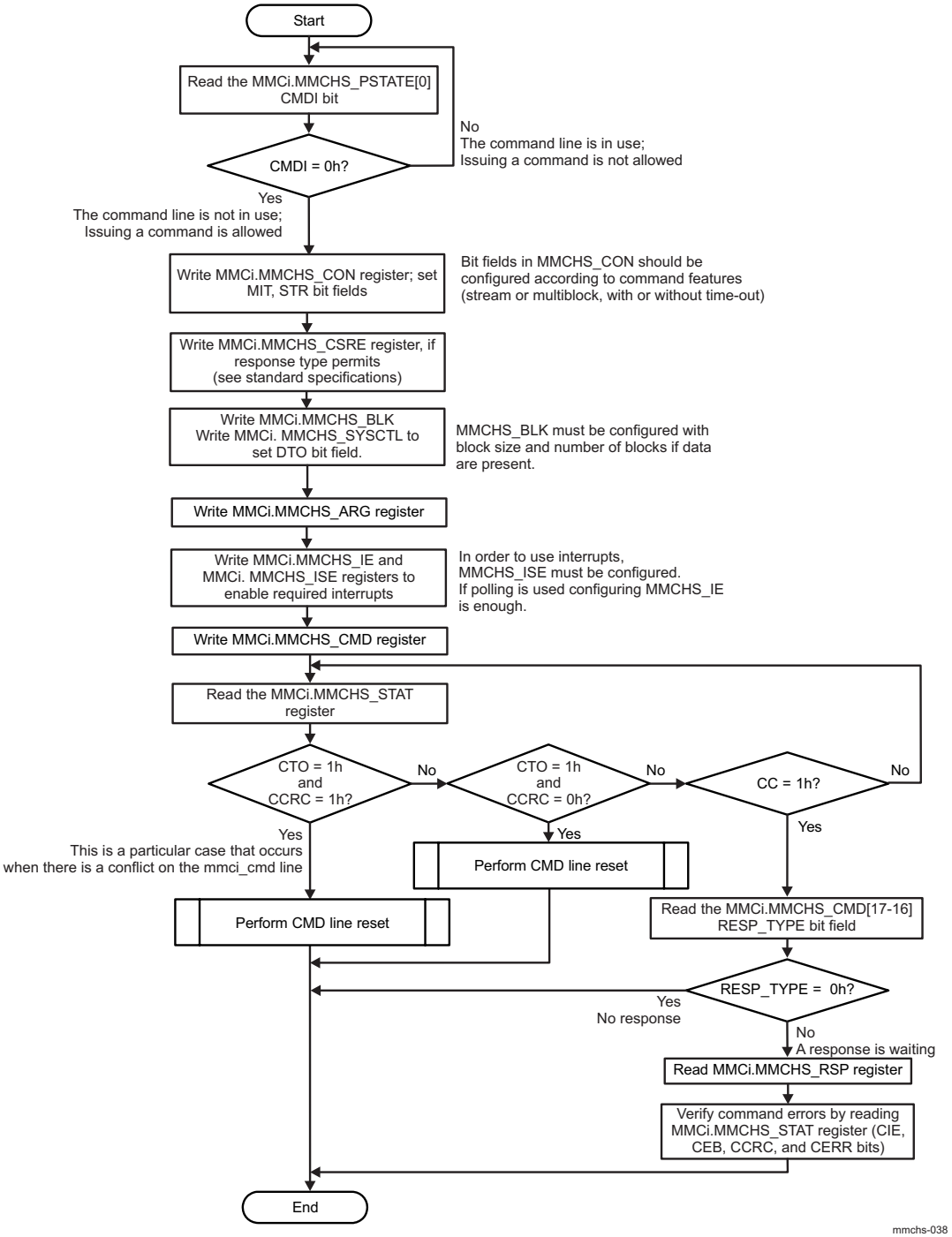


Table 11-1851. Subprocess Call Summary for Main Sequence – Command Transfer Flow With Polling

Subprocess Name	Cross-Reference
Perform CMD line reset.	See Section 11.12.5.1.2.1.1.1, *CMD Line Reset Procedure*.

Figure 11-859 shows how to send a command to the card using interrupts for event signaling.

Figure 11-859. MMC Controller Command Transfer Flow With interrupts

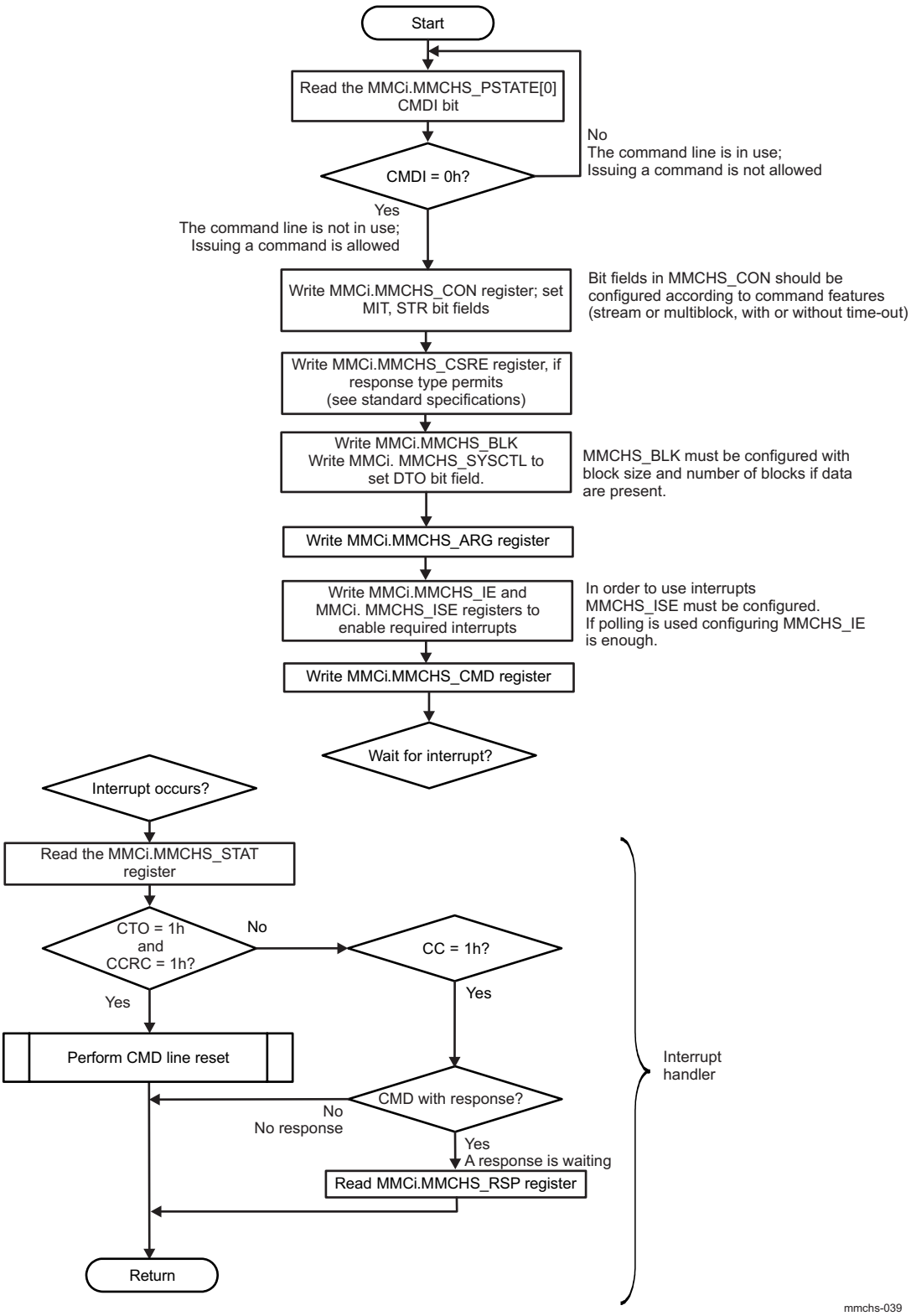


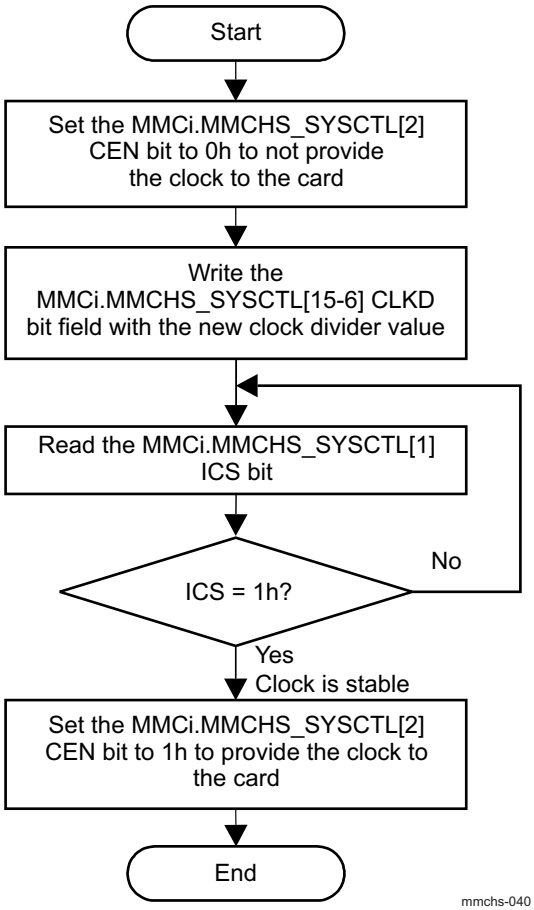
Table 11-1852. Subprocess Call Summary for Main Sequence – Command Transfer Flow With Interrupts

Subprocess Name	Cross-Reference
Perform CMD line reset.	See Section 11.12.5.1.2.1.1.1, *CMD Line Reset Procedure*.

11.12.5.1.2.1.7.2 MMC Clock Frequency Change

Figure 11-860 shows the different steps that allow changing the MMC output clock frequency.

Figure 11-860. MMC Controller Clock Frequency Change Flow



11.12.5.1.2.1.7.3 Bus Width Selection

Figure 11-861 shows the different steps that allow changing the MMC bus width.

Figure 11-861. MMC Controller Bus Width Configuration Flow

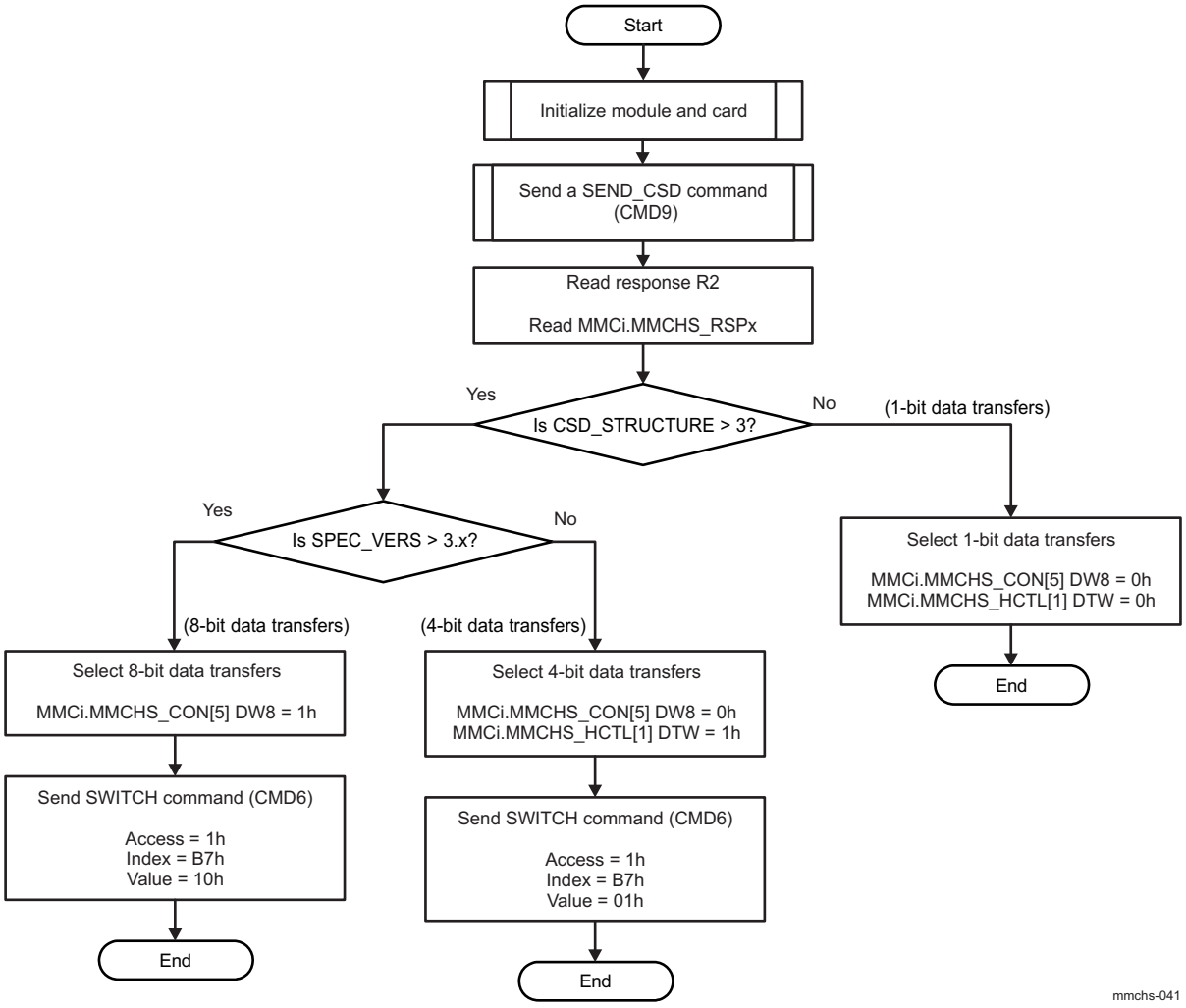


Table 11-1853. Subprocess Call Summary for Main Sequence – Bus Width Configuration Flow

Subprocess Name	Cross-Reference
Initialize module and card.	See Section 11.12.5.1.1.2, *MMC Host Controller Initialization Flow*. See Section 11.12.5.1.2.1.1, *Card Detection, Identification, and Selection*.
Send a SEND_CSD command (CMD9).	See Section 11.12.5.1.2.1.7.1, *Command Transfer Flow*.
Send SWITCH command (CMD6).	See Section 11.12.5.1.2.1.7.1, *Command Transfer Flow*.

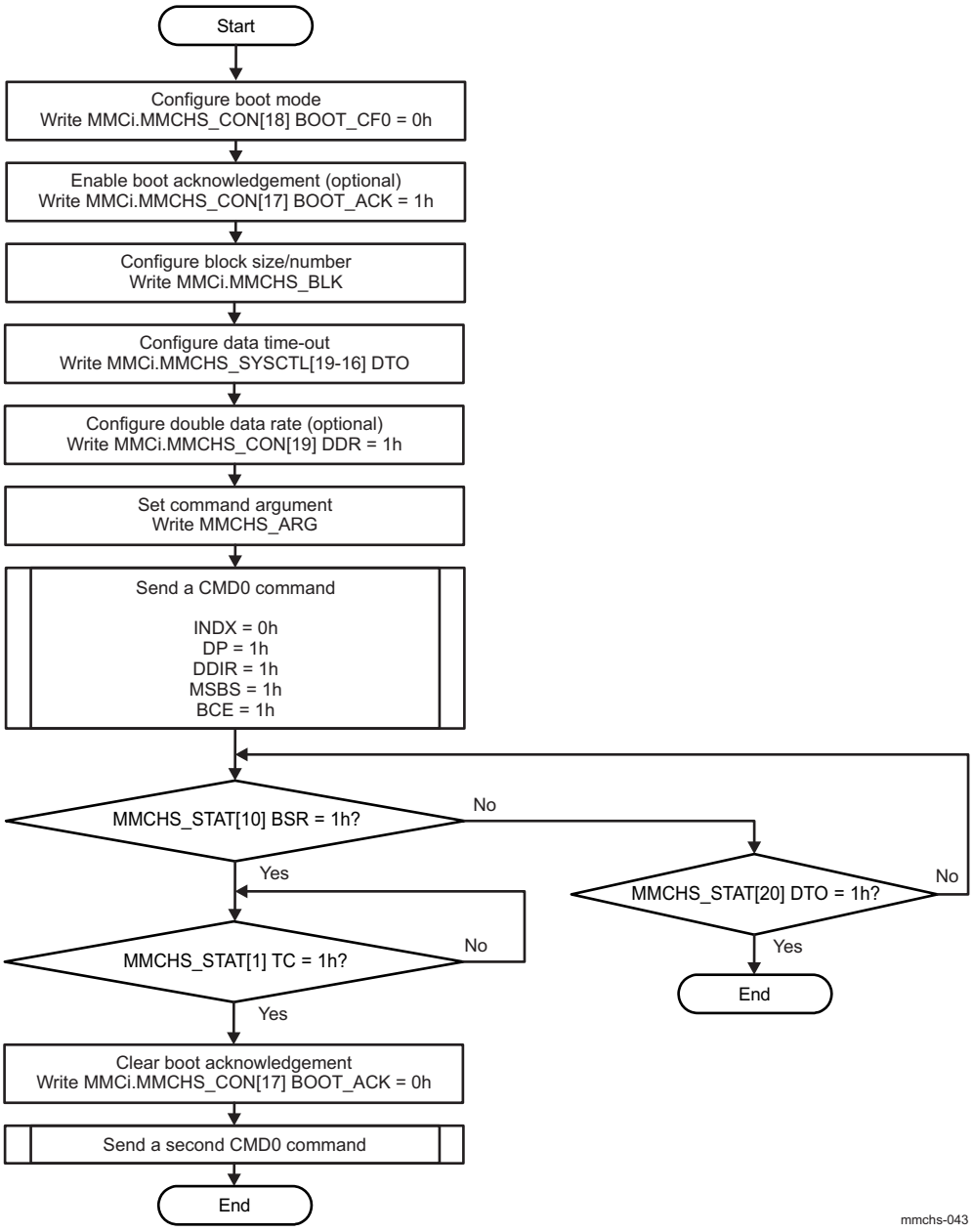
11.12.5.1.2.2 Boot Mode Configuration

The following sections describe the two possible ways to issue a boot command: issue a CMD0 or drive the CMD line to 0 during the whole boot phase.

11.12.5.1.2.2.1 Boot Using CMD0

Figure 11-862 shows the necessary steps to configure the controller boot mode using CMD0.

Figure 11-862. MMC Controller Boot Using CMD0



To abort a boot sequence, the system must issue a CMD0 with the MMCHS_CMD[23-22] CMD_TYPE bit field set to 3h (the MMCHS_CON[17] BOOT_ACK bit previously cleared to 0h) during the transfer to abort the transfer and enable the card to exit from boot state.

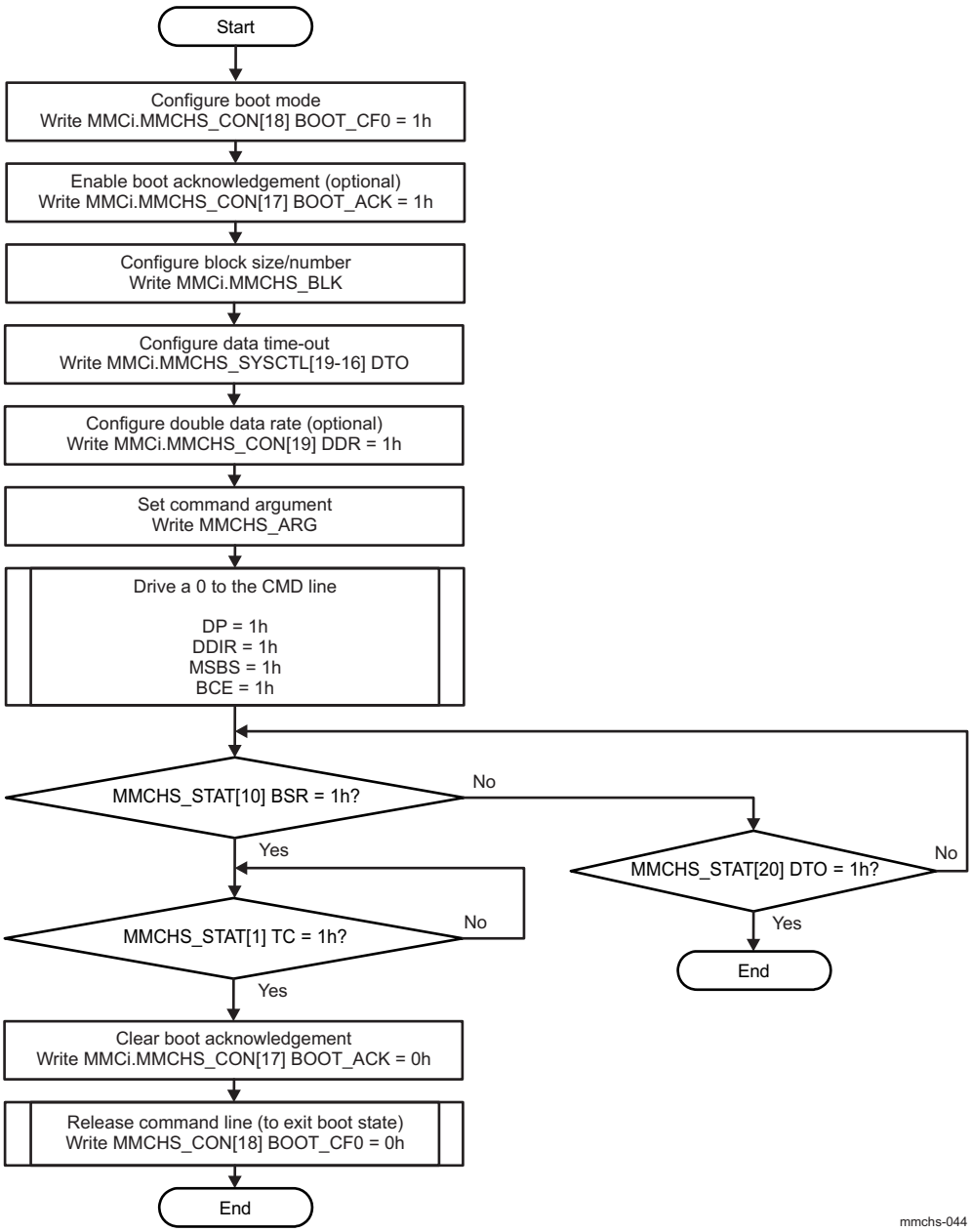
Table 11-1854. Subprocess Call Summary for Main Sequence – Boot Using CMD0

Subprocess Name	Cross-Reference
Send a CMD0 command.	See Section 11.12.5.1.2.1.7.1, *Command Transfer Flow*.

11.12.5.1.2.2.2 Boot With CMD Line Tied to 0

Figure 11-863 shows the necessary steps to configure the controller in this mode; the driver must follow this sequence.

Figure 11-863. MMC Controller Boot With CMD Line Tied to 0



To abort the boot sequence, the system must clear the MMCHS_CON[18] BOOT_CF0 bit to 0h during the transfer to abort the transfer and enable the card to exit from boot state.

Table 11-1855. Subprocess Call Summary for Main Sequence – Boot Using CMD0

Subprocess Name	Cross-Reference
Send a CMD0 command.	See Section 11.12.5.1.2.1.7.1, *Command Transfer Flow*.

11.12.6 MMC/SD Registers

Table 11-1857 lists the memory-mapped registers for the MMC. All register offset addresses not listed in Table 11-1857 should be considered as reserved locations and the register contents should not be modified.

Table 11-1856. MMC Instances

Instance	Base Address
MMCSD_0	2300 0000h
MMCSD_1	2310 0000h

Table 11-1857. MMC Registers

Offset	Acronym	Register Name	MMCSD_0 Physical Address	MMCSD_1 Physical Address	Section
0h	MMCHS_HL_REV	MMC Revision Identifier	2300 0000h	2310 0000h	Section 11.12.6.1
4h	MMCHS_HL_HWINFO	Information about the MMC module's hardware configuration.	2300 0004h	2310 0004h	Section 11.12.6.2
10h	MMCHS_HL_SYSCONFIG	Clock Management Configuration Register	2300 0010h	2310 0010h	Section 11.12.6.3
110h	MMCHS_SYSCONFIG	System Configuration Register	2300 0110h	2310 0110h	Section 11.12.6.4
114h	MMCHS_SYSSTATUS	System Status Register	2300 0114h	2310 0114h	Section 11.12.6.5
124h	MMCHS_CSRE	Card Status Response Error	2300 0124h	2310 0124h	Section 11.12.6.6
128h	MMCHS_SYSTEST	System Test Register	2300 0128h	2310 0128h	Section 11.12.6.7
12Ch	MMCHS_CON	Configuration Register	2300 012Ch	2310 012Ch	Section 11.12.6.8
130h	MMCHS_PWCNT	Power Counter Register	2300 0130h	2310 0130h	Section 11.12.6.9
134h	MMCHS_DLL(1)	DLL control and status register	2300 0134h	2310 0134h	Section 11.12.6.10
200h	MMCHS_SDMASA	SDMA System Address / Argument 2 Register	2300 0200h	2310 0200h	Section 11.12.6.11
204h	MMCHS_BLK	Transfer Length Configuration Register	2300 0204h	2310 0204h	Section 11.12.6.12
208h	MMCHS_ARG	Command Argument Register	2300 0208h	2310 0208h	Section 11.12.6.13
20Ch	MMCHS_CMD	Command and Transfer Mode Register	2300 020Ch	2310 020Ch	Section 11.12.6.14
210h	MMCHS_RSP10	Command Response[31:0] Register	2300 0210h	2310 0210h	Section 11.12.6.15
214h	MMCHS_RSP32	Command Response[63:32] Register	2300 0214h	2310 0214h	Section 11.12.6.16
218h	MMCHS_RSP54	Command Response[95:64] Register	2300 0218h	2310 0218h	Section 11.12.6.17
21Ch	MMCHS_RSP76	Command Response[127:96] Register	2300 021Ch	2310 021Ch	Section 11.12.6.18
220h	MMCHS_DATA	Data Register	2300 0220h	2310 0220h	Section 11.12.6.19
224h	MMCHS_PSTATE	Present State Register	2300 0224h	2310 0224h	Section 11.12.6.20
228h	MMCHS_HCTL	Host Control Register	2300 0228h	2310 0228h	Section 11.12.6.21

(1) DLL feature and SDR104/HS200 modes are not supported.

Table 11-1857. MMC Registers (continued)

Offset	Acronym	Register Name	MMCSD_0 Physical Address	MMCSD_1 Physical Address	Section
22Ch	MMCHS_SYSCTL	SD System Control Register	2300 022Ch	2310 022Ch	Section 11.12.6.22
230h	MMCHS_STAT	Interrupt Status Register	2300 0230h	2310 0230h	Section 11.12.6.23
234h	MMCHS_IE	Interrupt Status Enable Register	2300 0234h	2310 0234h	Section 11.12.6.24
238h	MMCHS_ISE	Interrupt Signal Enable Register	2300 0238h	2310 0238h	Section 11.12.6.25
23Ch	MMCHS_AC12	Host Control 2 Register and Auto CMD Error Status Register	2300 023Ch	2310 023Ch	Section 11.12.6.26
240h	MMCHS_CAPA	Capabilities Register	2300 0240h	2310 0240h	Section 11.12.6.27
244h	MMCHS_CAPA2	Capabilities 2 Register	2300 0244h	2310 0244h	Section 11.12.6.28
248h	MMCHS_CUR_CAPA	Maximum Current Capabilities Register	2300 0248h	2310 0248h	Section 11.12.6.29
250h	MMCHS_FE	Force Event Register for Auto CMD Error Status and Error Interrupt status	2300 0250h	2310 0250h	Section 11.12.6.30
254h	MMCHS_ADMAES	ADMA Error Status Register	2300 0254h	2310 0254h	Section 11.12.6.31
258h	MMCHS_ADMASAL	ADMA System address Low bits	2300 0258h	2310 0258h	Section 11.12.6.32
260h	MMCHS_PVINITSD	Preset Value for Initialization and Default Speed modes	2300 0260h	2310 0260h	Section 11.12.6.33
264h	MMCHS_PVHSSDR12	Preset Value for High Speed and SDR12 speed modes	2300 0264h	2310 0264h	Section 11.12.6.34
268h	MMCHS_PVSDR25SDR50[(2)]	Preset Values for SDR25 and SDR50 speed modes	2300 0268h	2310 0268h	Section 11.12.6.35
26Ch	MMCHS_PVSDR104DDR50[(3)]	Preset Values for SDR104 and DDR50 speed modes	2300 026Ch	2310 026Ch	Section 11.12.6.36
2FCh	MMCHS_REV	Versions Register	2300 02FCh	2310 02FCh	Section 11.12.6.37

(2) SDR50 mode is not supported.
(3) SDR104 mode is not supported.

11.12.6.1 MMCHS_HL_REV Register (Offset = 0h) [reset = 40200303h]

MMCHS_HL_REV is shown in Figure 11-864 and described in Table 11-1859.

MMC Revision Identifier
Used by software to track features, bugs, and compatibility.

Table 11-1858. MMCHS_HL_REV Instances

Instance	Physical Address
MMCSD_0	2300 0000h
MMCSD_1	2310 0000h

Figure 11-864. MMCHS_HL_REV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REVISION																															
R-40200303h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1859. MMCHS_HL_REV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REVISION	R	40200303h	TI internal data. Identifies revision of peripheral.

Table 11-1860. Register Call Summary for MMCHS_HL_REV

MMC/SD Registers
- MMCHS_HL_REV Register (Offset = 0h) [reset = 40200303h]: [0]
- MMC/SD Registers: [0]

11.12.6.2 MMCHS_HL_HWINFO Register (Offset = 4h) [reset = -h]

MMCHS_HL_HWINFO is shown in Figure 11-865 and described in Table 11-1862.

Information about the MMC module's hardware configuration.

Table 11-1861. MMCHS_HL_HWINFO Instances

Instance	Physical Address
MMCSD_0	2300 0004h
MMCSD_1	2310 0004h

Figure 11-865. MMCHS_HL_HWINFO Register

31	30	29	28	27	26	25	24
RESERVED							
R--h							
23	22	21	20	19	18	17	16
RESERVED							
R--h							
15	14	13	12	11	10	9	8
RESERVED							
R--h							
7	6	5	4	3	2	1	0
RESERVED	RETMODE	MEM_SIZE				MERGE_MEM	MADMA_EN
R--h	R--h	R--h				R--h	R--h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1862. MMCHS_HL_HWINFO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	-h	Reserved
6	RETMODE	R	-h	Retention Mode generic parameter This bit field indicates whether the retention mode is supported using the pin PIRFFRET. 0h (R) = Retention mode disabled. 1h (R) = Retention mode enabled. **Note:** Retention mode is not supported. Read returns 0.
5-2	MEM_SIZE	R	-h	Memory size for FIFO buffer 1h (R) = Memory of 512 bytes, max block length is 512 bytes. 2h (R) = Memory of 1024 bytes, max block length is 1024 bytes. 4h (R) = Memory of 2048 bytes, max block length is 2048 bytes. 8h (R) = Memory of 4096 bytes, max block length is 2048 bytes.
1	MERGE_MEM	R	-h	Memory merged for FIFO buffer This bit field defines the configuration of FIFO buffer architecture. If the bit is set STA and DFT shall support clock multiplexing and balancing. 0h (R) = 2 memories instantiated, one per data transfer direction. 1h (R) = A single memory is used with multiplexed addresses, data and clocks.
0	MADMA_EN	R	-h	Master DMA enabled generic parameter This bit defines the configuration of the controller to know if it supports the master DMA management called ADMA. 0h (R) = No Master DMA (ADMA) management supported. 1h (R) = Controller supports ADMA. **Note:** Read returns 1.

Table 11-1863. Register Call Summary for MMCHS_HL_HWINFO

MMC/SD Registers • MMCHS_HL_HWINFO Register (Offset = 4h) [reset = -h]: [0] • MMCHS_CON Register (Offset = 12Ch) [reset = 600h]: [0] • MMCHS_SYSCONFIG Register (Offset = 110h) [reset = 2015h]: [0] • MMC/SD Registers: [0]
MMC/SD Functional Description • Master DMA Operations: [0] • DMA Modes: [0][1] • Memory Size, Block Length, and Buffer-Management Relationship: [0] • Slave DMA Operations: [0]
MMC/SD • MMC/SD Overview: [0]

11.12.6.3 MMCHS_HL_SYSCONFIG Register (Offset = 10h) [reset = 28h]

MMCHS_HL_SYSCONFIG is shown in Figure 11-866 and described in Table 11-1865.

Clock Management Configuration Register.

Table 11-1864. MMCHS_HL_SYSCONFIG Instances

Instance	Physical Address
MMCSD_0	2300 0010h
MMCSD_1	2310 0010h

Figure 11-866. MMCHS_HL_SYSCONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			STANDBYMODE		IDLEMODE	FREEEMU	SOFTRESET
R-0h			R/W-2h		R/W-2h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1865. MMCHS_HL_SYSCONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5-4	STANDBYMODE	R/W	2h	Configuration of the local initiator state management mode By definition, initiator may generate read/write transaction as long as it is out of STANDBY state. 0h (R/W) = Force-standby mode: local initiator is unconditionally placed in standby state. Backup mode, for debug only. 1h (R/W) = No-standby mode: local initiator is unconditionally placed out of standby state. Backup mode, for debug only. 2h (R/W) = Smart-standby mode: local initiator standby status depends on local conditions, that is, the module's functional requirement from the initiator. MMC module shall not generate (initiator-related) wakeup events. 3h (R/W) = Smart-Standby wakeup-capable mode: local initiator standby status depends on local conditions, that is, the module's functional requirement from the initiator. MMC module may generate (master-related) wakeup events when in standby state. Mode is only relevant if the appropriate MMC module "mwakeup" output is implemented. **Note:** Wakeup feature is not supported.

Table 11-1865. MMCHS_HL_SYSCONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	IDLEMODE	R/W	2h	Configuration of the local target state management mode By definition, target can handle read/write transaction as long as it is out of IDLE state. 0h (R/W) = Force-idle mode: local target's idle state follows (acknowledges) the system's IDLE requests unconditionally, that is, regardless of the MMC module's internal requirements. Backup mode, for debug only. 1h (R/W) = No-idle mode: local target never enters idle state. Backup mode, for debug only. 2h (R/W) = Smart-idle mode: local target's idle state eventually follows (acknowledges) the system's IDLE requests, depending on the MMC module's internal requirements. MMC module shall not generate (IRQ- or DMA-request-related) wakeup events. 3h (R/W) = Smart-idle wakeup-capable mode: local target's idle state eventually follows (acknowledges) the system's IDLE requests, depending on the MMC module's internal requirements. MMC module may generate (IRQ- or DMA-request-related) wakeup events when in idle state. Mode is only relevant if the appropriate MMC module "swakeup" output(s) is (are) implemented. **Note:** Wakeup feature is not supported.
1	FREEEMU	R/W	0h	Sensitivity to emulation (debug) suspend input signal Functionality NOT implemented in MMC. 0h (R/W) = MMC module is sensitive to emulation suspend. 1h (R/W) = MMC module is not sensitive to emulation suspend.
0	SOFTRESET	R/W	0h	Software reset (Optional) 0h (W) = No action. 1h (W) = Initiate software reset . 0h (R) = Reset done, no pending action. 1h (R) = Reset (software or other) ongoing.

Table 11-1866. Register Call Summary for MMCHS_HL_SYSCONFIG

MMC/SD Registers
- MMCHS_HL_SYSCONFIG Register (Offset = 10h) [reset = 28h]: [0]
- MMC/SD Registers: [0]

11.12.6.4 MMCHS_SYSCONFIG Register (Offset = 110h) [reset = 2015h]

MMCHS_SYSCONFIG is shown in Figure 11-867 and described in Table 11-1868.

System Configuration Register
This register allows controlling various parameters of the Interconnect interface.

Table 11-1867. MMCHS_SYSCONFIG Instances

Instance	Physical Address
MMCSD_0	2300 0110h
MMCSD_1	2310 0110h

Figure 11-867. MMCHS_SYSCONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		STANDBYMODE		RESERVED		CLOCKACTIVITY	
R-0h		R/W-2h		R-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED			SIDLEMODE		ENAWAKEUP	SOFTRESET	AUTOIDLE
R-0h			R/W-2h		R/W-1h	R/W-0h	R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1868. MMCHS_SYSCONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13-12	STANDBYMODE	R/W	2h	Master interface Power Management, standby/wait control The bit field is only useful when generic parameter MMCHS_HL_HWINFO[0] MADMA_EN (Master ADMA enable) is set as active, otherwise it is a read only register read a '0'. 0h (R/W) = Force-standby. Mstandby is forced unconditionnaly. 1h (R/W) = No-standby. Mstandby is never asserted. 2h (R/W) = Smart-standby mode: local initiator standby status depends on local conditions, that is, the module's functional requirement from the initiator. MMC module shall not generate (initiator-related) wakeup events.
11-10	RESERVED	R	0h	Reserved
9-8	CLOCKACTIVITY	R/W	0h	Clocks activity Bit8: Interface clock. Bit9: Functional clock. 0h (R/W) = Interface and Functional clock may be switched off. 1h (R/W) = Interface clock is maintained. Functional clock may be switched-off. 2h (R/W) = Functional clock is maintained. Interface clock may be switched-off. 3h (R/W) = Interface and Functional clocks are maintained.
7-5	RESERVED	R	0h	Reserved

Table 11-1868. MMCHS_SYSCONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	SIDLEMODE	R/W	2h	Power management 0h (R/W) = If an IDLE request is detected, the MMC acknowledges it unconditionally and goes in Inactive mode. Interrupt and DMA requests are unconditionally de-asserted. 1h (R/W) = If an IDLE request is detected, the request is ignored and the module keeps on behaving normally. 2h (R/W) = Smart-idle mode: local target's idle state eventually follows (acknowledges) the system's IDLE requests, depending on the MMC module's internal requirements. MMC module shall not generate (IRQ- or DMA-request-related) wakeup events. 3h (R/W) = Smart-idle wakeup-capable mode: local target's idle state eventually follows (acknowledges) the system's IDLE requests, depending on the MMC module's internal requirements. MMC module may generate (IRQ- or DMA-request-related) wakeup events when in idle state. Mode is only relevant if the appropriate MMC module "swakeup" output(s) is (are) implemented. **Note:** Wakeup feature is not supported. Smart-idle wakeup-capable mode is not supported.
2	ENAWAKEUP	R/W	1h	Wakeup feature control 0h (R/W) = Wakeup capability is disabled. 1h (R/W) = Wakeup capability is enabled. **Note:** Wakeup feature is not supported.
1	SOFTRESET	R/W	0h	Software reset The bit is automatically reset by the hardware. During reset, it always returns 0. 0h (W) = No effect. 1h (W) = Trigger a module reset. 0h (R) = Normal mode. 1h (R) = The module is reset.
0	AUTOIDLE	R/W	1h	Internal Clock gating strategy 0h (R/W) = Clocks are free-running. 1h (R/W) = Automatic clock gating strategy is applied, based on the Interconnect and MMC interface activity.

Table 11-1869. Register Call Summary for MMCHS_SYSCONFIG

MMC/SD Registers • MMCHS_HCTL Register (Offset = 228h) [reset = 0h]: [0][1][2][3] • MMCHS_SYSCONFIG Register (Offset = 110h) [reset = 2015h]: [0] • MMC/SD Registers: [0] • MMCHS_SYSCTL Register (Offset = 22Ch) [reset = 0h]: [0][1]
MMC/SD Functional Description • Power Management: [0][1][2] • Software Reset: [0][1]
MMC/SD Programming Guide

11.12.6.5 MMCHS_SYSSTATUS Register (Offset = 114h) [reset = 0h]

MMCHS_SYSSTATUS is shown in Figure 11-868 and described in Table 11-1871.

System Status Register
This register provides status information about the module excluding the interrupt status information.

Table 11-1870. MMCHS_SYSSTATUS Instances

Instance	Physical Address
MMCSD_0	2300 0114h
MMCSD_1	2310 0114h

Figure 11-868. MMCHS_SYSSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RESETDONE
R-0h							R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1871. MMCHS_SYSSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	RESETDONE	R	0h	Internal Reset Monitoring **Note:** the debounce clock, the system clock (Interface) and the functional clock shall be provided to the MMC host controller to allow the internal reset monitoring. 0h (R) = Internal module reset is on-going. 1h (R) = Reset completed.

Table 11-1872. Register Call Summary for MMCHS_SYSSTATUS

MMC/SD Registers • MMCHS_SYSSTATUS Register (Offset = 114h) [reset = 0h]: [0] • MMC/SD Registers: [0]
MMC/SD Functional Description • Software Reset: [0] • Hardware Reset: [0][1]

11.12.6.6 MMCHS_CSRE Register (Offset = 124h) [reset = 0h]

MMCHS_CSRE is shown in Figure 11-869 and described in Table 11-1874.

Card Status Response Error
This register enables the host controller to detect card status errors of response type R1, R1b for all cards and of R5, R5b and R6 response for cards types SD or SDIO.
When a bit MMCHS_CSRE[i] is set to 1, if the corresponding bit at the same position in the response MMCHS_RSP10[i] is set to 1, the host controller indicates a card error (MMCHS_STAT[CERR]) interrupt status to avoid the host driver reading the response register (MMCHS_RSP10).
Note: No automatic card error detection for auto CMD12 is implemented; the host system has to check auto CMD12 response register (MMCHS_RSP76) for possible card errors.

Table 11-1873. MMCHS_CSRE Instances

Instance	Physical Address
MMCSD_0	2300 0124h
MMCSD_1	2310 0124h

Figure 11-869. MMCHS_CSRE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSRE																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1874. MMCHS_CSRE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CSRE	R/W	0h	Card status response error

Table 11-1875. Register Call Summary for MMCHS_CSRE

MMC/SD Registers • MMCHS_STAT Register (Offset = 230h) [reset = 0h]: [0] • MMCHS_CSRE Register (Offset = 124h) [reset = 0h]: [0][1] • MMC/SD Registers: [0]
MMC/SD Functional Description • Interrupt Requests: [0]

11.12.6.7 MMCHS_SYSTEST Register (Offset = 128h) [reset = 0h]

MMCHS_SYSTEST is shown in Figure 11-870 and described in Table 11-1877.

System Test Register
This register is used to control the signals that connect to I/O pins when the module is configured in system test (SYSTEST) mode for boundary connectivity verification.
Note: In SYSTEST mode, a write into MMCHS_CMD register will not start a transfer. The buffer behaves as a stack accessible only by the local host (push and pop operations). In this mode, the Transfer Block Size (MMCHS_BLK[BLEN]) and the Blocks count for current transfer (MMCHS_BLK[NBLK]) are needed to generate a Buffer write ready interrupt (MMCHS_STAT[BWR]) or a Buffer read ready interrupt (MMCHS_STAT[BRR]) and DMA requests if enabled.

Table 11-1876. MMCHS_SYSTEST Instances

Instance	Physical Address
MMCSD_0	2300 0128h
MMCSD_1	2310 0128h

Figure 11-870. MMCHS_SYSTEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							OBI
R-0h							R-0h
15	14	13	12	11	10	9	8
SDCD	SDWP	WAKD	SSB	D7D	D6D	D5D	D4D
R-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
D3D	D2D	D1D	D0D	DDIR	CDAT	CDIR	MCKD
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1877. MMCHS_SYSTEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	OBI	R	0h	Out-Of-Band Interrupt (OBI) data value 0h (R) = The Out-of-Band Interrupt pin is driven low. 1h (R) = The Out-of-Band Interrupt pin is driven high. **Note:** Out-Of-Band Interrupt feature is not supported. Read returns 0.
15	SDCD	R	0h	Card detect input signal (MMCi_SDCD) data value 0h (R) = The card detect pin is driven low. 1h (R) = The card detect pin is driven high.
14	SDWP	R	0h	Write protect input signal (MMCi_SDWP) data value 0h (R) = The write protect pin MMCi_SDWP is driven low. 1h (R) = The write protect pin MMCi_SDWP is driven high.

Table 11-1877. MMCHS_SYSTEST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	WAKD	R/W	0h	Wake request output signal data value 0h (W) = The pin SWAKEUP is driven low. 1h (W) = The pin SWAKEUP is driven high. 0h (R) = No action. Returns 0. 1h (R) = No action. Returns 1. **Note:** Wakeup feature is not supported.
12	SSB	R/W	0h	Set status bit This bit must be cleared prior attempting to clear a status bit of the interrupt status register (MMCHS_STAT). 0h (W) = Clear this SSB bitfield. Writing 0 does not clear already set status bits. 1h (W) = Force to 1 all status bits of the interrupt status register (MMCHS_STAT) only if the corresponding bitfield in the Interrupt signal enable register (MMCHS_ISE) is set. 0h (R) = No action. Returns 0. 1h (R) = No action. Returns 1.
11	D7D	R/W	0h	DAT7 input/output signal data value 0h (W) = If SYSTEST[DDIR] = 0 (output mode direction), the DAT7 line is driven low. If SYSTEST[DDIR] = 1 (input mode direction), no effect. 1h (W) = If SYSTEST[DDIR] = 0 (output mode direction), the DAT7 line is driven high. If SYSTEST[DDIR] = 1 (input mode direction), no effect. 0h (R) = If SYSTEST[DDIR] = 1 (input mode direction), returns the value on the DAT7 line (low). If SYSTEST[DDIR] = 0 (output mode direction), returns 0. 1h (R) = If SYSTEST[DDIR] = 1 (input mode direction), returns the value on the DAT7 line (high) If SYSTEST[DDIR] = 0 (output mode direction), returns 1.
10	D6D	R/W	0h	DAT6 input/output signal data value 0h (W) = If SYSTEST[DDIR] = 0 (output mode direction), the DAT6 line is driven low. If SYSTEST[DDIR] = 1 (input mode direction), no effect. 1h (W) = If SYSTEST[DDIR] = 0 (output mode direction), the DAT6 line is driven high. If SYSTEST[DDIR] = 1 (input mode direction), no effect. 0h (R) = If SYSTEST[DDIR] = 1 (input mode direction), returns the value on the DAT6 line (low). If SYSTEST[DDIR] = 0 (output mode direction), returns 0. 1h (R) = If SYSTEST[DDIR] = 1 (input mode direction), returns the value on the DAT6 line (high) If SYSTEST[DDIR] = 0 (output mode direction), returns 1.
9	D5D	R/W	0h	DAT5 input/output signal data value 0h (W) = If SYSTEST[DDIR] = 0 (output mode direction), the DAT5 line is driven low. If SYSTEST[DDIR] = 1 (input mode direction), no effect. 1h (W) = If SYSTEST[DDIR] = 0 (output mode direction), the DAT5 line is driven high. If SYSTEST[DDIR] = 1 (input mode direction), no effect. 0h (R) = If SYSTEST[DDIR] = 1 (input mode direction), returns the value on the DAT5 line (low). If SYSTEST[DDIR] = 0 (output mode direction), returns 0. 1h (R) = If SYSTEST[DDIR] = 1 (input mode direction), returns the value on the DAT5 line (high) If SYSTEST[DDIR] = 0 (output mode direction), returns 1.

Table 11-1877. MMCHS_SYSTEST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	D4D	R/W	0h	DAT4 input/output signal data value 0h (W) = If SYSTEST[DDIR] = 0 (output mode direction), the DAT4 line is driven low. If SYSTEST[DDIR] = 1 (input mode direction), no effect. 1h (W) = If SYSTEST[DDIR] = 0 (output mode direction), the DAT4 line is driven high. If SYSTEST[DDIR] = 1 (input mode direction), no effect. 0h (R) = If SYSTEST[DDIR] = 1 (input mode direction), returns the value on the DAT4 line (low). If SYSTEST[DDIR] = 0 (output mode direction), returns 0. 1h (R) = If SYSTEST[DDIR] = 1 (input mode direction), returns the value on the DAT4 line (high) If SYSTEST[DDIR] = 0 (output mode direction), returns 1.
7	D3D	R/W	0h	DAT3 input/output signal data value 0h (W) = If SYSTEST[DDIR] = 0 (output mode direction), the DAT3 line is driven low. If SYSTEST[DDIR] = 1 (input mode direction), no effect. 1h (W) = If SYSTEST[DDIR] = 0 (output mode direction), the DAT3 line is driven high. If SYSTEST[DDIR] = 1 (input mode direction), no effect. 0h (R) = If SYSTEST[DDIR] = 1 (input mode direction), returns the value on the DAT3 line (low). If SYSTEST[DDIR] = 0 (output mode direction), returns 0. 1h (R) = If SYSTEST[DDIR] = 1 (input mode direction), returns the value on the DAT3 line (high) If SYSTEST[DDIR] = 0 (output mode direction), returns 1.
6	D2D	R/W	0h	DAT2 input/output signal data value 0h (W) = If SYSTEST[DDIR] = 0 (output mode direction), the DAT2 line is driven low. If SYSTEST[DDIR] = 1 (input mode direction), no effect. 1h (W) = If SYSTEST[DDIR] = 0 (output mode direction), the DAT2 line is driven high. If SYSTEST[DDIR] = 1 (input mode direction), no effect. 0h (R) = If SYSTEST[DDIR] = 1 (input mode direction), returns the value on the DAT2 line (low). If SYSTEST[DDIR] = 0 (output mode direction), returns 0. 1h (R) = If SYSTEST[DDIR] = 1 (input mode direction), returns the value on the DAT2 line (high) If SYSTEST[DDIR] = 0 (output mode direction), returns 1.
5	D1D	R/W	0h	DAT1 input/output signal data value 0h (W) = If SYSTEST[DDIR] = 0 (output mode direction), the DAT1 line is driven low. If SYSTEST[DDIR] = 1 (input mode direction), no effect. 1h (W) = If SYSTEST[DDIR] = 0 (output mode direction), the DAT1 line is driven high. If SYSTEST[DDIR] = 1 (input mode direction), no effect. 0h (R) = If SYSTEST[DDIR] = 1 (input mode direction), returns the value on the DAT1 line (low). If SYSTEST[DDIR] = 0 (output mode direction), returns 0. 1h (R) = If SYSTEST[DDIR] = 1 (input mode direction), returns the value on the DAT1 line (high) If SYSTEST[DDIR] = 0 (output mode direction), returns 1.

Table 11-1877. MMCHS_SYSTEST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	D0D	R/W	0h	DAT0 input/output signal data value 0h (W) = If SYSTEST[DDIR] = 0 (output mode direction), the DAT0 line is driven low. If SYSTEST[DDIR] = 1 (input mode direction), no effect. 1h (W) = If SYSTEST[DDIR] = 0 (output mode direction), the DAT0 line is driven high. If SYSTEST[DDIR] = 1 (input mode direction), no effect. 0h (R) = If SYSTEST[DDIR] = 1 (input mode direction), returns the value on the DAT0 line (low). If SYSTEST[DDIR] = 0 (output mode direction), returns 0. 1h (R) = If SYSTEST[DDIR] = 1 (input mode direction), returns the value on the DAT0 line (high) If SYSTEST[DDIR] = 0 (output mode direction), returns 1.
3	DDIR	R/W	0h	Control of the DAT[7:0] pins direction 0h (W) = The DAT lines are outputs (host to card) 1h (W) = The DAT lines are inputs (card to host) 0h (R) = No action. Returns 0. 1h (R) = No action. Returns 1.
2	CDAT	R/W	0h	CMD input/output signal data value 0h (W) = If SYSTEST[CDIR] = 0 (output mode direction), the CMD line is driven low. If SYSTEST[CDIR] = 1 (input mode direction), no effect. 1h (W) = If SYSTEST[CDIR] = 0 (output mode direction), the CMD line is driven high. If SYSTEST[CDIR] = 1 (input mode direction), no effect. 0h (R) = If SYSTEST[CDIR] = 1 (input mode direction), returns the value on the CMD line (low). If SYSTEST[CDIR] = 0 (output mode direction), returns 0. 1h (R) = If SYSTEST[CDIR] = 1 (input mode direction), returns the value on the CMD line (high) If SYSTEST[CDIR] = 0 (output mode direction), returns 1.
1	CDIR	R/W	0h	Control of the CMD pin direction 0h (W) = The CMD line is an output (host to card) 1h (W) = The CMD line is an input (card to host) 0h (R) = No action. Returns 0. 1h (R) = No action. Returns 1.
0	MCKD	R/W	0h	MMC clock output signal data value 0h (W) = The output clock is driven low. 1h (W) = The output clock is driven high. 0h (R) = No action. Returns 0. 1h (R) = No action. Returns 1.

Table 11-1878. Register Call Summary for MMCHS_SYSTEST

MMC/SD Registers • MMCHS_SYSTEST Register (Offset = 128h) [reset = 0h]: [0] • MMC/SD Registers: [0]
MMC/SD Functional Description • Test Registers: [0]

11.12.6.8 MMCHS_CON Register (Offset = 12Ch) [reset = 600h]

MMCHS_CON is shown in Figure 11-871 and described in Table 11-1880.

Configuration Register
This register is used:
- to select the functional mode or the SYSTEST mode for any card.
- to send an initialization sequence to any card.
- to enable the detection on DAT[1] of a card interrupt for SDIO cards only.
and also to configure:
- specific data and command transfers for MMC cards only.
- the parameters related to the card detect and write protect input signals.

Table 11-1879. MMCHS_CON Instances

Instance	Physical Address
MMCSD_0	2300 012Ch
MMCSD_1	2310 012Ch

Figure 11-871. MMCHS_CON Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED		SDMA_LNE	DMA_MNS	DDR	BOOT_CF0	BOOT_ACK	CLKEXTFREE
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
PADEN	OBIE	OBIP	CEATA	CTPL	DVAL		WPP
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-3h		R/W-0h
7	6	5	4	3	2	1	0
CDP	MIT	DW8	MODE	STR	HR	INIT	OD
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1880. MMCHS_CON Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved
21	SDMA_LNE	R/W	0h	Slave DMA Level/Edge Request The waveform of the DMA request can be configured either edge sensitive with early de-assertion on first access to MMCHS_DATA register or late de-assertion, request remains active until last allowed data written into MMCHS_DATA. 0h (R/W) = Slave DMA edge sensitive, Early DMA de-assertion. 1h (R/W) = Slave DMA level sensitive, Late DMA de-assertion.
20	DMA_MNS	R/W	0h	DMA Master or Slave selection When this bit is set and the controller is configured to use the DMA, Interconnect master interface is used to get datas from system using ADMA2 procedure (direct access to the memory). This option is only available if generic parameter MMCHS_HL_HWINFO[0] MADMA_EN is asserted to '1'. 0h (R/W) = The controller is slave on data transfers with system. 1h (R/W) = The controller is master on data exchange with system, controller must be configured as using DMA.

Table 11-1880. MMCHS_CON Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	DDR	R/W	0h	Dual Data Rate mode When this bit field is set, the controller uses both clock edge to emit or receive data. Odd bytes are transmitted on falling edges and even bytes are transmitted on rise edges. It only applies on Data bytes and CRC, Start, end bits and CRC status are kept full cycle. This bit field is only meaningful and active for even clock divider ratio of MMCHS_SYSCTL[CLKD], it is insensitive to MMCHS_HCTL[HSPE] setting. 0h (R/W) = Standard mode: data are transmitted on a single edge depending on MMCHS_HCTL[HSPE]. 1h (R/W) = Data Bytes and CRC are transmitted on both edge.
18	BOOT_CF0	R/W	0h	Boot status supported This bit field is set when the CMD line need to be forced to '0' for a boot sequence. CMD line is driven to '0' after writing in MMCHS_CMD. The line is released when this bit field is de-asserted and abort data transfer in case of a pending transaction. 0h (W) = CMD line is released when it was previously forced to '0' by a boot sequence. 1h (W) = CMD line forced to '0' is enabled and will be active after writing into MMCHS_CMD. 0h (R) = CMD line not forced. 1h (R) = CMD line forced to '0' is enabled.
17	BOOT_ACK	R/W	0h	Boot acknowledge received When this bit is set the controller should receive a boot status on DAT0 line after next command issued. If no status is received a data timeout will be generated. 0h (R/W) = No acknowledge to be received. 1h (R/W) = A boot status will be received on DAT0 line after issuing a command.
16	CLKEXTFREE	R/W	0h	External clock free running This bit field is used to maintain card clock out of transfer transaction to enable slave module for example to generate a synchronous interrupt on DAT[1]. The Clock will be maintain only if MMCHS_SYSCTL[CEN] is set. 0h (R/W) = External card clock is cut off outside active transaction period. 1h (R/W) = External card clock is maintain even out of active transaction period only if MMCHS_SYSCTL[CEN] is set.
15	PADEN	R/W	0h	Control Power for MMC Lines This bit field is only useful when MMC PADs contain power saving mechanism to minimize its leakage power. It works as a GPIO that directly control the ACTIVE pin of PADs. Excepted for DAT[1], the signal is also combine outside the module with the dedicated power control MMCHS_CON[CTPL] bit. 0h (R/W) = ADPIDLE module pin is not forced, it is automatically generated by the MMC fsms. 1h (R/W) = ADPIDLE module pin is forced to active state. **Note:** This feature is not supported.

Table 11-1880. MMCHS_CON Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	OBIE	R/W	0h	Out-of-Band Interrupt Enable MMC cards only: This bit enables the detection of Out-of-Band Interrupt on MMC OBI input pin. The usage of the Out-of-Band signal (OBI) is optional and depends on the system integration. 0h (R/W) = Out-of-Band interrupt detection disabled. 1h (R/W) = Out-of-Band interrupt detection enabled. **Note:** Out-Of-Band Interrupt feature is not supported.
13	OBIP	R/W	0h	Out-of-Band Interrupt Polarity MMC cards only: This bit selects the active level of the out-of-band interrupt coming from MMC cards. The usage of the Out-of-Band signal (OBI) is optional and depends on the system integration. 0h (R/W) = Active high level. 1h (R/W) = Active low level. **Note:** Out-Of-Band Interrupt feature is not supported.
12	CEATA	R/W	0h	CE-ATA control mode MMC cards compliant with CE-ATA: By default, this bit is set to 0. It is used to indicate that next commands are considered as specific CE-ATA commands that potentially use 'command completion' features. 0h (R/W) = Standard MMC/SD/SDIO mode. 1h (R/W) = CE-ATA mode next commands are considered as CE-ATA commands.
11	CTPL	R/W	0h	Control Power for DAT[1] line MMC and SD cards: By default, this bit is set to 0 and the host controller automatically disables all the input buffers outside of a transaction to minimize the leakage current. SDIO cards: When this bit is set to 1, the host controller automatically disables all the input buffers except the buffer of DAT[1] outside of a transaction in order to detect asynchronous card interrupt on DAT[1] line and minimize the leakage current of the buffers. 0h (R/W) = Disable all the input buffers outside of a transaction. 1h (R/W) = Disable all the input buffers except the buffer of DAT[1] outside of a transaction.
10-9	DVAL	R/W	3h	Debounce filter value All cards: This bit field is used to define a debounce period to filter the card detect input signal (MMCi_SDCD). The usage of the card detect input signal (MMCi_SDCD) is optional and depends on the system integration and the type of the connector housing that accommodates the card. 0h (R/W) = 33 us debounce period. 1h (R/W) = 231 us debounce period. 2h (R/W) = 1 ms debounce period. 3h (R/W) = 8,4 ms debounce period.

Table 11-1880. MMCHS_CON Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	WPP	R/W	0h	Write protect polarity For SD and SDIO cards only: This bit selects the active level of the write protect input signal (MMCi_SDWP). The usage of the write protect input signal (MMCi_SDWP) is optional and depends on the system integration and the type of the connector housing that accommodates the card. 0h (R/W) = Active high level. 1h (R/W) = Active low level.
7	CDP	R/W	0h	Card detect polarity All cards: This bit selects the active level of the card detect input signal (MMCi_SDCD). The usage of the card detect input signal (MMCi_SDCD) is optional and depends on the system integration and the type of the connector housing that accommodates the card. 0h (R/W) = Active low level. 1h (R/W) = Active high level.
6	MIT	R/W	0h	MMC interrupt command Only for MMC cards: This bit must be set to 1, when the next write access to the command register (MMCHS_CMD) is for writing a MMC interrupt command (CMD40) requiring the command timeout detection to be disabled for the command response. 0h (R/W) = Command timeout enabled. 1h (R/W) = Command timeout disabled.
5	DW8	R/W	0h	8-bit mode MMC select For SD/SDIO cards, this bit must be set to 0. For MMC card, this bit must be set following a valid SWITCH command (CMD6) with the correct value and extend CSD index written in the argument. Prior to this command, the MMC card configuration register (CSD and EXT_CSD) must be verified for compliancy with MMC standard specification 4.x (see section 3.6). 0h (R/W) = 1-bit or 4-bit Data width (DAT[0] used, MMC, SD cards). 1h (R/W) = 8-bit Data width (DAT[7:0] used, MMC cards).
4	MODE	R/W	0h	Mode select All cards: This bit select between Functional mode and SYSTEST mode. 0h (R/W) = Functional mode. Transfers to the MMC/SD/SDIO cards follow the card protocol. MMC clock is enabled. MMC/SD transfers are operated under the control of the CMD register. 1h (R/W) = SYSTEST mode The signal pins are configured as general-purpose input/output and the 1024-byte buffer is configured as a stack memory accessible only by the local host or system DMA. The pins retain their default type (input, output or in-out). SYSTEST mode is operated under the control of the SYSTEST register.

Table 11-1880. MMCHS_CON Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	STR	R/W	0h	Stream command Only for MMC cards: This bit must be set to 1 only for the stream data transfers (read or write) of the adtc commands. Stream read is a class 1 command (CMD11: READ_DAT_UNTIL_STOP). Stream write is a class 3 command (CMD20: WRITE_DAT_UNTIL_STOP). 0h (R/W) = Block oriented data transfer. 1h (R/W) = Stream oriented data transfer.
2	HR	R/W	0h	Broadcast host response Only for MMC cards: This bit field is used to force the host to generate a 48-bit response for bc command type. It can be used to terminate the interrupt mode by generating a CMD40 response by the core (see section 4.3, "Interrupt Mode", in the MMC specification). In order to have the host response to be generated in open drain mode, the register MMCHS_CON[OD] must be set to 1. When MMCHS_CON[CEATA] is set to 1 and MMCHS_ARG set to 0000 0000h when writing 0000 0000h into MMCHS_CMD register, the host controller performs a 'command completion signal disable' token that is, CMD line held to '0' during 47 cycles followed by a 1. 0h (R/W) = The host does not generate a 48-bit response instead of a command. 1h (R/W) = The host generates a 48-bit response instead of a command or a command completion signal disable token.
1	INIT	R/W	0h	Send initialization stream All cards: When this bit is set to 1, and the card is idle, an initialization sequence is sent to the card. An initialization sequence consists of setting the CMD line to 1 during 80 clock cycles. The initialisation sequence is mandatory - but it is not required to do it through this bit - this bit makes it easier. Clock divider (MMCHS_SYSCTL[CLKD]) should be set to ensure that 80 clock periods are greater than 1ms (see section 9.3, "Power-Up", in the MMC card specification, or section 6.4 in the SD card specification). **Note:** in this mode, there is no command sent to the card and no response is expected. 0h (R/W) = The host does not send an initialization sequence. 1h (R/W) = The host sends an initialization sequence.
0	OD	R/W	0h	Card open drain mode Only for MMC cards: This bit must be set to 1 for MMC card commands 1, 2, 3 and 40, and if the MMC card bus is operating in open-drain mode during the response phase to the command sent. Typically, during card identification mode when the card is either in idle, ready or ident state. It is also necessary to set this bit to 1, for a broadcast host response (see Broadcast host response register MMCHS_CON[HR]). 0h (R/W) = No Open Drain. 1h (R/W) = Open Drain or Broadcast host response.

Table 11-1881. Register Call Summary for MMCHS_CON

MMC/SD Registers • MMCHS_STAT Register (Offset = 230h) [reset = 0h]: [0][1][2] • MMCHS_PSTATE Register (Offset = 224h) [reset = 0---0000h]: [0][1][2][3][4][5] • MMCHS_IE Register (Offset = 234h) [reset = 0h]: [0][1] • MMCHS_ISE Register (Offset = 238h) [reset = 0h]: [0][1] • MMCHS_HCTL Register (Offset = 228h) [reset = 0h]: [0][1][2] • MMCHS_CON Register (Offset = 12Ch) [reset = 600h]: [0][1][2][3][4] • MMCHS_DATA Register (Offset = 220h) [reset = 0h]: [0] • MMC/SD Registers: [0]
MMC/SD Functional Description • MMC CE-ATA Command Completion Disable Management: [0][1][2] • Test Registers: [0] • DMA Modes: [0][1][2] • Interrupt Requests: [0] • Slave DMA Operations: [0]
MMC/SD • MMC/SD Overview: [0]
MMC/SD Programming Guide • Boot Mode Configuration: [0][1]

11.12.6.9 MMCHS_PWCNT Register (Offset = 130h) [reset = 0h]

MMCHS_PWCNT is shown in Figure 11-872 and described in Table 11-1883.

Power Counter Register
This register is used to program a MMC counter to delay command transfers after activating the PAD power, this value depends on PAD characteristics and voltage.

Table 11-1882. MMCHS_PWCNT Instances

Instance	Physical Address
MMCSD_0	2300 0130h
MMCSD_1	2310 0130h

Figure 11-872. MMCHS_PWCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PWRCNT															
R-0h																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1883. MMCHS_PWCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	PWRCNT	R/W	0h	Power counter register This bit field is used to introduce a delay between the PAD ACTIVE pin assertion and the command issued. 0h (R/W) = No additional delay added. 1h (R/W) = TCF delay (card clock period). 2h (R/W) = TCF x 2 delay (card clock period). FFFEh (R/W) = TCF x 65534 delay (card clock period). FFFFh (R/W) = TCF x 65535 delay (card clock period).

Table 11-1884. Register Call Summary for MMCHS_PWCNT

MMC/SD Registers • MMCHS_PWCNT Register (Offset = 130h) [reset = 0h]: [0] • MMC/SD Registers: [0]
MMC/SD Functional Description

11.12.6.10 MMCHS_DLL Register (Offset = 134h) [reset = 80000000h]

MMCHS_DLL is shown in Figure 11-873 and described in Table 11-1886.

DLL control and status register
This register is used for tuning procedure required for SDR104 speed mode.
It gives visibility and control on the DLL.

Note: DLL feature and SDR104/HS200 Mode are not supported.

Table 11-1885. MMCHS_DLL Instances

Instance	Physical Address
MMCSD_0	2300 0134h
MMCSD_1	2310 0134h

Figure 11-873. MMCHS_DLL Register

31	30	29	28	27	26	25	24
DLL_SOFT_RESET	LOCK_TIMER	MAX_LOCK_DIFF					
R/W-1h	R/W-0h	R/W-0h					
23	22	21	20	19	18	17	16
MAX_LOCK_DIFF		FORCE_SR_F	SWT	FORCE_SR_C			
R/W-0h		R/W-0h	R/W-0h	R/W-0h			
15	14	13	12	11	10	9	8
FORCE_SR_C			FORCE_VALUE	SLAVE_RATIO			
R/W-0h			R/W-0h	R/W-0h			
7	6	5	4	3	2	1	0
SLAVE_RATIO		RESERVED		DLL_UNLOCK_CLEAR	DLL_UNLOCK_STICKY	DLL_CALIB	DLL_LOCK
R/W-0h		R-0h		R/W-0h	R-0h	R/W-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1886. MMCHS_DLL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DLL_SOFT_RESET	R/W	1h	Soft reset for DLL, active HIGH 0h (W) = No action. 1h (W) = Issue soft reset. 0h (R) = Reset completed. 1h (R) = Reset is in progress.
30	LOCK_TIMER	R/W	0h	Timer for the DLL_LOCK signal to be asserted after reset 0h (R/W) = 1024 cycles (equivalent to DLL fast mode lock). 1h (R/W) = 66560 cycles.
29-22	MAX_LOCK_DIFF	R/W	0h	Maximum number of taps that the master DLL clock period measurement can deviate without resulting in the master DLL losing lock
21	FORCE_SR_F	R/W	0h	Forced fine delay value

Table 11-1886. MMCHS_DLL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	SWT	R/W	0h	Software Tuning enable The bit shall be set to manage the tuning sequence fully in software. **Note:** For proper operation when SDR104/HS200 mode is used this bit must be set to 1h which disables the Conflict Error (CFT Error) on the CMD line. 0h: No software tuning sequence. 1h: Execute software tuning sequence.
19-13	FORCE_SR_C	R/W	0h	Forced coarse delay value
12	FORCE_VALUE	R/W	0h	Put forced values to slave DLL, ignoring master DLL output and ratio value 0h (R/W) = Do not put force value. 1h (R/W) = Put force value.
11-6	SLAVE_RATIO	R/W	0h	Fraction of a clock cycle for the shift to be implemented, in units of 256ths of a clock cycle 0h (R/W) = 0 degree delay. 2h (R/W) = 45 degrees delay. 4h (R/W) = 90 degrees delay. 6h (R/W) = 135 degrees delay. 8h (R/W) = 180 degrees delay. Ah (R/W) = 225 degrees delay. Ch (R/W) = 270 degrees delay. Eh (R/W) = 315 degrees delay. 10h (R/W) = Full clock delay. 3Fh (R/W) = 4 clocks delay.
5-4	RESERVED	R	0h	Reserved
3	DLL_UNLOCK_CLEAR	R/W	0h	Clears the PHY_REG_STATUS_MDLL_UNLOCK_STICKY flags of the DLL 0h (R/W) = No effect. 1h (R/W) = Clears the flag.
2	DLL_UNLOCK_STICKY	R	0h	Asserted when any single period measurement exceeds MAX_LOCK_DIFF
1	DLL_CALIB	R/W	0h	Enables Slave DLL to update new delay values 0h (R/W) = Disabled. 1h (R/W) = Enabled.
0	DLL_LOCK	R	0h	Master DLL lock status 0h (R) = DLL is not locked. 1h (R) = DLL is locked.

Table 11-1887. Register Call Summary for MMCHS_DLL

MMC/SD Registers
- MMCHS_DLL Register (Offset = 134h) [reset = 80000000h]: [0]
- MMC/SD Registers: [0]

11.12.6.11 MMCHS_SDMASA Register (Offset = 200h) [reset = 0h]

MMCHS_SDMASA is shown in Figure 11-874 and described in Table 11-1889.

SDMA System Address / Argument 2 Register.

Table 11-1888. MMCHS_SDMASA Instances

Instance	Physical Address
MMCSD_0	2300 0200h
MMCSD_1	2310 0200h

Figure 11-874. MMCHS_SDMASA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDMA_ARG2																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1889. MMCHS_SDMASA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SDMA_ARG2	R/W	0h	SDMA System Address / Argument 2 This register contains the physical system memory address used for DMA transfers or the second argument for the Auto CMD23. (1) SDMA System Address This register contains the system memory address for a SDMA transfer. When the Host Controller stops a SDMA transfer, this register shall point to the system address of the next contiguous data position. It can be accessed only if no transaction is executing (after a transaction has stopped). Read operations during transfers may return an invalid value. The Host Driver shall initialize this register before starting a SDMA transaction. After SDMA has stopped, the next system address of the next contiguous data position can be read from this register. The SDMA transfer waits at the every boundary specified by the Host SDMA Buffer Boundary in the Block Size register. The Host Controller generates DMA Interrupt to request the Host Driver to update this register. The Host Driver sets the next system address of the next data position to this register. When the most upper byte of this register (003h) is written, the Host Controller restarts the SDMA transfer. When restarting SDMA by the Resume command or by setting Continue Request in the Block Gap Control register, the Host Controller shall start at the next contiguous address stored here in the SDMA System Address register. ADMA does not use this register. (2) Argument 2 This register is used with the Auto CMD23 to set a 32-bit block count value to the argument of the CMD23 while executing Auto CMD23. If Auto CMD23 is used with ADMA, the full 32-bit block count value can be used. If Auto CMD23 is used without AMDA, the available block count value is limited by the Block Count register. 65535 blocks is the maximum value in this case.

Table 11-1890. Register Call Summary for MMCHS_SDMASA

MMC/SD Registers
- MMCHS_SDMASA Register (Offset = 200h) [reset = 0h]: [0]
- MMCHS_CMD Register (Offset = 20Ch) [reset = 0h]: [0]
- MMC/SD Registers: [0]

11.12.6.12 MMCHS_BLK Register (Offset = 204h) [reset = 0h]

MMCHS_BLK is shown in Figure 11-875 and described in Table 11-1892.

Transfer Length Configuration Register
MMCHS_BLK[BLEN] is the block size register.
MMCHS_BLK[NBLK] is the block count register.
This register shall be used for any card.

Table 11-1891. MMCHS_BLK Instances

Instance	Physical Address
MMCSD_0	2300 0204h
MMCSD_1	2310 0204h

Figure 11-875. MMCHS_BLK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NBLK															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				BLEN											
R-0h				R/W-0h											

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1892. MMCHS_BLK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	NBLK	R/W	0h	Blocks count for current transfer This bit field is enabled when Block count Enable (MMCHS_CMD[BCE]) is set to 1 and is valid only for multiple block transfers. Setting the block count to 0 results no data blocks being transferred. **Note:** The host controller decrements the block count after each block transfer and stops when the count reaches zero. This bit field can be accessed only if no transaction is executing (that is, after a transaction has stopped). Read operations during transfers may return an invalid value and write operation will be ignored. In suspend context, the number of blocks yet to be transferred can be determined by reading this bit field. When restoring transfer context prior to issuing a Resume command, The local host shall restore the previously saved block count. 0h (R/W) = Stop count. 1h (R/W) = 1 block. 2h (R/W) = 2 blocks. FFFFh (R/W) = 65535 blocks.
15-12	RESERVED	R	0h	Reserved

Table 11-1892. MMCHS_BLK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11-0	BLEN	R/W	0h	Transfer Block Size This bit field specifies the block size for block data transfers. Read operations during transfers may return an invalid value, and write operations are ignored. When a CMD12 command is issued to stop the transfer, a read of the BLEN field after transfer completion (MMCHS_STAT[TC] set to 1) will not return the true byte number of data length while the stop occurs but the value written in this bit field before transfer is launched. 0h (R/W) = No data transfer. 1h (R/W) = 1 byte block length. 2h (R/W) = 2 bytes block length. 3h (R/W) = 3 bytes block length. 1FFh (R/W) = 511 bytes block length. 200h (R/W) = 512 bytes block length. 7FFh (R/W) = 2047 bytes block length. 800h (R/W) = 2048 bytes block length.

Table 11-1893. Register Call Summary for MMCHS_BLK

MMC/SD Registers • MMCHS_SYSTEST Register (Offset = 128h) [reset = 0h]: [0][1] • MMCHS_PSTATE Register (Offset = 224h) [reset = 0---0000h]: [0] • MMCHS_STAT Register (Offset = 230h) [reset = 0h]: [0][1] • MMCHS_CMD Register (Offset = 20Ch) [reset = 0h]: [0][1] • MMC/SD Registers: [0] • MMCHS_BLK Register (Offset = 204h) [reset = 0h]: [0][1][2]
MMC/SD Functional Description • DMA Transmit Mode: [0] • Data Buffer: [0] • Requirements for Descriptors: [0] • DMA Receive Mode: [0] • Interrupt Requests: [0][1]

11.12.6.13 MMCHS_ARG Register (Offset = 208h) [reset = 0h]

MMCHS_ARG is shown in Figure 11-876 and described in Table 11-1895.

Command Argument Register
This register contains command argument specified as bit 39-8 of Command-Format
These registers must be initialized prior to sending the command itself to the card (write action into the register MMCHS_CMD register). Only exception is for a command index specifying stuff bits in arguments, making a write unnecessary.

Table 11-1894. MMCHS_ARG Instances

Instance	Physical Address
MMCSD_0	2300 0208h
MMCSD_1	2310 0208h

Figure 11-876. MMCHS_ARG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARG																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1895. MMCHS_ARG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ARG	R/W	0h	Command argument bits [31-0]

Table 11-1896. Register Call Summary for MMCHS_ARG

MMC/SD Registers • MMCHS_CON Register (Offset = 12Ch) [reset = 600h]: [0] • MMCHS_ARG Register (Offset = 208h) [reset = 0h]: [0] • MMC/SD Registers: [0]
MMC/SD Functional Description • MMC CE-ATA Command Completion Disable Management: [0]

11.12.6.14 MMCHS_CMD Register (Offset = 20Ch) [reset = 0h]

MMCHS_CMD is shown in Figure 11-877 and described in Table 11-1898.

Command and Transfer Mode Register
MMCHS_CMD[31-16] = the command register
MMCHS_CMD[15-0] = the transfer mode.
This register configures the data and command transfers. A write into the most significant byte send the command. A write into MMCHS_CMD[15-0] registers during data transfer has no effect.
This register shall be used for any card.
Note: In SYSTEST mode, a write into MMCHS_CMD register will not start a transfer.

Table 11-1897. MMCHS_CMD Instances

Instance	Physical Address
MMCSD_0	2300 020Ch
MMCSD_1	2310 020Ch

Figure 11-877. MMCHS_CMD Register

31	30	29	28	27	26	25	24
RESERVED		INDX					
R-0h		R/W-0h					
23	22	21	20	19	18	17	16
CMD_TYPE		DP	CICE	CCCE	RESERVED	RSP_TYPE	
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		MSBS	DDIR	ACEN		BCE	DE
R-0h		R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1898. MMCHS_CMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved

Table 11-1898. MMCHS_CMD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
29-24	INDX	R/W	0h	Command index Binary encoded value from 0 to 63 specifying the command number send to card. 0h (R/W) = CMD0 or ACMD0. 1h (R/W) = CMD1 or ACMD1. 2h (R/W) = CMD2 or ACMD2. 3h (R/W) = CMD3 or ACMD3. 4h (R/W) = CMD4 or ACMD4. 5h (R/W) = CMD5 or ACMD5. 6h (R/W) = CMD6 or ACMD6. 7h (R/W) = CMD7 or ACMD7. 8h (R/W) = CMD8 or ACMD8. 9h (R/W) = CMD9 or ACMD9. Ah (R/W) = CMD10 or ACMD10. Bh (R/W) = CMD11 or ACMD11. Ch (R/W) = CMD12 or ACMD12. Dh (R/W) = CMD13 or ACMD13. Eh (R/W) = CMD14 or ACMD14. Fh (R/W) = CMD15 or ACMD15. 10h (R/W) = CMD16 or ACMD16. 11h (R/W) = CMD17 or ACMD17. 12h (R/W) = CMD18 or ACMD18. 13h (R/W) = CMD19 or ACMD19. 14h (R/W) = CMD20 or ACMD20. 15h (R/W) = CMD21 or ACMD21. 16h (R/W) = CMD22 or ACMD22. 17h (R/W) = CMD23 or ACMD23. 18h (R/W) = CMD24 or ACMD24. 19h (R/W) = CMD25 or ACMD25. 1Ah (R/W) = CMD26 or ACMD26. 1Bh (R/W) = CMD27 or ACMD27. 1Ch (R/W) = CMD28 or ACMD28. 1Dh (R/W) = CMD29 or ACMD29. 1Eh (R/W) = CMD30 or ACMD30. 1Fh (R/W) = CMD31 or ACMD31. 20h (R/W) = CMD32 or ACMD32. 21h (R/W) = CMD33 or ACMD33. 22h (R/W) = CMD34 or ACMD34. 23h (R/W) = CMD35 or ACMD35. 24h (R/W) = CMD36 or ACMD36. 25h (R/W) = CMD37 or ACMD37. 26h (R/W) = CMD38 or ACMD38. 27h (R/W) = CMD39 or ACMD39. 28h (R/W) = CMD40 or ACMD40. 29h (R/W) = CMD41 or ACMD41. 2Ah (R/W) = CMD42 or ACMD42. 2Bh (R/W) = CMD43 or ACMD43. 2Ch (R/W) = CMD44 or ACMD44. 2Dh (R/W) = CMD45 or ACMD45. 2Eh (R/W) = CMD46 or ACMD46. 2Fh (R/W) = CMD47 or ACMD47. 30h (R/W) = CMD48 or ACMD48. 31h (R/W) = CMD49 or ACMD49. 32h (R/W) = CMD50 or ACMD50. 33h (R/W) = CMD51 or ACMD51. 34h (R/W) = CMD52 or ACMD52.

Table 11-1898. MMCHS_CMD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				35h (R/W) = CMD53 or ACMD53. 36h (R/W) = CMD54 or ACMD54. 37h (R/W) = CMD55 or ACMD55. 38h (R/W) = CMD56 or ACMD56. 39h (R/W) = CMD57 or ACMD57. 3Ah (R/W) = CMD58 or ACMD58. 3Bh (R/W) = CMD59 or ACMD59. 3Ch (R/W) = CMD60 or ACMD60. 3Dh (R/W) = CMD61 or ACMD61. 3Eh (R/W) = CMD62 or ACMD62. 3Fh (R/W) = CMD63 or ACMD63.
23-22	CMD_TYPE	R/W	0h	Command type This bit field specifies three types of special command: Suspend, Resume and Abort. These bits shall be set to 00b for all other commands. 0h (R/W) = Others Commands. 1h (R/W) = CMD52 for writing "Bus Suspend" in CCCR. 2h (R/W) = CMD52 for writing "Function Select" in CCCR. 3h (R/W) = Abort command CMD12, CMD52 for writing " I/O Abort" in CCCR.
21	DP	R/W	0h	Data present select This bit field indicates that data is present and DAT line shall be used. It must be set to 0 in the following conditions: - command with no data transfer but using busy signal on DAT[0]. - command using only CMD line. - Resume command. 0h (R/W) = Command with no data transfer. 1h (R/W) = Command with data transfer.
20	CICE	R/W	0h	Command Index check enable This bit must be set to 1 to enable index check on command response to compare the index field in the response against the index of the command. If the index is not the same in the response as in the command, it is reported as a command index error (MMCHS_STAT[CIE] set to1). **Note:** The register CICE cannot be configured for an Auto CMD12, then index check is automatically checked when this command is issued. 0h (R/W) = Index check disable. 1h (R/W) = Index check enable.
19	CCCE	R/W	0h	Command CRC check enable This bit must be set to 1 to enable CRC7 check on command response to protect the response against transmission errors on the bus. If an error is detected, it is reported as a command CRC error (MMCHS_STAT[CCRC] set to 1). **Note:** The register CCCE cannot be configured for an Auto CMD12, and then CRC check is automatically checked when this command is issued. 0h (R/W) = CRC7 check disable. 1h (R/W) = CRC7 check enable.
18	RESERVED	R	0h	Reserved

Table 11-1898. MMCHS_CMD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17-16	RSP_TYPE	R/W	0h	Response type This bits defines the response type of the command. 0h (R/W) = No response. 1h (R/W) = Response Length 136 bits. 2h (R/W) = Response Length 48 bits. 3h (R/W) = Response Length 48 bits with busy after response.
15-6	RESERVED	R	0h	Reserved
5	MSBS	R/W	0h	Multi/Single block select This bit must be set to 1 for data transfer in case of multi block command. For any others command this bit shall be set to 0. 0h (R/W) = Single block. If this bit is 0, it is not necessary to set the register MMCHS_BLK[NBLK]. 1h (R/W) = Multi block. When Block Count is disabled (MMCHS_CMD[BCE] is set to 0) in Multiple block transfers (MMCHS_CMD[MSBS] is set to 1), the module can perform infinite transfer.
4	DDIR	R/W	0h	Data transfer Direction SelectThis bit defines either data transfer will be a read or a write 0h (R/W) = Data Write (host to card). 1h (R/W) = Data Read (card to host).
3-2	ACEN	R/W	0h	Auto CMD Enable - SD card only This field determines use of auto command functions. There are two methods to stop Multiple-block read and write operation. 1. Auto CMD12 Enable When this field is set to 01b, the Host Controller issues CMD12 automatically when last block transfer is completed. Auto CMD12 error is indicated to the Auto CMD Error Status register (MMCHS_AC12). The Host Driver shall not set this bit if the command does not require CMD12. In particular, secure commands defined in the Part 3 File Security specification do not require CMD12. 2. Auto CMD23 Enable When this bit field is set to 10b, the Host Controller issues a CMD23 automatically before issuing a command specified in the Command Register. The Host Controller Version 3.00 and later shall support this function. The following conditions are required to use the Auto CMD23. – Auto CMD23 Supported (Host Controller Version is 3.00 or later). – A memory card that supports CMD23 (SCR[33]=1). – If DMA is used, it shall be ADMA. – Only when CMD18 or CMD25 is issued. (**Note:** the Host Controller does not check command index). Auto CMD23 can be used with or without ADMA. By writing the Command register, the Host Controller issues a CMD23 first and then issues a command specified by the Command Index in Command register. If response errors of CMD23 are detected, the second command is not issued. A CMD23 error is indicated in the Auto CMD Error Status register (MMCHS_AC12). 32-bit block count value for CMD23 is set to SDMA System Address / Argument 2 register (MMCHS_SDMASA). 0h (R/W) = Auto Command Disabled. 1h (R/W) = Auto CMD12 enable or CCS detection enabled. 2h (R/W) = Auto CMD23 Enable. 3h (R/W) = Reserved.

Table 11-1898. MMCHS_CMD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	BCE	R/W	0h	Block Count Enable Multiple block transfers only. This bit is used to enable the block count register (MMCHS_BLK[NBLK]). When Block Count is disabled (MMCHS_CMD[BCE] is set to 0) in Multiple block transfers (MMCHS_CMD[MSBS] is set to 1), the module can perform infinite transfer. 0h (R/W) = Block count disabled for infinite transfer. 1h (R/W) = Block count enabled for multiple block transfer with known number of blocks.
0	DE	R/W	0h	DMA EnableThis bit is used to enable DMA mode for host data access 0h (R/W) = DMA mode disable. 1h (R/W) = DMA mode enable.

Table 11-1899. Register Call Summary for MMCHS_CMD

MMC/SD Registers • MMCHS_AC12 Register (Offset = 23Ch) [reset = 0h]: [0] • MMCHS_SYSTEST Register (Offset = 128h) [reset = 0h]: [0] • MMCHS_PSTATE Register (Offset = 224h) [reset = 0---0000h]: [0] • MMCHS_HCTL Register (Offset = 228h) [reset = 0h]: [0] • MMCHS_STAT Register (Offset = 230h) [reset = 0h]: [0][1] • MMCHS_CMD Register (Offset = 20Ch) [reset = 0h]: [0][1][2][3][4][5][6][7][8] • MMCHS_CON Register (Offset = 12Ch) [reset = 600h]: [0][1][2][3] • MMCHS_ARG Register (Offset = 208h) [reset = 0h]: [0] • MMC/SD Registers: [0] • MMCHS_BLK Register (Offset = 204h) [reset = 0h]: [0]
MMC/SD Functional Description • MMC CE-ATA Command Completion Disable Management: [0][1] • Requirements for Descriptors: [0] • Transfer Stop: [0] • Interrupt Requests: [0][1] • Data Buffer: [0] • Slave DMA Operations: [0] • Memory Size, Block Length, and Buffer-Management Relationship: [0][1]
MMC/SD Environment • Data Format: [0][1][2][3]
MMC/SD Programming Guide • Boot Mode Configuration: [0]

11.12.6.15 MMCHS_RSP10 Register (Offset = 210h) [reset = 0h]

MMCHS_RSP10 is shown in Figure 11-878 and described in Table 11-1901.

Command Response[31-0] Register (bits [31-0] of the internal RSP register)
This 32-bit register holds bits positions [31-0] of command response type R1/R1b/R2/R3/R4/R5/R5b/R6/R7.

Table 11-1900. MMCHS_RSP10 Instances

Instance	Physical Address
MMCSD_0	2300 0210h
MMCSD_1	2310 0210h

Figure 11-878. MMCHS_RSP10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSP1																RSP0															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1901. MMCHS_RSP10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RSP1	R	0h	Command Response [31:16]
15-0	RSP0	R	0h	Command Response [15:0]

Table 11-1902. Register Call Summary for MMCHS_RSP10

MMC/SD Registers • MMCHS_RSP10 Register (Offset = 210h) [reset = 0h]: [0] • MMCHS_CSRE Register (Offset = 124h) [reset = 0h]: [0][1] • MMC/SD Registers: [0]
MMC/SD Functional Description • Different Types of Responses: [0][1][2][3]

11.12.6.16 MMCHS_RSP32 Register (Offset = 214h) [reset = 0h]

MMCHS_RSP32 is shown in Figure 11-879 and described in Table 11-1904.

Command Response[63-32] Register (bits [63-32] of the internal RSP register)
This 32-bit register holds bits positions [63-32] of command response type R2.

Table 11-1903. MMCHS_RSP32 Instances

Instance	Physical Address
MMCSD_0	2300 0214h
MMCSD_1	2310 0214h

Figure 11-879. MMCHS_RSP32 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSP3																RSP2															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1904. MMCHS_RSP32 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RSP3	R	0h	Command Response [63:48]
15-0	RSP2	R	0h	Command Response [47:32]

Table 11-1905. Register Call Summary for MMCHS_RSP32

MMC/SD Registers • MMCHS_RSP32 Register (Offset = 214h) [reset = 0h]: [0] • MMC/SD Registers: [0]
MMC/SD Functional Description • Different Types of Responses: [0]

11.12.6.17 MMCHS_RSP54 Register (Offset = 218h) [reset = 0h]

MMCHS_RSP54 is shown in Figure 11-880 and described in Table 11-1907.

Command Response[95-64] Register (bits [95-64] of the internal RSP register)
This 32-bit register holds bits positions [95-64] of command response type R2.

Table 11-1906. MMCHS_RSP54 Instances

Instance	Physical Address
MMCSD_0	2300 0218h
MMCSD_1	2310 0218h

Figure 11-880. MMCHS_RSP54 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSP5																RSP4															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1907. MMCHS_RSP54 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RSP5	R	0h	Command Response [95:80]
15-0	RSP4	R	0h	Command Response [79:64]

Table 11-1908. Register Call Summary for MMCHS_RSP54

MMC/SD Registers • MMCHS_RSP54 Register (Offset = 218h) [reset = 0h]: [0] • MMC/SD Registers: [0]
MMC/SD Functional Description • Different Types of Responses: [0]

11.12.6.18 MMCHS_RSP76 Register (Offset = 21Ch) [reset = 0h]

MMCHS_RSP76 is shown in Figure 11-881 and described in Table 11-1910.

Command Response[127:96] Register (bits [127:96] of the internal RSP register)
This 32-bit register holds bits positions [127:96] of command response type R1(Auto CMD23)/R1b(Auto CMD12)/R2.

Table 11-1909. MMCHS_RSP76 Instances

Instance	Physical Address
MMCSD_0	2300 021Ch
MMCSD_1	2310 021Ch

Figure 11-881. MMCHS_RSP76 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSP7																RSP6															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1910. MMCHS_RSP76 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RSP7	R	0h	Command Response [127:112]
15-0	RSP6	R	0h	Command Response [111:96]

Table 11-1911. Register Call Summary for MMCHS_RSP76

MMC/SD Registers • MMCHS_RSP76 Register (Offset = 21Ch) [reset = 0h]: [0] • MMCHS_STAT Register (Offset = 230h) [reset = 0h]: [0] • MMCHS_CSRE Register (Offset = 124h) [reset = 0h]: [0] • MMC/SD Registers: [0]
MMC/SD Functional Description • Different Types of Responses: [0][1][2][3]

11.12.6.19 MMCHS_DATA Register (Offset = 220h) [reset = 0h]

MMCHS_DATA is shown in Figure 11-882 and described in Table 11-1913.

Data Register
This register is the 32-bit entry point of the buffer for read or write data transfers.
The buffer size is 32bits x256(1024 bytes). Bytes within a word are stored and read in little endian format. This buffer can be used as two 512 byte buffers to transfer data efficiently without reducing the throughput.
Sequential and contiguous access is necessary to increment the pointer correctly. Random or skipped access is not allowed. In little endian, if the local host accesses this register byte-wise or 16bit-wise, the least significant byte (bits [7-0]) must always be written/read first. The update of the buffer address is done on the most significant byte write for full 32-bit DATA register or on the most significant byte of the last word of block transfer.
Example 1: Byte or 16-bit access
Mbyteen[3-0]=0001 (1-byte) => Mbyteen[3-0]=0010 (1-byte) => Mbyteen[3-0]=1100 (2-bytes) OK
Mbyteen[3-0]=0001 (1-byte) => Mbyteen[3-0]=0010 (1-byte) => Mbyteen[3-0]=0100 (1-byte) OK
Mbyteen[3-0]=0001 (1-byte) => Mbyteen[3-0]=0010 (1-byte) => Mbyteen[3-0]=1000 (1-byte) Bad.

Table 11-1912. MMCHS_DATA Instances

Instance	Physical Address
MMCSD_0	2300 0220h
MMCSD_1	2310 0220h

Figure 11-882. MMCHS_DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1913. MMCHS_DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	Data Register [31-0] In functional mode (MMCHS_CON[MODE] set to the default value 0). A read access to this register is allowed only when the buffer read enable status is set to 1 (MMCHS_PSTATE[BRE]), otherwise a bad access (MMCHS_STAT[BADA]) is signaled. A write access to this register is allowed only when the buffer write enable status is set to 1 (MMCHS_PSTATE[BWE]), otherwise a bad access (MMCHS_STAT[BADA]) is signaled and the data is not written.

Table 11-1914. Register Call Summary for MMCHS_DATA

MMC/SD Registers • MMCHS_STAT Register (Offset = 230h) [reset = 0h]: [0][1] • MMCHS_HCTL Register (Offset = 228h) [reset = 0h]: [0] • MMCHS_CON Register (Offset = 12Ch) [reset = 600h]: [0][1] • MMCHS_DATA Register (Offset = 220h) [reset = 0h]: [0] • MMC/SD Registers: [0] • MMCHS_SYSCTL Register (Offset = 22Ch) [reset = 0h]: [0]
MMC/SD Functional Description • Data Buffer: [0][1][2][3][4][5][6][7] • Transfer or Command Status and Errors Reporting: [0] • Interrupt Requests: [0][1]

11.12.6.20 MMCHS_PSTATE Register (Offset = 224h) [reset = 0---0000h]

MMCHS_PSTATE is shown in Figure 11-883 and described in Table 11-1916.

Present State Register
The Host can get status of the Host Controller from this 32-bit read only register.

Table 11-1915. MMCHS_PSTATE Instances

Instance	Physical Address
MMCSD_0	2300 0224h
MMCSD_1	2310 0224h

Figure 11-883. MMCHS_PSTATE Register

31	30	29	28	27	26	25	24
RESERVED							CLEV
R-0h							R--h
23	22	21	20	19	18	17	16
DLEV				WP	CDPL	CSS	CINS
R--h				R--h	R--h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED				BRE	BWE	RTA	WTA
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED				RTR	DLA	DATI	CMDI
R-0h				R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1916. MMCHS_PSTATE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24	CLEV	R	-h	CMD line signal level This status is used to check the CMD line level to recover from errors, and for debugging. The value of this bit field after reset depends on the CMD line level at that time. 0h (R) = The CMD line level is 0. 1h (R) = The CMD line level is 1.
23-20	DLEV	R	-h	DAT[3:0] line signal level DAT[3] => bit 23 DAT[2] => bit 22 DAT[1] => bit 21 DAT[0] => bit 20 This status is used to check DAT line level to recover from errors, and for debugging. This is especially useful in detecting the busy signal level from DAT[0]. The value of these registers after reset depends on the DAT lines level at that time.

Table 11-1916. MMCHS_PSTATE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	WP	R	-h	Write protect switch pin level For SDIO cards only. This bit reflects the write protect input pin (MMCi_SDWP) level. The value of this bit field after reset depends on the protect input pin (MMCi_SDWP) level at that time. 0h (R) = If MMCHS_CON[WPP] is set to 0 (default), the card is write protected, otherwise the card is not protected. 1h (R) = If MMCHS_CON[WPP] is set to 0 (default), the card is not write protected, otherwise the card is protected.
18	CDPL	R	-h	Card detect pin level This bit reflects the inverse value of the card detect input pin (MMCi_SDCD), debouncing is not performed on this bit and bit is valid only when Card State Stable MMCHS_PSTATE[CSS]) is set to 1. Use of this bit is limited to testing since it must be debounced y software. The value of this bit field after reset depends on the card detect input pin (MMCi_SDCD) level at that time. 0h (R) = The value of the card detect input pin (MMCi_SDCD) is 1. 1h (R) = The value of the card detect input pin (MMCi_SDCD) is 0.
17	CSS	R	0h	Card State Stable This bit is used for testing. It is set to 1 only when Card Detect Pin Level is stable (MMCHS_PSTATE[CDPL]). Debouncing is performed on the card detect input pin (MMCi_SDCD) to detect card stability. 0h (R) = Reset or Debouncing. 1h (R) = No card or card inserted. **Note:** This bit is not affected by a software reset.
16	CINS	R	0h	Card inserted This bit is the debounced value of the card detect input pin (MMCi_SDCD). An inactive to active transition of the card detect input pin (MMCi_SDCD) will generate a card insertion interrupt (MMCHS_STAT[CINS]). A active to inactive transition of the card detect input pin (MMCi_SDCD) will generate a card removal interrupt (MMCHS_STAT[REM]). 0h (R) = If MMCHS_CON[CDP] is set to 1, no card is detected. The card may have been removed from the card slot. If MMCHS_CON[CDP] is set to 0, the card has been inserted. 1h (R) = If MMCHS_CON[CDP] is set to 1, the card has been inserted from the card slot. If MMCHS_CON[CDP] is set to 0, no card is detected. The card may have been removed from the card slot. **Note:** This bit is not affected by a software reset.
15-12	RESERVED	R	0h	Reserved

Table 11-1916. MMCHS_PSTATE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	BRE	R	0h	Buffer read enable This bit is used for non-DMA read transfers. It indicates that a complete block specified by MMCHS_BLK[BLEN] has been written in the buffer and is ready to be read. It is set to 0 when the entire block is read from the buffer. It is set to 1 when a block data is ready in the buffer and generates the Buffer read ready status of interrupt (MMCHS_STAT[BRR]). 0h (R) = Read BLEN bytes disable. 1h (R) = Read BLEN bytes enable. Readable data exists in the buffer.
10	BWE	R	0h	Buffer Write enable This status is used for non-DMA write transfers. It indicates if space is available for write data. 0h (R) = There is no room left in the buffer to write BLEN bytes of data. 1h (R) = There is enough space in the buffer to write BLEN bytes of data.
9	RTA	R	0h	Read transfer active This status is used for detecting completion of a read transfer. It is set to 1 after the end bit of read command or by activating a continue request (MMCHS_HCTL[CR]) following a stop at block gap request. This bit is set to 0 when all data have been read by the local host after last block or after a stop at block gap request. 0h (R) = No valid data on the DAT lines. 1h (R) = read data transfer on going.
8	WTA	R	0h	Write transfer active This status indicates a write transfer active. It is set to 1 after the end bit of write command or by activating a continue request (MMCHS_HCTL[CR]) following a stop at block gap request. This bit is set to 0 when CRC status has been received after last block or after a stop at block gap request. 0h (R) = No valid data on the DAT lines. 1h (R) = Write data transfer on going.
7-4	RESERVED	R	0h	Reserved
3	RTR	R	0h	Re-Tuning Request Host Controller may request Host Driver to execute re-tuning sequence by setting this bit when the data window is shifted by temperature drift and a tuned sampling point does not have a good margin to receive correct data. This bit is cleared when a command is issued with setting MMCHS_AC12[22] ET. This bit isn't set to 1 if MMCHS_AC12[23] SCLK_SEL is set to 0 (using fixed sampling clock). Refer to MMCHS_CAPA2[15-14] RTM for more detail. 0h (R) = Fixed or well tuned sampling clock. 1h (R) = Sampling clock needs re-tuning.

Table 11-1916. MMCHS_PSTATE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	DLA	R	0h	DAT line active This status bit indicates whether one of the DAT line is in use. In the case of read transactions (card to host): This bit is set to 1 after the end bit of read command or by activating continue request MMCHS_HCTL[CR]. This bit is set to 0 when the host controller received the end bit of the last data block or at the beginning of the read wait mode. In the case of write transactions (host to card): This bit is set to 1 after the end bit of write command or by activating continue request MMCHS_HCTL[CR]. This bit is set to 0 on the end of busy event for the last block; host controller must wait 8 clock cycles with line not busy to really consider not "busy state" or after the busy block as a result of a stop at gap request. 0h (R) = DAT Line inactive. 1h (R) = DAT Line active.
1	DATI	R	0h	Command inhibit (DAT) This status bit is generated if either DAT line is active (MMCHS_PSTATE[DLA]) or Read transfer is active (MMCHS_PSTATE[RTA]) or when a command with busy is issued. This bit prevents the local host to issue a command. A change of this bit from 1 to 0 generates a transfer complete interrupt (MMCHS_STAT[TC]). 0h (R) = Issuing of command using the DAT lines is allowed. 1h (R) = Issuing of command using DAT lines is not allowed.
0	CMDI	R	0h	Command inhibit (CMD) This status bit indicates that the CMD line is in use. This bit is set to 0 when the most significant byte is written into the command register. This bit is not set when Auto CMD12 is transmitted. This bit is set to 0 in either the following cases: - After the end bit of the command response, excepted if there is a command conflict error (MMCHS_STAT[CCRC] or MMCHS_STAT[CEB] set to 1) or a Auto CMD12 is not executed (MMCHS_AC12[ACNE]). - After the end bit of the command without response (MMCHS_CMD[RSP_TYPE] set to "00") In case of a command data error is detected (MMCHS_STAT[CTO] set to 1), this bit field is not automatically cleared. 0h (R) = Issuing of command using CMD line is allowed. 1h (R) = Issuing of command using CMD line is not allowed.

Table 11-1917. Register Call Summary for MMCHS_PSTATE

MMC/SD Registers

- MMCHS_STAT Register (Offset = 230h) [reset = 0h]: [0][1][2][3][4][5][6][7][8]
- MMCHS_PSTATE Register (Offset = 224h) [reset = 0---0000h]: [0][1][2][3][4]
- MMCHS_HCTL Register (Offset = 228h) [reset = 0h]: [0][1]
- MMCHS_DATA Register (Offset = 220h) [reset = 0h]: [0][1]
- MMC/SD Registers: [0]
- MMCHS_SYSCTL Register (Offset = 22Ch) [reset = 0h]: [0][1]

Table 11-1917. Register Call Summary for MMCHS_PSTATE (continued)

MMC/SD Functional Description
• Test Registers: [0] • Requirements for Descriptors: [0][1][2] • Interrupt Requests: [0][1][2][3][4][5][6] • Data Buffer: [0][1] • Data Buffer Status: [0][1] • MMC Hardware Status Features: [0][1][2][3][4][5][6][7][8][9][10][11][12][13] • Software Reset: [0]

11.12.6.21 MMCHS_HCTL Register (Offset = 228h) [reset = 0h]

MMCHS_HCTL is shown in Figure 11-884 and described in Table 11-1919.

Host Control Register
This register defines the host controls to set power, wakeup and transfer parameters.
MMCHS_HCTL[31-24] = Wakeup control
MMCHS_HCTL[23-16] = Block gap control
MMCHS_HCTL[15-8] = Power control
MMCHS_HCTL[7-0] = Host control.

Table 11-1918. MMCHS_HCTL Instances

Instance	Physical Address
MMCSD_0	2300 0228h
MMCSD_1	2310 0228h

Figure 11-884. MMCHS_HCTL Register

31	30	29	28	27	26	25	24
RESERVED				OBWE	REM	INS	IWE
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED				IBG	RWC	CR	SBGR
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED				SDVS			SDBP
R-0h				R/W-0h			R/W-0h
7	6	5	4	3	2	1	0
CDSS	CDTL	RESERVED	DMAS		HSPE	DTW	LED
R/W-0h	R/W-0h	R-0h	R/W-0h		R/W-0h	R/W-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1919. MMCHS_HCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27	OBWE	R/W	0h	Wakeup event enable for 'Out-of-Band' Interrupt This bit enables wakeup events for 'Out-of-Band' assertion. Wakeup is generated if the wakeup feature is enabled (MMCHS_SYSCONFIG[ENAWAKEUP]). The write to this bit field is ignored when MMCHS_CON[OBIE] is not set. 0h (R/W) = Disable wakeup on 'Out-of-Band' Interrupt. 1h (R/W) = Enable wakeup on 'Out-of-Band' Interrupt. **Note:** Out-Of-Band Interrupt feature is not supported. **Note:** Wakeup feature is not supported.
26	REM	R/W	0h	Wakeup event enable on SD card removal This bit enables wakeup events for card removal assertion. Wakeup is generated if the wakeup feature is enabled (MMCHS_SYSCONFIG[ENAWAKEUP]). 0h (R/W) = Disable wakeup on card removal. 1h (R/W) = Enable wakeup on card removal. **Note:** Wakeup feature is not supported.

Table 11-1919. MMCHS_HCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	INS	R/W	0h	Wakeup event enable on SD card insertion This bit enables wakeup events for card insertion assertion. Wakeup is generated if the wakeup feature is enabled (MMCHS_SYSCONFIG[ENAWAKEUP]). 0h (R/W) = Disable wakeup on card insertion. 1h (R/W) = Enable wakeup on card insertion. **Note:** Wakeup feature is not supported.
24	IWE	R/W	0h	Wakeup event enable on SD card interrupt This bit enables wakeup events for card interrupt assertion. Wakeup is generated if the wakeup feature is enabled (MMCHS_SYSCONFIG[ENAWAKEUP]). 0h (R/W) = Disable wakeup on card interrupt. 1h (R/W) = Enable wakeup on card interrupt. **Note:** Wakeup feature is not supported.
23-20	RESERVED	R	0h	Reserved
19	IBG	R/W	0h	Interrupt block at gap This bit is valid only in 4-bit mode of SDIO card to enable interrupt detection in the interrupt cycle at block gap for a multiple block transfer. For MMC cards and for SD card this bit should be set to 0. 0h (R/W) = Disable interrupt detection at the block gap in 4-bit mode. 1h (R/W) = Enable interrupt detection at the block gap in 4-bit mode.
18	RWC	R/W	0h	Read wait control The read wait function is optional only for SDIO cards. If the card supports read wait, this bit must be enabled, then requesting a stop at block gap (MMCHS_HCTL[SBGR]) generates a read wait period after the current end of block. Be careful, if read wait is not supported it may cause a conflict on DAT line. 0h (R/W) = Disable Read Wait Control. Suspend/Resume cannot be supported. 1h (R/W) = Enable Read Wait Control.
17	CR	R/W	0h	Continue request This bit is used to restart a transaction that was stopped by requesting a stop at block gap (MMCHS_HCTL[SBGR]). Set this bit to 1 restarts the transfer. The bit is automatically set to 0 by the host controller when transfer has restarted that is DAT line is active (MMCHS_PSTATE[DLA]) or transferring data (MMCHS_PSTATE[WTA]). The Stop at block gap request must be disabled (MMCHS_HCTL[SBGR]=0) before setting this bit. 0h (R/W) = No affect. 1h (R/W) = transfer restart.
16	SBGR	R/W	0h	Stop at block gap request This bit is used to stop executing a transaction at the next block gap. The transfer can restart with a continue request (MMCHS_HCTL[CR]) or during a suspend/resume sequence. In case of read transfer, the card must support read wait control. In case of write transfer, the host driver shall set this bit after all block data written. Until the transfer completion (MMCHS_STAT[TC] set to 1), the host driver shall leave this bit set to 1. If this bit is set, the local host shall not write to the data register (MMCHS_DATA). 0h (R/W) = Transfer mode. 1h (R/W) = Stop at block gap.

Table 11-1919. MMCHS_HCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-9	SDVS	R/W	0h	SD bus voltage select All cards. The host driver should set to these bits to select the voltage level for the card according to the voltage supported by the system (MMCHS_CAPA[VS18,VS30,VS33]) before starting a transfer. 5h (R/W) = 1.8 V (Typical). 6h (R/W) = 3.0 V (Typical). 7h (R/W) = 3.3 V (Typical).
8	SDBP	R/W	0h	SD bus power Before setting this bit, the host driver shall select the SD bus voltage (MMCHS_HCTL[SDVS]). If the host controller detects the No card state, this bit is automatically set to 0. If the module is power off, a write in the command register (MMCHS_CMD) will not start the transfer. A write to this bit has no effect if the selected SD bus voltage MMCHS_HCTL[SDVS] is not supported according to capability register (MMCHS_CAPA[VS*]). 0h (R/W) = Power off. 1h (R/W) = Power on.
7	CDSS	R/W	0h	Card Detect Signal Selection This bit selects source for the card detection. When the source for the card detection is switched, the interrupt should be disabled during the switching period by clearing the Interrupt Status/Signal Enable register in order to mask unexpected interrupts caused by the glitch. The Interrupt Status/Signal Enable should be disabled during over the period of debouncing. 0h (R/W) = MMCi_SDCD is selected (for normal use). 1h (R/W) = MMCHS_HCTL[6] CDTL is selected (for test purpose).
6	CDTL	R/W	0h	Card Detect Test Level This bit is enabled while MMCHS_HCTL[7] CDSS is set to 1 and it indicates whether the card is inserted or not. 0h (R/W) = No Card. 1h (R/W) = Card Inserted.
5	RESERVED	R	0h	Reserved
4-3	DMAS	R/W	0h	DMA Select Mode One of supported DMA modes can be selected. The host driver shall check support of DMA modes by referring the Capabilities register MMCHS_CAPA. Use of selected DMA is determined by DMA Enable of the Transfer Mode register. This bit field is only meaningful when MADMA_EN is set to 1. When MADMA_EN is set to 0 the bit field is read only and returned value is 0. 0h (R/W) = Reserved. 1h (R/W) = Reserved. 2h (R/W) = 32-bit Address ADMA2 is selected. 3h (R/W) = Reserved.
2	HSPE	R/W	0h	High Speed Enable: Before setting this bit, the Host Driver shall check the MMCHS_CAPA[21] HSS. This bit shall not be set when dual data rate mode is activated in MMCHS_CON[19] DDR. 0h (R/W) = The Host Controller outputs CMD line and DAT lines at the falling edge of the SD Clock. 1h (R/W) = The Host Controller outputs CMD line and DAT lines at the rising edge of the SD Clock. **Note:** Do not set this bit to 0x1 because device was timing closed with HSPE bit set to 0x0 for all supported modes of operation.

Table 11-1919. MMCHS_HCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	DTW	R/W	0h	Data transfer width For MMC card, this bit must be set following a valid SWITCH command (CMD6) with the correct value and extend CSD index written in the argument. Prior to this command, the MMC card configuration register (CSD and EXT_CSD) must be verified for compliance with MMC standard specification 4.x (see section 3.6). This bit field has no effect when the MMC 8-bit mode is selected (register MMCHS_CON[DW8] set to1), For SD/SDIO cards, this bit must be set following a valid SET_BUS_WIDTH command (ACMD6) with the value written in bit 1 of the argument. Prior to this command, the SD card configuration register (SCR) must be verified for the supported bus width by the SD card. 0h (R/W) = 1-bit Data width (DAT[0] used). 1h (R/W) = 4-bit Data width (DAT[3:0] used).
0	LED	R	0h	Reserved bit LED control feature is not supported. This bit is initialized to zero, and writes to it are ignored.

Table 11-1920. Register Call Summary for MMCHS_HCTL

MMC/SD Registers • MMCHS_STAT Register (Offset = 230h) [reset = 0h]: [0][1] • MMCHS_PSTATE Register (Offset = 224h) [reset = 0---0000h]: [0][1][2][3] • MMCHS_REV Register (Offset = 2FCh) [reset = -020000h]: [0] • MMCHS_HCTL Register (Offset = 228h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12] • MMCHS_CON Register (Offset = 12Ch) [reset = 600h]: [0][1] • MMC/SD Registers: [0] • MMCHS_SYSCTL Register (Offset = 22Ch) [reset = 0h]: [0]
MMC/SD Functional Description • Test Registers: [0][1] • Generation on Falling Edge of MMC Clock: [0] • Output Signals Generation: [0] • Interrupt Requests: [0][1] • Requirements for Descriptors: [0]
MMC/SD Programming Guide

11.12.6.22 MMCHS_SYSCTL Register (Offset = 22Ch) [reset = 0h]

MMCHS_SYSCTL is shown in Figure 11-885 and described in Table 11-1922.

SD System Control Register
This register defines the system controls to set software resets, clock frequency management and data timeout.
MMCHS_SYSCTL[31-24] = Software resets
MMCHS_SYSCTL[23-16] = Timeout control
MMCHS_SYSCTL[15-0] = Clock control.

Table 11-1921. MMCHS_SYSCTL Instances

Instance	Physical Address
MMCSD_0	2300 022Ch
MMCSD_1	2310 022Ch

Figure 11-885. MMCHS_SYSCTL Register

31	30	29	28	27	26	25	24
RESERVED					SRD	SRC	SRA
R-0h					R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED					DTO		
R-0h					R/W-0h		
15	14	13	12	11	10	9	8
CLKD							
R/W-0h							
7	6	5	4	3	2	1	0
CLKD		CGS	RESERVED		CEN	ICS	ICE
R/W-0h		R-0h	R-0h		R/W-0h	R-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1922. MMCHS_SYSCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	Reserved
26	SRD	R/W	0h	Software reset for DAT line This bit is set to 1 for reset and released to 0 when completed. For more information about SRD bit manipulation, see Section 11.12.5.1.2.1.2.1, *DATA Lines Reset Procedure*. DAT finite state machine in both clock domain are also reset. Here below are the registers cleared by MMCHS_SYSCTL[SRD]: - MMCHS_DATA - MMCHS_PSTATE: BRE, BWE, RTA, WTA, DLA and DATI - MMCHS_HCTL: SBGR and CR - MMCHS_STAT: BRR, BWR, BGE and TC Interconnect and MMC buffer data management is reinitialized. 0h (R/W) = Reset completed. 1h (R/W) = Software reset for DAT line.

Table 11-1922. MMCHS_SYSCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	SRC	R/W	0h	Software reset for CMD line For more information about SRC bit manipulation, see Section 11.12.5.1.2.1.1.1, *CMD Line Reset Procedure*. This bit is set to 1 for reset and released to 0 when completed. CMD finite state machine in both clock domain are also reset. Here below the registers cleared by MMCHS_SYSCTL[SRC]: - MMCHS_PSTATE: CMDI - MMCHS_STAT: CC Interconnect and MMC command status management is reinitialized. 0h (R/W) = Reset completed. 1h (R/W) = Software reset for CMD line.
24	SRA	R/W	0h	Software reset for all This bit is set to 1 for reset and released to 0 when completed. This reset affects the entire host controller except for the card detection circuit andcapabilities registers (MMCHS_CAPA and MMCHS_CUR_CAPA). 0h (R/W) = Reset completed. 1h (R/W) = Software reset for all the design.
23-20	RESERVED	R	0h	Reserved
19-16	DTO	R/W	0h	Data timeout counter value and busy timeout This value determines the interval by which DAT lines timeouts are detected. The host driver needs to set this bitfield based on - the maximum read access time (NAC) (Refer to the SD Specification Part1 Physical Layer), - the data read access time values (TAAC and NSAC) in the card specific data register (CSD) of the card, - the timeout clock base frequency (MMCHS_CAPA[TCF]). If the card does not respond within the specified number of cycles, a data timeout error occurs (MMCHS_STAT[DTO]). The MMCHS_SYSCTL[DTO] register is also used to check busy duration, to generate busy timeout for commands with busy response or for busy programming during a write command. Timeout on CRC status is generated if no CRC token is present after a block write. 0h (R/W) = TCF x 2^13. 1h (R/W) = TCF x 2^14. Eh (R/W) = TCF x 2^27. Fh (R/W) = Reserved.
15-6	CLKD	R/W	0h	Clock frequency select These bits define the ratio between MMC_CLK_ADPI and the output clock frequency on the CLK pin of either the memory card (MMC, SD or SDIO). 0h (R/W) = MMC_CLK_ADPI bypass. 1h (R/W) = MMC_CLK_ADPI bypass. 2h (R/W) = MMC_CLK_ADPI / 2. 3h (R/W) = MMC_CLK_ADPI / 3. 3FFh (R/W) = MMC_CLK_ADPI / 1023.

Table 11-1922. MMCHS_SYSCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	CGS	R	0h	Clock Generator Select - For SD cards Host Controller Version 3.00 supports this bit. This bit is used to select the clock generator mode in MMCHS_SYSCTL[15-6] CLKD. If the Programmable Clock Mode is supported (non-zero value is set to MMCHS_CAPA2[23-16] CM), this bit attribute is RW, and if not supported, this bit attribute is RO and zero is read. This bit depends on the setting of MMCHS_AC12[31] PV_ENABLE. If PV_ENABLE = 0, this bit is set by Host Driver. If PV_ENABLE = 1, this bit is automatically set to a value specified in one of Preset Value registers, see, Table 11-1839.
4-3	RESERVED	R	0h	Reserved
2	CEN	R/W	0h	Clock enable This bit controls if the clock is provided to the card or not. 0h (R/W) = The clock is not provided to the card. Clock frequency can be changed. 1h (R/W) = The clock is provided to the card and can be automatically gated when MMCHS_SYSCONFIG[AUTOIDLE] is set to 1 (default value). The host driver shall wait to set this bit to 1 until the Internal clock is stable (MMCHS_SYSCTL[ICS]).
1	ICS	R	0h	Internal clock stable (status) This bit indicates either the internal clock is stable or not. 0h (R) = The internal clock is not stable. 1h (R) = The internal clock is stable after enabling the clock (MMCHS_SYSCTL[ICE]) or after changing the clock ratio (MMCHS_SYSCTL[CLKD]).
0	ICE	R/W	0h	Internal clock enable This bit field controls the internal clock activity. In very low power state, the internal clock is stopped. **Note:** The activity of the debounce clock (used for wakeup events) and the interface clock (used for reads and writes to the module register map) are not affected by this bit field. 0h (R/W) = The internal clock is stopped (very low power state). 1h (R/W) = The internal clock oscillates and can be automatically gated when MMCHS_SYSCONFIG[AUTOIDLE] is set to 1 (default value).

Table 11-1923. Register Call Summary for MMCHS_SYSCTL

MMC/SD Registers • MMCHS_AC12 Register (Offset = 23Ch) [reset = 0h]: [0][1][2][3][4][5][6][7] • MMCHS_CAPA2 Register (Offset = 244h) [reset = F77h]: [0] • MMCHS_PVINITSD Register (Offset = 260h) [reset = 401E0h]: [0][1] • MMCHS_PVSDR104DDR50 Register (Offset = 26Ch) [reset = 20000h]: [0][1] • MMCHS_PVSDR25SDR50 Register (Offset = 268h) [reset = 10002h]: [0][1] • MMCHS_CON Register (Offset = 12Ch) [reset = 600h]: [0][1][2][3] • MMCHS_PVHSSDR12 Register (Offset = 264h) [reset = 40002h]: [0][1] • MMCHS_CAPA Register (Offset = 240h) [reset = 20E10080h]: [0] • MMC/SD Registers: [0] • MMCHS_SYSCTL Register (Offset = 22Ch) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10]
MMC/SD Functional Description • Software Reset: [0][1][2][3][4]
MMC/SD Environment • Protocol: [0]
MMC/SD Programming Guide • Basic Operations for MMC Host Controller: [0][1][2][3][4][5]

11.12.6.23 MMCHS_STAT Register (Offset = 230h) [reset = 0h]

MMCHS_STAT is shown in Figure 11-886 and described in Table 11-1925.

Interrupt Status Register
The interrupt status regroups all the status of the module internal events that can generate an interrupt.
MMCHS_STAT[31-16] = Error Interrupt Status
MMCHS_STAT[15-0] = Normal Interrupt Status.

Table 11-1924. MMCHS_STAT Instances

Instance	Physical Address
MMCSD_0	2300 0230h
MMCSD_1	2310 0230h

Figure 11-886. MMCHS_STAT Register

31	30	29	28	27	26	25	24
RESERVED		BADA	CERR	RESERVED	TE	ADMAE	ACE
R-0h		R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	DEB	DCRC	DTO	CIE	CEB	CCRC	CTO
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
ERRI	RESERVED				BSR	OBI	CIRQ
R-0h	R-0h				R/W-0h	R/W-0h	R-0h
7	6	5	4	3	2	1	0
CREM	CINS	BRR	BWR	DMA	BGE	TC	CC
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1925. MMCHS_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29	BADA	R/W	0h	Bad access to data space This bit is set automatically to indicate a bad access to buffer when not allowed: -This bit is set during a read access to the data register (MMCHS_DATA) while buffer reads are not allowed (MMCHS_PSTATE[BRE] =0) -This bit is set during a write access to the data register (MMCHS_DATA) while buffer writes are not allowed (MMCHS_STATE[BWE] =0). 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = No Interrupt. 1h (R) = Bad Access.

Table 11-1925. MMCHS_STAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
28	CERR	R/W	0h	Card error This bit is set automatically when there is at least one error in a response of type R1, R1b, R6, R5 or R5b. Only bits referenced as type E(error) in status field in the response can set a card status error. An error bit in the response is flagged only if corresponding bit in card status response error MMCHS_CSRE in set. There is no card error detection for auto CMD12 command. The host driver shall read MMCHS_RSP76 register to detect error bits in the command response. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = No Error. 1h (R) = Card error.
27	RESERVED	R	0h	Reserved
26	TE	R/W	0h	Tuning Error This bit is set when an unrecoverable error is detected in a tuning circuit except during tuning procedure (Occurrence of an error during tuning procedure is indicated by Sampling Select). By detecting Tuning Error, Host Driver needs to abort a command executing and perform tuning. To reset tuning circuit, Sampling Clock shall be set to 0 before executing tuning procedure. The Tuning Error is higher priority than the other error interrupts generated during data transfer. By detecting Tuning Error, the Host Driver should discard data transferred by a current read/write command and retry data transfer after the Host Controller retrieved from tuning circuit error. The bit is set if the lock is lost (but not during the tuning process) or if the lock counter expires without the lock being asserted. If the latter happens, the SW can decide to ignore the interrupt and wait some more for the lock to be set. 0h (R/W) = No Error. 1h (R/W) = Error.
25	ADMAE	R/W	0h	ADMA Error This bit is set when the Host Controller detects errors during ADMA based data transfer. The state of the ADMA at an error occurrence is saved in the ADMA Error Status Register. In addition, the Host Controller generates this interrupt when it detects invalid descriptor data (Valid=0) at the ST_FDS state. ADMA Error State in the ADMA Error Status indicates that an error occurs in ST_FDS state. The Host Driver may find that Valid bit is not set at the error descriptor. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = No Interrupt. 1h (R) = ADMA error.
24	ACE	R/W	0h	Auto CMD error Auto CMD12 and Auto CMD23 use this error status. This bit is set when detecting that one of the bits D00-D04 in Auto CMD Error Status register (MMCHS_AC12) has changed from 0 to 1. In case of Auto CMD12, this bit is set to 1, not only when the errors in Auto CMD12 occur but also when Auto CMD12 is not executed due to the previous command error. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = No Error. 1h (R) = Auto CMD error.
23	RESERVED	R	0h	Reserved

Table 11-1925. MMCHS_STAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	DEB	R/W	0h	Data End Bit error This bit is set automatically when detecting a 0 at the end bit position of read data on DAT line or at the end position of the CRC status in write mode. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = No Error. 1h (R) = Data end bit error.
21	DCRC	R/W	0h	Data CRC Error This bit is set automatically when there is a CRC16 error in the data phase response following a block read command or if there is a 3-bit CRC status different of a position "010" token during a block write command. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = No Error. 1h (R) = Data CRC error.
20	DTO	R/W	0h	Data timeout error This bit is set automatically according to the following conditions: - busy timeout for R1b, R5b response type - busy timeout after write CRC status - write CRC status timeout - read data timeout. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = No error. 1h (R) = Time out.
19	CIE	R/W	0h	Command index error This bit is set automatically when response index differs from corresponding command index previously emitted. It depends on the enable in MMCHS_CMD[CICE] register. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = No error. 1h (R) = Command index error.
18	CEB	R/W	0h	Command end bit error This bit is set automatically when detecting a 0 at the end bit position of a command response. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = No error. 1h (R) = Command end bit error.
17	CCRC	R/W	0h	Command CRC Error This bit is set automatically when there is a CRC7 error in the command response depending on the enable in MMCHS_CMD[CCCE] register. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = No Error. 1h (R) = Command CRC error.

Table 11-1925. MMCHS_STAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	CTO	R/W	0h	Command Timeout Error This bit is set automatically when no response is received within 64 clock cycles from the end bit of the command. For commands that reply within 5 clock cycles - the timeout is still detected at 64 clock cycles. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = No error. 1h (R) = Time Out.
15	ERRI	R	0h	Error Interrupt If any of the bits in the Error Interrupt Status register (MMCHS_STAT[31-16]) are set, then this bit is set to 1. Therefore the host driver can efficiently test for an error by checking this bit first. Writes to this bit are ignored. 0h (R) = No Interrupt. 1h (R) = Error interrupt event(s) occurred.
14-11	RESERVED	R	0h	Reserved
10	BSR	R/W	0h	Boot status received interrupt This bit is set automatically when MMCHS_CON[BOOT] is set 1h or 2h and a boot status is received on DAT[0] line. This interrupt is only useful for MMC card. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = No Interrupt. 1h (R) = Boot status received interrupt.
9	OBI	R/W	0h	Out-Of-Band interrupt This bit is set automatically when MMCHS_CON[OBIE] is set and an Out-of-Band interrupt occurs on OBI pin. The interrupt detection depends on polarity controlled by MMCHS_CON[OBIP]. This interrupt is only useful for MMC card. The Out-of-Band interrupt signal is a system specific feature for future use, this signal is not required for existing specification implementation. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = No Out-Of-Band interrupt. 1h (R) = Interrupt Out-Of-Band occurs. **Note:** Out-Of-Band Interrupt feature is not supported.

Table 11-1925. MMCHS_STAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	CIRQ	R	0h	Card interrupt This bit is only used for SD and SDIO and CE-ATA cards. In 1-bit mode, interrupt source is asynchronous (can be a source of asynchronous wakeup). In 4-bit mode, interrupt source is sampled during the interrupt cycle. In CE-ATA mode, interrupt source is detected when the card drives CMD line to zero during one cycle after data transmission end. All modes above are fully exclusive. The controller interrupt must be clear by setting MMCHS_IE[CIRQ] to 0, then the host driver must start the interrupt service with card (clearing card interrupt status) to remove card interrupt source. Otherwise the Controller interrupt will be reasserted as soon as MMCHS_IE[CIRQ] is set to 1. Writes to this bit are ignored. 0h (R) = No card interrupt. 1h (R) = Generate card interrupt. **Note:** Wakeup feature is not supported.
7	CREM	R/W	0h	Card removal This bit is set automatically when MMCHS_PSTATE[CINS] changes from 1 to 0. A clear of this bit doesn't affect Card inserted present state (MMCHS_PSTATE[CINS]). 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = Card state stable or Debouncing. 1h (R) = Card removed. **Note:** Wakeup feature is not supported.
6	CINS	R/W	0h	Card insertion This bit is set automatically when MMCHS_PSTATE[CINS] changes from 0 to 1. A clear of this bit doesn't affect Card inserted present state (MMCHS_PSTATE[CINS]). 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = Card state stable or debouncing. 1h (R) = Card inserted. **Note:** Wakeup feature is not supported.
5	BRR	R/W	0h	Buffer read ready This bit is set automatically during a read operation to the card (see class 2 - block oriented read commands) when one block specified by MMCHS_BLK[BLEN] is completely written in the buffer. It indicates that the memory card has filled out the buffer and that the local host needs to empty the buffer by reading it. **Note:** If the DMA receive-mode is enabled, this bit is never set; instead a DMA receive request to the main DMA controller of the system is generated. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = Not Ready to read buffer. 1h (R) = Ready to read buffer.

Table 11-1925. MMCHS_STAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	BWR	R/W	0h	Buffer write ready This bit is set automatically during a write operation to the card (see class 4 - block oriented write command) when the host can write a complete block as specified by MMCHS_BLK[BLEN]. It indicates that the memory card has emptied one block from the buffer and that the local host is able to write one block of data into the buffer. **Note:** If the DMA transmit mode is enabled, this bit is never set; instead, a DMA transmit request to the main DMA controller of the system is generated. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = Not Ready to write buffer. 1h (R) = Ready to write buffer.
3	DMA	R/W	0h	DMA interrupt This status is set when an interrupt is required in the ADMA instruction and after the data transfer completion. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = Dma interrupt detected. 1h (R) = No dma interrupt.
2	BGE	R/W	0h	Block gap event When a stop at block gap is requested (MMCHS_HCTL[SBGR]), this bit is automatically set when transaction is stopped at the block gap during a read or write operation. This event does not occur when the stop at block gap is requested on the last block. In read mode, a 1-to-0 transition of the DAT Line active status (MMCHS_PSTATE[DLA]) between data blocks generates a Block gap event interrupt. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = No block gap event. 1h (R) = Transaction stopped at block gap.
1	TC	R/W	0h	Transfer completed This bit is always set when a read/write transfer is completed or between two blocks when the transfer is stopped due to a stop at block gap request (MMCHS_HCTL[SBGR]). In Read mode: This bit is automatically set on completion of a read transfer (MMCHS_PSTATE[RTA]). In write mode: This bit is set automatically on completion of the DAT line use (MMCHS_PSTATE[DLA]). 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = No transfer complete. 1h (R) = Data transfer complete.

Table 11-1925. MMCHS_STAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	CC	R/W	0h	Command complete This bit is set when a 1-to-0 transition occurs in the register command inhibit (MMCHS_PSTATE[CMDI]) If the command is a type for which no response is expected, then the command complete interrupt is generated at the end of the command. A command timeout error (MMCHS_STAT[CTO]) has higher priority than command complete (MMCHS_STAT[CC]). If a response is expected but none is received, then a command timeout error is detected and signaled instead of the command complete interrupt. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared. 0h (R) = No Command complete. 1h (R) = Command complete.

Table 11-1926. Register Call Summary for MMCHS_STAT

MMC/SD Registers

- MMCHS_SYSTEST Register (Offset = 128h) [reset = 0h]: [0][1][2][3]
- MMCHS_STAT Register (Offset = 230h) [reset = 0h]: [0][1][2][3][4][5]
- MMCHS_HCTL Register (Offset = 228h) [reset = 0h]: [0]
- MMCHS_AC12 Register (Offset = 23Ch) [reset = 0h]: [0]
- MMCHS_PSTATE Register (Offset = 224h) [reset = 0---0000h]: [0][1][2][3][4][5][6]
- MMCHS_CSRE Register (Offset = 124h) [reset = 0h]: [0]
- MMCHS_BLK Register (Offset = 204h) [reset = 0h]: [0]
- MMCHS_CMD Register (Offset = 20Ch) [reset = 0h]: [0][1]
- MMCHS_DATA Register (Offset = 220h) [reset = 0h]: [0][1]
- MMCHS_CAPA Register (Offset = 240h) [reset = 20E10080h]: [0][1]
- MMC/SD Registers: [0]
- MMCHS_SYSCTL Register (Offset = 22Ch) [reset = 0h]: [0][1][2]

MMC/SD Functional Description

- Interrupt-Driven Operation: [0][1][2][3][4][5]
- Interrupt Requests: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25][26][27][28][29][30][31][32]
- Read Data Time-Out: [0][1]
- Polling: [0][1]
- Requirements for Descriptors: [0]
- MMC CE-ATA Command Completion Disable Management: [0]
- Busy Time-Out After Write CRC Status: [0]
- Busy Time-Out for R1b, R5b Response Type: [0]
- Transfer or Command Status and Errors Reporting: [0][1]
- Data Buffer: [0][1][2]
- Test Registers: [0][1]
- Data Buffer Status: [0][1][2]
- Boot Acknowledge Time-Out: [0][1][2][3][4][5]
- Write CRC Status Time-Out: [0]

11.12.6.24 MMCHS_IE Register (Offset = 234h) [reset = 0h]

MMCHS_IE is shown in Figure 11-887 and described in Table 11-1928.

Interrupt Status Enable Register
This register allows to enable/disable the module to set status bits, on an event-by-event basis.
MMCHS_IE[31-16] = Error Interrupt Status Enable
MMCHS_IE[15-0] = Normal Interrupt Status Enable.

Table 11-1927. MMCHS_IE Instances

Instance	Physical Address
MMCSD_0	2300 0234h
MMCSD_1	2310 0234h

Figure 11-887. MMCHS_IE Register

31	30	29	28	27	26	25	24
RESERVED		BADA_ENABLE	CERR_ENABLE	RESERVED	TE_ENABLE	ADMAE_ENABLE	ACE_ENABLE
R-0h		R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	DEB_ENABLE	DCRC_ENABLE	DTO_ENABLE	CIE_ENABLE	CEB_ENABLE	CCRC_ENABLE	CTO_ENABLE
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
NULL	RESERVED				BSR_ENABLE	OBI_ENABLE	CIRQ_ENABLE
R-0h	R-0h				R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CREM_ENABLE	CINS_ENABLE	BRR_ENABLE	BWR_ENABLE	DMA_ENABLE	BGE_ENABLE	TC_ENABLE	CC_ENABLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1928. MMCHS_IE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29	BADA_ENABLE	R/W	0h	Bad access to data space Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
28	CERR_ENABLE	R/W	0h	Card Error Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
27	RESERVED	R	0h	Reserved
26	TE_ENABLE	R/W	0h	Tuning Error Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
25	ADMAE_ENABLE	R/W	0h	ADMA Error Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
24	ACE_ENABLE	R/W	0h	Auto CMD Error Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.

Table 11-1928. MMCHS_IE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	RESERVED	R	0h	Reserved
22	DEB_ENABLE	R/W	0h	Data End Bit Error Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
21	DCRC_ENABLE	R/W	0h	Data CRC Error Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
20	DTO_ENABLE	R/W	0h	Data Timeout Error Status Enable 0h (R/W) = The data timeout detection is deactivated. The host controller provides the clock to the card until the card sends the data or the transfer is aborted. 1h (R/W) = The data timeout detection is enabled.
19	CIE_ENABLE	R/W	0h	Command Index Error Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
18	CEB_ENABLE	R/W	0h	Command End Bit Error Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
17	CCRC_ENABLE	R/W	0h	Command CRC Error Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
16	CTO_ENABLE	R/W	0h	Command Timeout Error Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
15	NULL	R	0h	Fixed to 0 The host driver shall control error interrupts using the Error Interrupt Signal Enable register. Writes to this bit are ignored.
14-11	RESERVED	R	0h	Reserved
10	BSR_ENABLE	R/W	0h	Boot Status Enable A write to this bit field when MMCHS_CON[BOOT_ACK] is set to 0h is ignored. 0h (R/W) = Masked. 1h (R/W) = Enabled.
9	OBI_ENABLE	R/W	0h	Out-of-Band Status Enable A write to this bit field when MMCHS_CON[OBIE] is set to '0' is ignored. 0h (R/W) = Masked. 1h (R/W) = Enabled. **Note:** Out-Of-Band Interrupt feature is not supported.
8	CIRQ_ENABLE	R/W	0h	Card Status Enable A clear of this bit also clears the corresponding status bit. During 1-bit mode, if the interrupt routine doesn't remove the source of a card interrupt in the SDIO card, the status bit is reasserted when this bit is set to 1. 0h (R/W) = Masked. 1h (R/W) = Enabled. **Note:** Wakeup feature is not supported.

Table 11-1928. MMCHS_IE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	CREM_ENABLE	R/W	0h	Card Removal Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled. **Note:** Wakeup feature is not supported.
6	CINS_ENABLE	R/W	0h	Card Insertion Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled. **Note:** Wakeup feature is not supported.
5	BRR_ENABLE	R/W	0h	Buffer Read Ready Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
4	BWR_ENABLE	R/W	0h	Buffer Write Ready Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
3	DMA_ENABLE	R/W	0h	DMA Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
2	BGE_ENABLE	R/W	0h	Block Gap Event Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
1	TC_ENABLE	R/W	0h	Transfer Complete Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
0	CC_ENABLE	R/W	0h	Command Complete Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.

Table 11-1929. Register Call Summary for MMCHS_IE

MMC/SD Registers • MMCHS_STAT Register (Offset = 230h) [reset = 0h]: [0][1] • MMCHS_IE Register (Offset = 234h) [reset = 0h]: [0][1][2] • MMC/SD Registers: [0]
MMC/SD Functional Description • Interrupt-Driven Operation: [0][1] • Interrupt Requests: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25]
MMC/SD Programming Guide

11.12.6.25 MMCHS_ISE Register (Offset = 238h) [reset = 0h]

MMCHS_ISE is shown in Figure 11-888 and described in Table 11-1931.

Interrupt Signal Enable Register
This register allows to enable/disable the module internal sources of status, on an event-by-event basis.
MMCHS_ISE[31-16] = Error Interrupt Signal Enable
MMCHS_ISE[15-0] = Normal Interrupt Signal Enable.

Table 11-1930. MMCHS_ISE Instances

Instance	Physical Address
MMCSD_0	2300 0238h
MMCSD_1	2310 0238h

Figure 11-888. MMCHS_ISE Register

31	30	29	28	27	26	25	24
RESERVED		BADA_SIGEN	CERR_SIGEN	RESERVED	TE_SIGEN	ADMAE_SIGEN	ACE_SIGEN
R-0h		R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	DEB_SIGEN	DCRC_SIGEN	DTO_SIGEN	CIE_SIGEN	CEB_SIGEN	CCRC_SIGEN	CTO_SIGEN
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
NULL	RESERVED				BSR_SIGEN	OBI_SIGEN	CIRQ_SIGEN
R-0h	R-0h				R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CREM_SIGEN	CINS_SIGEN	BRR_SIGEN	BWR_SIGEN	DMA_SIGEN	BGE_SIGEN	TC_SIGEN	CC_SIGEN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1931. MMCHS_ISE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29	BADA_SIGEN	R/W	0h	Bad access to data space Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
28	CERR_SIGEN	R/W	0h	Card Error Interrupt Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
27	RESERVED	R	0h	Reserved
26	TE_SIGEN	R/W	0h	Tuning Error Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
25	ADMAE_SIGEN	R/W	0h	ADMA Error Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
24	ACE_SIGEN	R/W	0h	Auto CMD Error Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
23	RESERVED	R	0h	Reserved

Table 11-1931. MMCHS_ISE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	DEB_SIGEN	R/W	0h	Data End Bit Error Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
21	DCRC_SIGEN	R/W	0h	Data CRC Error Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
20	DTO_SIGEN	R/W	0h	Data Timeout Error Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
19	CIE_SIGEN	R/W	0h	Command Index Error Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
18	CEB_SIGEN	R/W	0h	Command End Bit Error Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
17	CCRC_SIGEN	R/W	0h	Command CRC Error Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
16	CTO_SIGEN	R/W	0h	Command timeout Error Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
15	NULL	R	0h	Fixed to 0 The host driver shall control error interrupts using the Error Interrupt Signal Enable register. Writes to this bit are ignored.
14-11	RESERVED	R	0h	Reserved
10	BSR_SIGEN	R/W	0h	Boot Status Signal Enable A write to this bit field when MMCHS_CON[BOOT_ACK] is set to 0h is ignored. 0h (R/W) = Masked. 1h (R/W) = Enabled.
9	OBI_SIGEN	R/W	0h	Out-Of-Band Interrupt Signal Enable A write to this bit field when MMCHS_CON[OBIE] is set to '0' is ignored. 0h (R/W) = Masked. 1h (R/W) = Enabled. **Note:** Out-Of-Band Interrupt feature is not supported.
8	CIRQ_SIGEN	R/W	0h	Card Interrupt Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
7	CREM_SIGEN	R/W	0h	Card Removal Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
6	CINS_SIGEN	R/W	0h	Card Insertion Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.

Table 11-1931. MMCHS_ISE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	BRR_SIGEN	R/W	0h	Buffer Read Ready Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
4	BWR_SIGEN	R/W	0h	Buffer Write Ready Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
3	DMA_SIGEN	R/W	0h	DMA Interrupt Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
2	BGE_SIGEN	R/W	0h	Black Gap Event Signal Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
1	TC_SIGEN	R/W	0h	Transfer Completed Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.
0	CC_SIGEN	R/W	0h	Command Complete Status Enable 0h (R/W) = Masked. 1h (R/W) = Enabled.

Table 11-1932. Register Call Summary for MMCHS_ISE

MMC/SD Registers • MMCHS_SYSTEST Register (Offset = 128h) [reset = 0h]: [0] • MMCHS_ISE Register (Offset = 238h) [reset = 0h]: [0][1][2] • MMC/SD Registers: [0]
MMC/SD Functional Description • Polling: [0] • Interrupt Requests: [0][1][2][3][4]

11.12.6.26 MMCHS_AC12 Register (Offset = 23Ch) [reset = 0h]

MMCHS_AC12 is shown in Figure 11-889 and described in Table 11-1934.

Host Control 2 Register and Auto CMD Error Status Register
This register is used to indicate CMD12 response error of Auto CMD12 and CMD23 response error of Auto CMD23. The Host driver can determine what kind of Auto CMD12 / CMD23 errors occur by this register. Auto CMD23 errors are indicated only in bits[4-1]. Bits[7-0] are valid only when the MMCHS_CMD[3-2] ACEN bitfield is configured to enable Auto CMD and the Auto CMD Error bit (MMCHS_STAT[24]ACE) is set.

Table 11-1933. MMCHS_AC12 Instances

Instance	Physical Address
MMCSD_0	2300 023Ch
MMCSD_1	2310 023Ch

Figure 11-889. MMCHS_AC12 Register

31	30	29	28	27	26	25	24
PV_ENABLE	AI_ENABLE	RESERVED					
R/W-0h	R/W-0h	R-0h					
23	22	21	20	19	18	17	16
SCLK_SEL	ET	DS_SEL		V1V8_SIGEN	UHSMS		
R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h		
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CNI	RESERVED		ACIE	ACEB	ACCE	ACTO	ACNE
R-0h	R-0h		R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1934. MMCHS_AC12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	PV_ENABLE	R/W	0h	Preset Value Enable Host Controller Version 3.00 supports this bit. As the operating SDCLK frequency and I/O driver strength depend on the Host System implementation, it is difficult to determine these parameters in the Standard Host Driver. When Preset Value Enable is set, automatic SDCLK frequency generation and driver strength selection is performed without considering system specific conditions. This bit enables the functions defined in the Preset Value registers, see, Table 11-1839. If this bit is set to 0, MMCHS_SYSCTL[15-6] CLKD, MMCHS_SYSCTL[5] CGS and MMCHS_AC12[21-20] DS_SEL are set by Host Driver. If this bit is set to 1, MMCHS_SYSCTL[15-6] CLKD, MMCHS_SYSCTL[5] CGS and MMCHS_AC12[21-20] DS_SEL are set by Host Controller as specified in the Preset Value registers, see, Table 11-1839. 0h (R/W) = SDCLK and Driver Strength (DS_SEL) are controlled by Host Driver. 1h (R/W) = Automatic Selection by Preset Value are Enabled.

Table 11-1934. MMCHS_AC12 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
30	AI_ENABLE	R/W	0h	Asynchronous Interrupt Enable This bit can be set to 1 if a card supports asynchronous interrupts and MMCHS_CAPA[29] AIS is set to 1. Asynchronous interrupt is effective when DAT[1] interrupt is used in 4-bit SD mode (and zero is set to Interrupt Pin Select in the Shared Bus Control register). If this bit is set to 1, the Host Driver can stop the SDCLK during asynchronous interrupt period to save power. During this period, the Host Controller continues to deliver the Card Interrupt to the host when it is asserted by the Card. 0h (R/W) = Disabled. 1h (R/W) = Enabled.
29-24	RESERVED	R	0h	Reserved
23	SCLK_SEL	R/W	0h	Sampling Clock Select Host Controller uses this bit to select sampling clock to receive CMD and DAT. This bit is set by tuning procedure and valid after the completion of tuning (when MMCHS_AC12[22] ET is cleared). Setting 1 means that tuning is completed successfully and setting 0 means that tuning is failed. Writing 1 to this bit is meaningless and ignored. A tuning circuit is reset by writing to 0. This bit can be cleared with setting MMCHS_AC12[22] ET. Once the tuning circuit is reset, it will take time to complete tuning sequence. Therefore, Host Driver should keep this bit to 1 to perform re-tuning sequence to compete re-tuning sequence in a short time. Change of this bit is not allowed while the Host Controller is receiving response or a read data block. 0h (R/W) = Fixed clock is used to sample data. 1h (R/W) = Tuned clock is used to sample data.
22	ET	R/W	0h	Execute Tuning This bit is set to 1 to start tuning procedure and automatically cleared when tuning procedure is completed. The result of tuning is indicated to MMCHS_AC12[23] SCLK_SEL. Tuning procedure is aborted by writing 0. This is Read-Write with automatic clear register. 0h (R/W) = Not Tuned or Tuning Completed. 1h (R/W) = Execute Tuning.
21-20	DS_SEL	R/W	0h	Driver Strength Select Host Controller output driver in 1.8 V signaling is selected by this bit. In 3.3 V signaling, this field is not effective. This field can be set depending on Driver Type A, C and D support bits (DTA, DTC and DTD respectively) in the MMCHS_CAPA2 register. This bit depends on setting of Preset Value Enable MMCHS_AC12[31] PV_ENABLE bit. If Preset Value Enable = 0, this field is set by Host Driver. If Preset Value Enable = 1, this field is automatically set by a value specified in the one of Preset Value registers, see, Table 11-1839. 0h (R/W) = Driver Type B is selected (Default). 1h (R/W) = Driver Type A is selected. 2h (R/W) = Driver Type C is selected. 3h (R/W) = Driver Type D is selected. **Note:** This feature is not supported.

Table 11-1934. MMCHS_AC12 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	V1V8_SIGEN	R/W	0h	1.8 V Signaling Enable This bit controls voltage regulator for I/O cell. 3.3 V is supplied to the card regardless of signaling voltage. Setting this bit from 0 to 1 starts changing signal voltage from 3.3 V to 1.8 V. 1.8 V regulator output shall be stable within 5ms. Host Controller clears this bit if switching to 1.8 V signaling fails. Clearing this bit from 1 to 0 starts changing signal voltage from 1.8 V to 3.3 V. 3.3 V regulator output shall be stable within 5ms. Host Driver can set this bit to 1 when Host Controller supports 1.8 V signaling (One of support bits is set to 1: SDR50, SDR104 or DDR50 in MMCHS_CAPA2 register) and the card or device supports UHS-I (S18A=1. Refer to Bus Signal Voltage Switch Sequence in the Physical Layer Specification Version 3.0x). 0h (R/W) = 3.3 V Signaling. 1h (R/W) = 1.8 V Signaling. **Note:** This feature is not supported (MMCSD_0 supports 3.3 V only. MMCSD_1 supports 1.8 V only).
18-16	UHSMS	R/W	0h	UHS Mode Select This field is used to select one of UHS-I modes and effective when 1.8 V Signaling Enable is set to 1. If MMCHS_AC12[31] PV_ENABLE is set to 1, Host Controller sets MMCHS_SYSCTL[15-6] CLKD, MMCHS_SYSCTL[5] CGS and MMCHS_AC12[21-20] DS_SEL according to Preset Value registers, see, Table 11-1839. In this case, one of preset value registers is selected by this field. Host Driver needs to reset MMCHS_SYSCTL[2] CEN before changing this field to avoid generating clock glitch. After setting this field, Host Driver sets MMCHS_SYSCTL[2] CEN again. When SDR50, SDR104 or DDR50 is selected for SDIO card, interrupt detection at the block gap shall not be used. Read Wait timing is changed for these modes. Refer to the SDIO Specification Version 3.00 for more detail. 0h (R/W) = SDR12. 1h (R/W) = SDR25. 2h (R/W) = SDR50. 3h (R/W) = SDR104. 4h (R/W) = DDR50. 5h (R/W) = Reserved. 6h (R/W) = Reserved. 7h (R/W) = Reserved. **Note:** This feature is not supported.
15-8	RESERVED	R	0h	Reserved
7	CNI	R	0h	Command Not Issued By Auto CMD12 Error Setting this bit to 1 means CMD_wo_DAT is not executed due to an Auto CMD12 Error (D04-D01) in this bit field. This bit is set to 0 when Auto CMD Error is generated by Auto CMD23. 0h (R) = No error. 1h (R) = Command not issued.
6-5	RESERVED	R	0h	Reserved
4	ACIE	R	0h	Auto CMD Index Error - For Auto CMD12 and Auto CMD23 This bit is set if the Command Index error occurs in response to a command. 0h (R) = No error. 1h (R) = Error.

Table 11-1934. MMCHS_AC12 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	ACEB	R	0h	Auto CMD End Bit Error - For Auto CMD12 and Auto CMD23 This bit is set when detecting that the end bit of command response is 0. 0h (R) = No error. 1h (R) = End bit Error Generated.
2	ACCE	R	0h	Auto CMD CRC Error - For Auto CMD12 and Auto CMD23 This bit is set when detecting a CRC error in the command response. 0h (R) = No error. 1h (R) = CRC Error Generated.
1	ACTO	R	0h	Auto CMD Timeout Error - For Auto CMD12 and Auto CMD23 This bit is set if no response is returned within 64 SDCLK cycles from the end bit of command. If this bit is set to1, the other error status bits (D04-D02) are meaningless. 0h (R) = No error. 1h (R) = Auto CMD Time Out.
0	ACNE	R	0h	Auto CMD12 Not Executed If memory multiple block data transfer is not started due to command error, this bit is not set because it is not necessary to issue Auto CMD12. Setting this bit to 1 means the Host Controller cannot issue Auto CMD12 to stop memory multiple block data transfer due to some error. If this bit is set to 1, other error status bits (D04-D01) are meaningless. This bit is set to 0 when Auto CMD Error is generated by Auto CMD23. 0h (R) = Auto CMD12 Executed. 1h (R) = Auto CMD12 Not Executed.

Table 11-1935. Register Call Summary for MMCHS_AC12

MMC/SD Registers • MMCHS_STAT Register (Offset = 230h) [reset = 0h]: [0] • MMCHS_PSTATE Register (Offset = 224h) [reset = 0---0000h]: [0][1][2] • MMCHS_FE Register (Offset = 250h) [reset = 0h]: [0] • MMCHS_CMD Register (Offset = 20Ch) [reset = 0h]: [0][1] • MMCHS_AC12 Register (Offset = 23Ch) [reset = 0h]: [0][1][2][3][4][5][6][7][8] • MMC/SD Registers: [0] • MMCHS_SYSCTL Register (Offset = 22Ch) [reset = 0h]: [0]
MMC/SD Functional Description • Asynchronous Interrupt: [0][1] • Different Types of Responses: [0] • Interrupt Requests: [0]

11.12.6.27 MMCHS_CAPA Register (Offset = 240h) [reset = 20E10080h]

MMCHS_CAPA is shown in Figure 11-890 and described in Table 11-1937.

Capabilities Register
This register lists the capabilities of the MMC host controller.

Table 11-1936. MMCHS_CAPA Instances

Instance	Physical Address
MMCSD_0	2300 0240h
MMCSD_1	2310 0240h

Figure 11-890. MMCHS_CAPA Register

31	30	29	28	27	26	25	24
RESERVED		AIS	BIT64	RESERVED	VS18	VS30	VS33
R-0h		R-1h	R-0h	R-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
SRS	DS	HSS	RESERVED	AD2S	RESERVED	MBL	
R-1h	R-1h	R-1h	R-0h	R-0h	R-0h	R-1h	
15	14	13	12	11	10	9	8
BCF							
R-0h							
7	6	5	4	3	2	1	0
TCU	RESERVED	TCF					
R-1h	R-0h	R-0h					

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1937. MMCHS_CAPA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29	AIS	R	1h	Asynchronous Interrupt Support Refer to SDIO Specification Version 3.00 about asynchronous interrupt. 0h (R) = Asynchronous Interrupt Not Supported. 1h (R) = Asynchronous Interrupt Supported.
28	BIT64	R	0h	64 Bit System Bus Support Setting 1 to this bit indicates that the Host Controller supports 64-bit address descriptor mode and is connected to 64-bit address system bus. 0h (R) = 32 bit System bus address. 1h (R) = 64 bit System bus address.
27	RESERVED	R	0h	Reserved
26	VS18	R/W	0h	Voltage support 1.8 V Initialization of this bit field (via a write access to this bit field) depends on the system capabilities. The host driver shall not modify this bit field after the initilaization. This bit field is only reinitialized by a hard reset (via MMC_RST signal). 0h (W) = 1.8 V Not supported. 1h (W) = 1.8 V Supported. 0h (R) = 1.8 V Not Supported. 1h (R) = 1.8 V Supported.

Table 11-1937. MMCHS_CAPA Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	VS30	R/W	0h	Voltage support 3.0 V Initialization of this bit field (via a write access to this bit field) depends on the system capabilities. The host driver shall not modify this bit field after the initilaization. This bit field is only reinitialized by a hard reset (via MMC_RST signal). 0h (W) = 3.0 V Not supported. 1h (W) = 3.0 V Supported. 0h (R) = 3.0 V Not Supported. 1h (R) = 3.0 V Supported.
24	VS33	R/W	0h	Voltage support 3.3 V Initialization of this bit field (via a write access to this bit field) depends on the system capabilities. The host driver shall not modify this bit field after the initilaization. This bit field is only reinitialized by a hard reset (via MMC_RST signal). 0h (W) = 3.3 V Not supported. 1h (W) = 3.3 V Supported. 0h (R) = 3.3 V Not Supported. 1h (R) = 3.3 V Supported.
23	SRS	R	1h	Suspend/Resume support (SDIO cards only) This bit indicates whether the host controller supports Suspend/Resume functionality. 0h (R) = The Host controller does not Suspend/Resume functionality. 1h (R) = The Host controller supports Suspend/Resume functionality.
22	DS	R	1h	DMA support This bit indicates that the Host Controller is able to use DMA to transfer data between system memory and the Host Controller directly. 0h (R) = DMA Not Supported. 1h (R) = DMA Supported.
21	HSS	R	1h	High speed support This bit indicates that the host controller supports high speed operations and can supply an up-to maximum card frequency. 0h (R) = High Speed Not Supported. 1h (R) = High Speed Supported. **Note:** High Speed modes are supported, but MMCHS_HCTL[2] HSPE bit must always be set to 0x0 because device was timing closed with MMCHS_HCTL[2] HSPE bit set to 0x0 for all supported modes of operation.
20	RESERVED	R	0h	Reserved
19	AD2S	R	0h	ADMA2 Support This bit indicates whether the Host Controller is capable of using ADMA2. It depends on setting of generic parameter MADMA_EN. 0h (R) = ADMA2 not Supported. 1h (R) = ADMA2 Supported.
18	RESERVED	R	0h	Reserved

Table 11-1937. MMCHS_CAPA Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17-16	MBL	R	1h	Maximum block length This value indicates the maximum block size that the host driver can read and write to the buffer in the host controller. This value depends on definition of generic parameter with a max value of 2048 bytes. The host controller supports 512 bytes and 1024 bytes block transfers. 0h (R) = 512 bytes. 1h (R) = 1024 bytes. 2h (R) = 2048 bytes.
15-8	BCF	R	0h	Base Clock Frequency For SD Clock This value indicates the base (maximum) clock frequency for the SD Clock. 8-bit Base Clock Frequency This mode is supported by the Host Controller Version 3.00. Unit values are 1 MHz. The supported clock range is 10 MHz to 255 MHz. FFh : 255 MHz ..... : ..... 02h : 2 MHz 01h : 1 MHz 00h : Get information via another method If the real frequency is 16.5 MHz, the lager value shall be set 0001 0001b (17 MHz) because the Host Driver use this value to calculate the clock divider value (Refer to MMCHS_SYSCTL[15-6] CLKD) and it shall not exceed upper limit of the SD Clock frequency. If these bits are all 0, the Host System has to get information via another method. 0h (R) = The value indicating the base (maximum) frequency for the output clock provided to the card is system dependent and is not available in this bit filed. Get the information via another method.
7	TCU	R	1h	Timeout clock unit This bit shows the unit of base clock frequency used to detect Data Timeout Error (MMCHS_STAT[DTO]). 0h (R) = KHz. 1h (R) = MHz.
6	RESERVED	R	0h	Reserved
5-0	TCF	R	0h	Timeout clock frequency The timeout clock frequency is used to detect Data Timeout Error (MMCHS_STAT[DTO]). 0h (R) = The timeout clock frequency depends on the frequency of the clock provided to the card. The value of the timeout clock frequency is not available in this bit field.

Table 11-1938. Register Call Summary for MMCHS_CAPA

MMC/SD Registers • MMCHS_HCTL Register (Offset = 228h) [reset = 0h]: [0][1][2][3] • MMCHS_AC12 Register (Offset = 23Ch) [reset = 0h]: [0] • MMCHS_CUR_CAPA Register (Offset = 248h) [reset = 0h]: [0] • MMCHS_CAPA Register (Offset = 240h) [reset = 20E10080h]: [0] • MMC/SD Registers: [0] • MMCHS_SYSCTL Register (Offset = 22Ch) [reset = 0h]: [0][1]
MMC/SD Functional Description • Data Buffer: [0] • Asynchronous Interrupt: [0] • Software Reset: [0]
MMC/SD Programming Guide • MMC Host Controller Initialization Flow: [0]

11.12.6.28 MMCHS_CAPA2 Register (Offset = 244h) [reset = F77h]

MMCHS_CAPA2 is shown in Figure 11-891 and described in Table 11-1940.

Capabilities 2 Register
This register provides the Host Driver with information specific to the Host Controller implementation. The Host Controller may implement these values as fixed or loaded from flash memory during power on initialization. Refer to Software Reset For All in the Software Reset register for loading from flash memory and completion timing control.

Table 11-1939. MMCHS_CAPA2 Instances

Instance	Physical Address
MMCSD_0	2300 0244h
MMCSD_1	2310 0244h

Figure 11-891. MMCHS_CAPA2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
CM							
R-0h							
15	14	13	12	11	10	9	8
RTM		TSDR50	RESERVED	TCRT			
R-0h		R-0h	R-0h	R-Fh			
7	6	5	4	3	2	1	0
RESERVED	DTD	DTC	DTA	RESERVED	DDR50	SDR104	SDR50
R-0h	R-1h	R-1h	R-1h	R-0h	R-1h	R-1h	R-1h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1940. MMCHS_CAPA2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	CM	R	0h	Clock Multiplier This field indicates clock multiplier value of programmable clock generator. Refer to MMCHS_SYSCTL [15-0]. Setting 00h means that Host Controller does not support programmable clock generator. 00h : Clock Multiplier is Not Supported 01h : Clock Multiplier M = 2 02h : Clock Multiplier M = 3 ..... : ..... FFh : Clock Multiplier M = 256.

Table 11-1940. MMCHS_CAPA2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-14	RTM	R	0h	Re-Tuning Modes This field selects re-tuning method and limits the maximum data length. Bit47-46 Re-Tuning Mode Re-Tuning Method Data Length. There are two re-tuning timings: Re-Tuning Request controlled by the Host Controller and expiration of a Re-Tuning Timer controlled by the Host Driver. By receiving either timing, the Host Driver executes the re-tuning procedure just before a next command issue. The maximum data length per read/write command is restricted so that re-tuning procedures can be inserted during data transfers. (1) Re-Tuning Mode 1 The host controller does not have any internal logic to detect when the re-tuning needs to be performed. In this case, the Host Driver should maintain all re-tuning timings by using a Re-Tuning Timer. To enable inserting the re-tuning procedure during data transfers, the data length per read/write command shall be limited up to 4 MB. (2) Re-Tuning Mode 2 The host controller has the capability to indicate the re-tuning timing by Re-Tuning Request during data transfers. Then the data length per read/write command shall be limited up to 4 MB. During non data transfer, re-tuning timing is determined by either Re-Tuning Request or Re-Tuning Timer. If Re-Tuning Request is used, Re-Tuning Timer should be disabled. (3) Re-Tuning Mode 3 The host controller has the capability to take care of the re-tuning during data transfer (Auto Re-Tuning). Re-Tuning Request shall not be generated during data transfers and there is no limitation to data length per read/write command. During non data transfer, re-tuning timing is determined by either Re-Tuning Request or Re-Tuning Timer. If Re-Tuning Request is used, Re-Tuning Timer should be disabled. Re-Tuning Timer Control Example for Re-Tuning Mode 1 The initial value of re-tuning timer is provided by Timer Count for Re-Tuning field in this bit field. The timer starts counting by loading the initial value. When the timer expires, the Host Driver marks an expiration flag. On receiving a command request, the Host driver checks the expiration flag. If the expiration flag is set, then the Host Driver should perform the re-tuning procedure before issuing a command. If the expiration flag is not set, then the Host Driver issues a command without performing the re-tuning procedure. Every time the re-tuning procedure is performed, the timer loads the new initial value and the expiration flag is cleared. Re-Tuning Timer Control Example for Re-Tuning Mode 2 and Mode 3 The timer control is almost the same as Re-Tuning Mode 1 except the timer loads the new initial value after data transfer (when receiving Transfer Complete). In case of Mode 3, Timer Count for Re-Tuning is set either smaller value: Tuning effective time after re-tuning procedure or after data transfer. If a Host System goes into power down mode, the Host Driver should stop the re-tuning timer and set the expiration flag to 1 when the Host System resumes from power down mode. 0h (R) = Timer - Max data length 4 MB. 1h (R) = Timer and Re-Tuning Request - Max data length 4 MB. 2h (R) = Auto Re-Tuning (for transfer) - Timer and Re-Tuning Request. 3h (R) = Reserved.

Table 11-1940. MMCHS_CAPA2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	TSDR50	R	0h	Use Tuning for SDR50 If this bit is set to 1, this Host Controller requires tuning to operate SDR50. (Tuning is always required to operate SDR104). 0h (R) = SDR50 does not require tuning. 1h (R) = SDR50 requires tuning. **Note:** SDR50 mode is not supported.
12	RESERVED	R	0h	Reserved
11-8	TCRT	R	Fh	Timer Count for Re-Tuning This field indicates an initial value of the Re-Tuning Timer for Re-Tuning Mode 1 to 3. Setting to 0 disables Re-Tuning Timer. 0h (R) = Re-Tuning Timer disabled. 1h (R) = 1 second. 2h (R) = 2 seconds. 3h (R) = 4 seconds. 4h (R) = 8 seconds. 5h (R) = 16 seconds. 6h (R) = 32 seconds. 7h (R) = 64 seconds. 8h (R) = 128 seconds. 9h (R) = 256 seconds. Ah (R) = 512 seconds. Bh (R) = 1024 seconds. Ch (R) = Reserved. Dh (R) = Reserved. Eh (R) = Reserved. Fh (R) = Get information from other source.
7	RESERVED	R	0h	Reserved
6	DTD	R	1h	Driver Type D Support This bit indicates support of Driver Type D for 1.8 Signaling. 0h (R) = Driver Type D is Not Supported. 1h (R) = Driver Type D is Supported.
5	DTC	R	1h	Driver Type C Support This bit indicates support of Driver Type C for 1.8 Signaling. 0h (R) = Driver Type C is Not Supported. 1h (R) = Driver Type C is Supported.
4	DTA	R	1h	Driver Type A Support This bit indicates support of Driver Type A for 1.8 Signaling. 0h (R) = Driver Type A is Not Supported. 1h (R) = Driver Type A is Supported.
3	RESERVED	R	0h	Reserved
2	DDR50	R	1h	DDR50 Support 0h (R) = DDR50 is Not Supported. 1h (R) = DDR50 is Supported.
1	SDR104	R	1h	SDR104 Support SDR104 requires tuning. 0h (R) = SDR104 is Not Supported. 1h (R) = SDR104 is Supported. **Note:** SDR104 mode is not supported.

Table 11-1940. MMCHS_CAPA2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	SDR50	R	1h	SDR50 Support If SDR104 is supported, this bit shall be set to 1. Bit 13 indicates whether SDR50 requires tuning or not. 0h (R) = SDR50 is Not Supported. 1h (R) = SDR50 is Supported. **Note:** SDR50 mode is not supported.

Table 11-1941. Register Call Summary for MMCHS_CAPA2

MMC/SD Registers

- MMCHS_CAPA2 Register (Offset = 244h) [reset = F77h]: [0]
- MMCHS_PSTATE Register (Offset = 224h) [reset = 0---0000h]: [0]
- MMCHS_AC12 Register (Offset = 23Ch) [reset = 0h]: [0][1]
- MMC/SD Registers: [0]
- MMCHS_SYSCTL Register (Offset = 22Ch) [reset = 0h]: [0]

11.12.6.29 MMCHS_CUR_CAPA Register (Offset = 248h) [reset = 0h]

MMCHS_CUR_CAPA is shown in Figure 11-892 and described in Table 11-1943.

Maximum Current Capabilities Register
This register indicates the maximum current capability for each voltage. The value is meaningful if the voltage support is set in the capabilities register (MMCHS_CAPA).
Initialization of this register (via a write access to this register) depends on the system capabilities. The host driver shall not modify this register after the initilaization.
This register is only reinitialized by a hard reset (via MMC_RST signal).

Table 11-1942. MMCHS_CUR_CAPA Instances

Instance	Physical Address
MMCSD_0	2300 0248h
MMCSD_1	2310 0248h

Figure 11-892. MMCHS_CUR_CAPA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								CUR_1V8								CUR_3V0								CUR_3V3							
R-0h								R/W-0h								R/W-0h								R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1943. MMCHS_CUR_CAPA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	CUR_1V8	R/W	0h	Maximum current for 1.8 V 0h (R) = The maximum current capability for this voltage is not available. Feature not implemented.
15-8	CUR_3V0	R/W	0h	Maximum current for 3.0 V 0h (R) = The maximum current capability for this voltage is not available. Feature not implemented.
7-0	CUR_3V3	R/W	0h	Maximum current for 3.3 V 0h (R) = The maximum current capability for this voltage is not available. Feature not implemented.

Table 11-1944. Register Call Summary for MMCHS_CUR_CAPA

MMC/SD Registers • MMCHS_CUR_CAPA Register (Offset = 248h) [reset = 0h]: [0] • MMC/SD Registers: [0] • MMCHS_SYSCTL Register (Offset = 22Ch) [reset = 0h]: [0]
MMC/SD Functional Description • Software Reset: [0]
MMC/SD Programming Guide • MMC Host Controller Initialization Flow: [0]

11.12.6.30 MMCHS_FE Register (Offset = 250h) [reset = 0h]

MMCHS_FE is shown in Figure 11-893 and described in Table 11-1946.

Force Event Register for Auto CMD Error Status and Error Interrupt status
The Force Event Register is not a physically implemented register. Rather, it is an address at which the Auto CMD Error Status Register (MMCHS_AC12) can be written.
Writing 1: set each bit of the Auto CMD Error Status Register
Writing 0: no effect
Rather, it is an address at which the Error Interrupt Status register can be written. The effect of a write to this address will be reflected in the Error Interrupt Status Register if the corresponding bit of the Error Interrupt Status Enable Register is set.
Writing 1: set each bit of the Error Interrupt Status Register
Writing 0: no effect
Note: By setting this register, the Error Interrupt can be set in the Error Interrupt Status register. In order to generate interrupt signal, both the Error Interrupt Status Enable and Error Interrupt Signal Enable shall be set.

Table 11-1945. MMCHS_FE Instances

Instance	Physical Address
MMCSD_0	2300 0250h
MMCSD_1	2310 0250h

Figure 11-893. MMCHS_FE Register

31	30	29	28	27	26	25	24
RESERVED		FE_BADA	FE_CERR	RESERVED		FE_ADMAE	FE_ACE
R-0h		W-0h	W-0h	R-0h		W-0h	W-0h
23	22	21	20	19	18	17	16
RESERVED	FE_DEB	FE_DCRC	FE_DTO	FE_CIE	FE_CEB	FE_CCRC	FE_CTO
R-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
FE_CNI	RESERVED		FE_ACIE	FE_ACEB	FE_ACCE	FE_ACTO	FE_ACNE
W-0h	R-0h		W-0h	W-0h	W-0h	W-0h	W-0h

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 11-1946. MMCHS_FE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29	FE_BADA	W	0h	Force Event Bad access to data space 0h (W) = No effect, No Interrupt. 1h (W) = Interrupt Forced.
28	FE_CERR	W	0h	Force Event Card error 0h (W) = No effect, No Interrupt. 1h (W) = Interrupt Forced.
27-26	RESERVED	R	0h	Reserved
25	FE_ADMAE	W	0h	Force Event ADMA Error 0h (W) = No effect, No Interrupt. 1h (W) = Interrupt Forced.

Table 11-1946. MMCHS_FE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24	FE_ACE	W	0h	Force Event for Auto CMD Error - For Auto CMD12 and Auto CMD23 0h (W) = No effect, No Interrupt. 1h (W) = Interrupt Forced.
23	RESERVED	R	0h	Reserved
22	FE_DEB	W	0h	Force Event Data End Bit error 0h (W) = No effect, No Interrupt. 1h (W) = Interrupt Forced.
21	FE_DCRC	W	0h	Force Event Data CRC Error 0h (W) = No effect, No Interrupt. 1h (W) = Interrupt Forced.
20	FE_DTO	W	0h	Force Event Data Timeout Error 0h (W) = No effect, No Interrupt. 1h (W) = Interrupt Forced.
19	FE_CIE	W	0h	Force Event Command Index Error 0h (W) = No effect, No Interrupt. 1h (W) = Interrupt Forced.
18	FE_CEB	W	0h	Force Event Command End Bit Error 0h (W) = No effect, No Interrupt. 1h (W) = Interrupt Forced.
17	FE_CCRC	W	0h	Force Event Command CRC Error 0h (W) = No effect, No Interrupt. 1h (W) = Interrupt Forced.
16	FE_CTO	W	0h	Command Timeout Error This bit is set automatically when no response is received within 64 clock cycles from the end bit of the command. For commands that reply within 5 clock cycles - the timeout is still detected at 64 clock cycles. 0h (W) = Status bit unchanged. 1h (W) = Status is cleared.
15-8	RESERVED	R	0h	Reserved
7	FE_CNI	W	0h	Force Event Command not issue by Auto CMD12 error 0h (W) = No effect, No Interrupt. 1h (W) = Interrupt Forced.
6-5	RESERVED	R	0h	Reserved
4	FE_ACIE	W	0h	Force Event for Auto CMD Index Error - For Auto CMD12 and Auto CMD23 0h (W) = No effect, No Interrupt. 1h (W) = Interrupt Forced.
3	FE_ACEB	W	0h	Force Event Auto CMD End Bit Error 0h (W) = No effect, No Interrupt. 1h (W) = Interrupt Forced.
2	FE_ACCE	W	0h	Force Event Auto CMD CRC Error 0h (W) = No effect, No Interrupt. 1h (W) = Interrupt Forced.

Table 11-1946. MMCHS_FE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	FE_ACTO	W	0h	Force Event Auto CMD Timeout Error 0h (W) = No effect, No Interrupt. 1h (W) = Interrupt Forced.
0	FE_ACNE	W	0h	Force Event Auto CMD12 Not Executed 0h (W) = No effect, No Interrupt. 1h (W) = Interrupt Forced.

Table 11-1947. Register Call Summary for MMCHS_FE

MMC/SD Registers • MMCHS_FE Register (Offset = 250h) [reset = 0h]: [0] • MMC/SD Registers: [0]
MMC/SD Functional Description • Test Registers: [0]

11.12.6.31 MMCHS_ADMAES Register (Offset = 254h) [reset = 0h]

MMCHS_ADMAES is shown in Figure 11-894 and described in Table 11-1949.

ADMA Error Status Register
When ADMA Error Interrupt is occurred, the ADMA Error States field in this register holds the ADMA state and the ADMA System Address Register holds the address around the error descriptor. For recovering the error, the Host Driver requires the ADMA state to identify the error descriptor address as follows:
ST_STOP: Previous location set in the ADMA System Address register is the error descriptor address
ST_FDS: Current location set in the ADMA System Address register is the error descriptor address
ST_CADR: This sate is never set because do not generate ADMA error in this state.
ST_TFR: Previous location set in the ADMA System Address register is the error descriptor address
In case of write operation, the Host Driver should use ACMD22 to get the number of written block rather than using this information, since unwritten data may exist in the Host Controller. The Host Controller generates the ADMA Error Interrupt when it detects invalid descriptor data (Valid=0) at the ST_FDS state. In this case, ADMA Error State indicates that an error occurs at ST_FDS state. The Host Driver may find that the Valid bit is not set in the error descriptor.

Table 11-1948. MMCHS_ADMAES Instances

Instance	Physical Address
MMCSD_0	2300 0254h
MMCSD_1	2310 0254h

Figure 11-894. MMCHS_ADMAES Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													LME	AES	
R-0h													R/W-0h	R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-1949. MMCHS_ADMAES Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2	LME	R/W	0h	ADMA Length Mismatch Error (1) While Block Count Enable being set, the total data length specified by the Descriptor table is different from that specified by the Block Count and Block Length. (2) Total data length can not be divided by the block length. 0h (R/W) = No Error. 1h (R/W) = Error.
1-0	AES	R/W	0h	ADMA Error State his field indicates the state of ADMA when error is occurred during ADMA data transfer. This field never indicates "10" because ADMA never stops in this state. 0h (R/W) = ST_STOP (Stop DMA) Contents of SYS_SDR register. 1h (R/W) = ST_STOP (Stop DMA) Points the error descriptor. 2h (R/W) = Never set this state (Not used). 3h (R/W) = ST_TFR (Transfer Data) Points the next of the error descriptor.

Table 11-1950. Register Call Summary for MMCHS_ADMAES

MMC/SD Registers • MMCHS_ADMAES Register (Offset = 254h) [reset = 0h]: [0] • MMC/SD Registers: [0]
MMC/SD Functional Description • Requirements for Descriptors: [0]

11.12.6.32 MMCHS_ADMASAL Register (Offset = 258h) [reset = 0h]

MMCHS_ADMASAL is shown in Figure 11-895 and described in Table 11-1952.

ADMA System address Low bits.

Table 11-1951. MMCHS_ADMASAL Instances

Instance	Physical Address
MMCSD_0	2300 0258h
MMCSD_1	2310 0258h

Figure 11-895. MMCHS_ADMASAL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADMA_A32B																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-1952. MMCHS_ADMASAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADMA_A32B	R/W	0h	ADMA System address 32 bits This register holds byte address of executing command of the Descriptor table. 32-bit Address Descriptor uses lower 32-bit of this register. At the start of ADMA, the Host Driver shall set start address of the Descriptor table. The ADMA increments this register address, which points to next line, when every fetching a Descriptor line. When the ADMA Error Interrupt is generated, this register shall hold valid Descriptor address depending on the ADMA state. The Host Driver shall program Descriptor Table on 32-bit boundary and set 32-bit boundary address to this register. ADMA2 ignores lower 2-bit of this register and assumes it to be 00b.

Table 11-1953. Register Call Summary for MMCHS_ADMASAL

MMC/SD Registers • MMCHS_ADMASAL Register (Offset = 258h) [reset = 0h]: [0] • MMC/SD Registers: [0]
MMC/SD Functional Description • Advanced DMA Description: [0][1][2] • Requirements for Descriptors: [0] • Master DMA Operations: [0][1]

11.12.6.33 MMCHS_PVINITSD Register (Offset = 260h) [reset = 401E0h]

MMCHS_PVINITSD is shown in Figure 11-896 and described in Table 11-1955.

Preset Value for Initialization and Default Speed modes.

Table 11-1954. MMCHS_PVINITSD Instances

Instance	Physical Address
MMCSD_0	2300 0260h
MMCSD_1	2310 0260h

Figure 11-896. MMCHS_PVINITSD Register

31	30	29	28	27	26	25	24
DSDS_SEL		RESERVED			DSCLKGEN_SEL	DSSDCLK_SEL	
R-0h		R-0h			R-0h	R-4h	
23	22	21	20	19	18	17	16
DSSDCLK_SEL							
R-4h							
15	14	13	12	11	10	9	8
INITDS_SEL		RESERVED			INITCLKGEN_SEL	INITSDCLK_SEL	
R-0h		R-0h			R-0h	R-1E0h	
7	6	5	4	3	2	1	0
INITSDCLK_SEL							
R-1E0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1955. MMCHS_PVINITSD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	DSDS_SEL	R	0h	Driver Strength Select Value - Default Speed mode Driver Strength is supported by 1.8 V signaling bus speed modes. This field is meaningless for 3.3 V signaling. 0h (R) = Driver Type B is Selected. 1h (R) = Driver Type A is Selected. 2h (R) = Driver Type C is Selected. 3h (R) = Driver Type D is Selected.
29-27	RESERVED	R	0h	Reserved
26	DSCLKGEN_SEL	R	0h	Clock Generator Select Value - Default Speed mode This bit is effective when Host Controller supports programmable clock generator. 0h (R) = Host Controller Ver2.00 Compatible Clock Generator. 1h (R) = Programmable Clock Generator.
25-16	DSSDCLK_SEL	R	4h	SDCLK Frequency Select Value - Default Speed mode 10-bit preset value to set MMCHS_SYSCTL[15-6] CLKD is described by a host system.
15-14	INITDS_SEL	R	0h	Driver Strength Select Value - Initialization mode Driver Strength is supported by 1.8 V signaling bus speed modes. This field is meaningless for 3.3 V signaling. 0h (R) = Driver Type B is Selected. 1h (R) = Driver Type A is Selected. 2h (R) = Driver Type C is Selected. 3h (R) = Driver Type D is Selected.

Table 11-1955. MMCHS_PVINITSD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13-11	RESERVED	R	0h	Reserved
10	INITCLKGEN_SEL	R	0h	Clock Generator Select Value - Initialization mode This bit is effective when Host Controller supports programmable clock generator. 0h (R) = Host Controller Ver2.00 Compatible Clock Generator. 1h (R) = Programmable Clock Generator.
9-0	INITSDCLK_SEL	R	1E0h	SDCLK Frequency Select Value - Initialization mode 10-bit preset value to set MMCHS_SYSCTL[15-6] CLKD is described by a host system.

Table 11-1956. Register Call Summary for MMCHS_PVINITSD

MMC/SD Registers • MMCHS_PVINITSD Register (Offset = 260h) [reset = 401E0h]: [0] • MMC/SD Registers: [0]
MMC/SD Functional Description • MMC Hardware Status Features: [0]

11.12.6.34 MMCHS_PVHSSDR12 Register (Offset = 264h) [reset = 40002h]

MMCHS_PVHSSDR12 is shown in Figure 11-897 and described in Table 11-1958.

Preset Value for High Speed and SDR12 speed modes.

Table 11-1957. MMCHS_PVHSSDR12 Instances

Instance	Physical Address
MMCSD_0	2300 0264h
MMCSD_1	2310 0264h

Figure 11-897. MMCHS_PVHSSDR12 Register

31	30	29	28	27	26	25	24
SDR12DS_SEL		RESERVED			SDR12CLKGEN_SEL	SDR12SDCLK_SEL	
R-0h		R-0h			R-0h	R-4h	
23	22	21	20	19	18	17	16
SDR12SDCLK_SEL							
R-4h							
15	14	13	12	11	10	9	8
HSDS_SEL		RESERVED			HSCLKGEN_SEL	HSSDCLK_SEL	
R-0h		R-0h			R-0h	R-2h	
7	6	5	4	3	2	1	0
HSSDCLK_SEL							
R-2h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1958. MMCHS_PVHSSDR12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	SDR12DS_SEL	R	0h	Driver Strength Select Value - SDR12 mode Driver Strength is supported by 1.8 V signaling bus speed modes. This field is meaningless for 3.3 V signaling. 0h (R) = Driver Type B is Selected. 1h (R) = Driver Type A is Selected. 2h (R) = Driver Type C is Selected. 3h (R) = Driver Type D is Selected.
29-27	RESERVED	R	0h	Reserved
26	SDR12CLKGEN_SEL	R	0h	Clock Generator Select Value - SDR12 mode This bit is effective when Host Controller supports programmable clock generator. 0h (R) = Host Controller Ver2.00 Compatible Clock Generator. 1h (R) = Programmable Clock Generator.
25-16	SDR12SDCLK_SEL	R	4h	SDCLK Frequency Select Value - SDR12 mode 10-bit preset value to set MMCHS_SYSCTL[15-6] CLKD is described by a host system.
15-14	HSDS_SEL	R	0h	Driver Strength Select Value - High Speed mode Driver Strength is supported by 1.8 V signaling bus speed modes. This field is meaningless for 3.3 V signaling. 0h (R) = Driver Type B is Selected. 1h (R) = Driver Type A is Selected. 2h (R) = Driver Type C is Selected. 3h (R) = Driver Type D is Selected.

Table 11-1958. MMCHS_PVHSSDR12 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13-11	RESERVED	R	0h	Reserved
10	HSCLKGEN_SEL	R	0h	Clock Generator Select Value - High Speed mode This bit is effective when Host Controller supports programmable clock generator. 0h (R) = Host Controller Ver2.00 Compatible Clock Generator. 1h (R) = Programmable Clock Generator.
9-0	HSSDCLK_SEL	R	2h	SDCLK Frequency Select Value - High Speed mode 10-bit preset value to set MMCHS_SYSCTL[15-6] CLKD is described by a host system.

Table 11-1959. Register Call Summary for MMCHS_PVHSSDR12

MMC/SD Registers • MMCHS_PVHSSDR12 Register (Offset = 264h) [reset = 40002h]: [0] • MMC/SD Registers: [0]
MMC/SD Functional Description • MMC Hardware Status Features: [0]

11.12.6.35 MMCHS_PVSDR25SDR50 Register (Offset = 268h) [reset = 10002h]

MMCHS_PVSDR25SDR50 is shown in Figure 11-898 and described in Table 11-1961.

Preset Value for SDR25 and SDR50 speed modes.

Note: SDR50 mode is not supported.

Table 11-1960. MMCHS_PVSDR25SDR50 Instances

Instance	Physical Address
MMCSD_0	2300 0268h
MMCSD_1	2310 0268h

Figure 11-898. MMCHS_PVSDR25SDR50 Register

31	30	29	28	27	26	25	24
SDR50DS_SEL		RESERVED			SDR50CLKGEN_SEL	SDR50SDCLK_SEL	
R-0h		R-0h			R-0h	R-1h	
23	22	21	20	19	18	17	16
SDR50SDCLK_SEL							
R-1h							
15	14	13	12	11	10	9	8
SDR25DS_SEL		RESERVED			SDR25CLKGEN_SEL	SDR25SDCLK_SEL	
R-0h		R-0h			R-0h	R-2h	
7	6	5	4	3	2	1	0
SDR25SDCLK_SEL							
R-2h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1961. MMCHS_PVSDR25SDR50 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	SDR50DS_SEL	R	0h	Driver Strength Select Value - SDR50 mode Driver Strength is supported by 1.8 V signaling bus speed modes. This field is meaningless for 3.3 V signaling. 0h (R) = Driver Type B is Selected. 1h (R) = Driver Type A is Selected. 2h (R) = Driver Type C is Selected. 3h (R) = Driver Type D is Selected. **Note:** SDR50 mode is not supported.
29-27	RESERVED	R	0h	Reserved
26	SDR50CLKGEN_SEL	R	0h	Clock Generator Select Value - SDR50 mode This bit is effective when Host Controller supports programmable clock generator. 0h (R) = Host Controller Ver2.00 Compatible Clock Generator. 1h (R) = Programmable Clock Generator. **Note:** SDR50 mode is not supported.
25-16	SDR50SDCLK_SEL	R	1h	SDCLK Frequency Select Value - SDR50 mode 10-bit preset value to set MMCHS_SYSCTL[15-6] CLKD is described by a host system. **Note:** SDR50 mode is not supported.

Table 11-1961. MMCHS_PVSDR25SDR50 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-14	SDR25DS_SEL	R	0h	Driver Strength Select Value - SDR25 mode Driver Strength is supported by 1.8 V signaling bus speed modes. This field is meaningless for 3.3 V signaling. 0h (R) = Driver Type B is Selected. 1h (R) = Driver Type A is Selected. 2h (R) = Driver Type C is Selected. 3h (R) = Driver Type D is Selected.
13-11	RESERVED	R	0h	Reserved
10	SDR25CLKGEN_SEL	R	0h	Clock Generator Select Value - SDR25 mode This bit is effective when Host Controller supports programmable clock generator. 0h (R) = Host Controller Ver2.00 Compatible Clock Generator. 1h (R) = Programmable Clock Generato.
9-0	SDR25SDCLK_SEL	R	2h	SDCLK Frequency Select Value - SDR25 mode 10-bit preset value to set MMCHS_SYSCTL[15-6] CLKD is described by a host system.

Table 11-1962. Register Call Summary for MMCHS_PVSDR25SDR50

MMC/SD Registers • MMCHS_PVSDR25SDR50 Register (Offset = 268h) [reset = 10002h]: [0] • MMC/SD Registers: [0]
MMC/SD Functional Description • MMC Hardware Status Features: [0]

11.12.6.36 MMCHS_PVSDR104DDR50 Register (Offset = 26Ch) [reset = 20000h]

MMCHS_PVSDR104DDR50 is shown in Figure 11-899 and described in Table 11-1964.

Preset Value for SDR104 and DDR50 speed modes.

Note: SDR104 mode is not supported.

Table 11-1963. MMCHS_PVSDR104DDR50 Instances

Instance	Physical Address
MMCSD_0	2300 026Ch
MMCSD_1	2310 026Ch

Figure 11-899. MMCHS_PVSDR104DDR50 Register

31	30	29	28	27	26	25	24
DDR50DS_SEL		RESERVED			DDR50CLKGEN_SEL	DDR50SDCLK_SEL	
R-0h		R-0h			R-0h	R-2h	
23	22	21	20	19	18	17	16
DDR50SDCLK_SEL							
R-2h							
15	14	13	12	11	10	9	8
SDR104DS_SEL		RESERVED			SDR104CLKGEN_SEL	SDR104SDCLK_SEL	
R-0h		R-0h			R-0h	R-0h	
7	6	5	4	3	2	1	0
SDR104SDCLK_SEL							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1964. MMCHS_PVSDR104DDR50 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	DDR50DS_SEL	R	0h	Driver Strength Select Value - DDR50 mode Driver Strength is supported by 1.8 V signaling bus speed modes. This field is meaningless for 3.3 V signaling. 0h (R) = Driver Type B is Selected. 1h (R) = Driver Type A is Selected. 2h (R) = Driver Type C is Selected. 3h (R) = Driver Type D is Selected.
29-27	RESERVED	R	0h	Reserved
26	DDR50CLKGEN_SEL	R	0h	Clock Generator Select Value - DDR50 mode This bit is effective when Host Controller supports programmable clock generator. 0h (R) = Host Controller Ver2.00 Compatible Clock Generator. 1h (R) = Programmable Clock Generator.
25-16	DDR50SDCLK_SEL	R	2h	SDCLK Frequency Select Value - DDR50 mode 10-bit preset value to set MMCHS_SYSCTL[15-6] CLKD is described by a host system.

Table 11-1964. MMCHS_PVSDR104DDR50 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-14	SDR104DS_SEL	R	0h	Driver Strength Select Value - SDR104 mode Driver Strength is supported by 1.8 V signaling bus speed modes. This field is meaningless for 3.3 V signaling. 0h (R) = Driver Type B is Selected. 1h (R) = Driver Type A is Selected. 2h (R) = Driver Type C is Selected. 3h (R) = Driver Type D is Selected. **Note:** SDR104 mode is not supported.
13-11	RESERVED	R	0h	Reserved
10	SDR104CLKGEN_SEL	R	0h	Clock Generator Select Value - SDR104 mode This bit is effective when Host Controller supports programmable clock generator. 0h (R) = Host Controller Ver2.00 Compatible Clock Generator. 1h (R) = Programmable Clock Generator. **Note:** SDR104 mode is not supported.
9-0	SDR104SDCLK_SEL	R	0h	SDCLK Frequency Select Value - SDR104 mode 10-bit preset value to set MMCHS_SYSCTL[15-6] CLKD is described by a host system. **Note:** SDR104 mode is not supported.

Table 11-1965. Register Call Summary for MMCHS_PVSDR104DDR50

MMC/SD Registers • MMCHS_PVSDR104DDR50 Register (Offset = 26Ch) [reset = 20000h]: [0] • MMC/SD Registers: [0]
MMC/SD Functional Description • MMC Hardware Status Features: [0]

11.12.6.37 MMCHS_REV Register (Offset = 2FCh) [reset = -020000h]

MMCHS_REV is shown in Figure 11-900 and described in Table 11-1967.

Versions Register
This register contains the hard coded RTL vendor revision number, the version number of SD specification compliancy and a slot status bit.
MMCHS_REV[31-16] = Host controller version
MMCHS_REV[15-0] = Slot Interrupt Status.

Table 11-1966. MMCHS_REV Instances

Instance	Physical Address
MMCSD_0	2300 02FCh
MMCSD_1	2310 02FCh

Figure 11-900. MMCHS_REV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VREV								SREV							
R--h								R-2h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															SIS
R-0h															R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-1967. MMCHS_REV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	VREV	R	-h	Vendor Version Number: MMC revision [7-4] Major revision [3-0] Minor revision Examples: 10h for 1.0 21h for 2.1.
23-16	SREV	R	2h	Specification Version Number This status indicates the Host Controller Spec. Version. The upper and lower 4-bits indicate the version. 0h (R) = SD Host Specification Version 1.00. 1h (R) = SD Host Specification Version 2.00 - Including the feature of the ADMA and Test Register. 2h (R) = SD Host Specification Version 3.00. 3h (R) = Reserved.
15-1	RESERVED	R	0h	Reserved
0	SIS	R	0h	Slot Interrupt Status This status bit indicates the inverted state of interrupt signal for the module. By a power on reset or by setting a software reset for all (MMCHS_HCTL[SRA]), the interrupt signal shall be de-asserted and this status shall read 0.

Table 11-1968. Register Call Summary for MMCHS_REV

MMC/SD Registers
- MMCHS_REV Register (Offset = 2FCh) [reset = -020000h]: [0][1][2]
- MMC/SD Registers: [0]

11.13 Networking Subsystem (NSS)

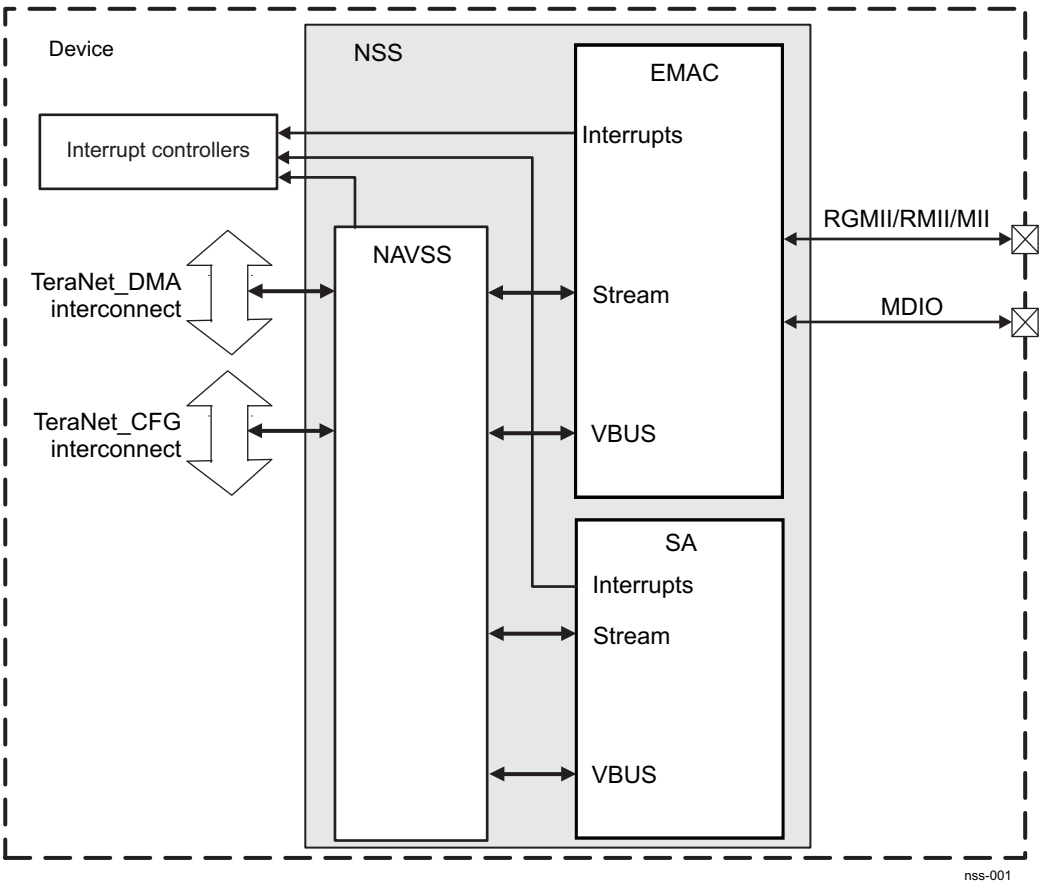
This chapter describes the features and functions of the Networking Subsystem (NSS) in the device.

11.13.1 NSS Overview

Networking Subsystem (NSS) consists of DMA/Queue Management components – Navigator Subsystem (NAVSS), an Ethernet MAC (EMAC) Subsystem, and a packet Security Accelerator (SA).

Figure 11-901 shows the NSS overview.

Figure 11-901. NSS Overview



NAVSS – The Navigator Subsystem consists of a CPPI DMA, CPPI Queue Manager (QM), and miscellanous infrastructure. The NAVSS also contains an embedded Streaming Switch, which connects the CDMA with the I/O Peripheral streaming ports (EMAC and SA).

EMAC – The two-port (one Ethernet and one host) Gigabit Ethernet Subsystem functions as an Ethernet MAC. Packets can be transmitted on one of 8 priorities by being queued through the CDMA. Received packets are queued to the CDMA and steered to specific queues via the RX FLOW field.

SA – The Security Accelerator (SA) is an encryption/decryption accelerator. Packets for encryption/decryption can be queued to it via the CDMA threads. Ask your TI representative for SA specification with programming details.

NOTE: Networking Subsystem is a notional subsystem. Besides full-feature mode, NSS modules can be used also in NAVSS-only, EMAC + NAVSS, and SA + NAVSS modes, if required.

11.13.1.1 NSS Features

The NSS, presented by its general sub-components, supports the following features:

- NAVSS
  - High Performance CPPI DMA Controller, 32 Receive Flows, 4 Loopback threads for infrastructure mode
  - CPPI Queue Manager (QM) features:
    - Single QM
    - Supports up to 128 queues – 21 QPEND signals for TX use, remaining 107 QPEND signals are for host use
    - 2048 buffers supported in Internal Linking RAM
    - Two Queue Proxies provided for host interaction (one per DSP and ARMSS)
      - Queue Proxy 0 assigned to DSP
      - Queue Proxy 1 assigned to ARMSS
  - Support for SER protection (SECDED)
- EMAC Subsystem
  - One Gigabit Ethernet port: MII/RMII/RGMII interfaces
    - Supports 10/100/1000 Mbps full duplex
    - Supports 10/100 Mbps half duplex
  - One Host Port 0 CPPI Streaming Packet Interface (PSI)
  - Support Ethernet Audio/Video Bridging (eAVB) (P802.1Qav/D6.0)
  - Maximum frame size 2016 bytes (2020 bytes with VLAN)
  - Eight priority level QOS support (802.1p)
  - IEEE 1588 (2008 annex D, annex E, and annex F) to facilitate Audio/Video bridge 802.1AS Precision Time Protocol
    - Timestamp module capable of time stamping external timesync events like Pulse Per Second and also generating Pulse Per Second outputs
    - CPTS module that supports time stamping for IEEE1588 with support for 8 hardware push events and generation of compare output pulses
  - DSCP Priority Mapping (IPv4 and IPv6)
  - Energy Efficient Ethernet Support (802.3az)
  - Maximum frame size 2016 bytes (2020 with VLAN)
  - Address Lookup Engine (ALE)
  - Castagnoli or Ethernet CRC selectable for Ethernet ingress/egress (Host Port0 CRC is Ethernet only)
  - MDIO module for PHY management
  - EtherStats and 802.3Stats RMON statistics gathering
  - Support for SER protection (SECDED)
- Security Accelerator (SA)
  - Support IPSec and SRTP protocol stack
  - Support various encryption modes and algorithms such as:
    - ECB, CBC, CFB, OFB, F8, CTR, CBC-MAC, CCM, GCM, GMAC and AES-CMAC
    - AES, DES, 3DES, SHA-1, SHA-2 (224, 256-bit operation) and MD5
  - Support for True random number generator (TRNG) and Public Key Accelerator (PKA)
  - Support for SER protection (SECDED)

The NSS does not support the following features:

- No external queue RAM supported

- Priority Based Flow Control is not supported.
- No Castignoli CRC to Host CPPI port.

11.13.1.2 NSS Modes of Operation

The NSS can operate in five modes:

1. Full Disable – In this mode, all modules are in clock stop state
2. Infrastructure Mode – In this mode, the SA and EMAC are in clock stop state, and the NAVSS is enabled. In this mode, the CDMA and QM can be used for infrastructure DMA purposes
3. Security-only Mode – In this mode, the NAVSS and SA are enabled, and EMAC is in clock stop state. In this mode, the SA can be used for encryption/decryption, and NAVSS can be used for infrastructure purposes
4. Ethernet-only Mode – In this mode, the NAVSS and EMAC are enabled, and SA is in clock stop state. In this mode, the EMAC may be used to send and receive packets, and CDMA can be used for infrastructure purposes
5. Full Enable – In this mode, all sub-modules are enabled. All functions are available.

NOTE: NSS power domain (NAVSS + EMAC) is standalone and can be used with CRYPTO power domain either enabled or disabled. However, it is not possible to use CRYPTO PD when the NSS PD is OFF. During power-up sequence it is required to have the NSS power domain ON prior to turning the CRYPTO domain ON. Similarly, during power down sequence, it is required to power down CRYPTO domain prior to powering down the NSS domain.

11.13.1.3 NSS Memory Map

Table 11-1969 covers the main NSS components with their addresses and links to register descriptions.

Table 11-1969. NSS Top Level Memory Map

Address Offset	Physical Address	Size (bytes)	Region	Access	Reference
0000 0000h	0400 0000h	256	NAVSS Configuration Registers	ARMSS/DSP	Section 11.13.5.2.1
0000 0100h	0400 0100h	1792	Reserved (free)		
0000 0800h	0400 0800h	1024	NAVSS ECC Aggregator	ARMSS/DSP	Section 11.13.5.1.6
0000 0C00h	0400 0C00h	1024	Reserved (free)		
0000 1000h	0400 1000h	4096	INTD Registers	ARMSS/DSP	Section 11.13.5.2.2
0000 2000h	0400 2000h	56 K	Reserved (free)		
0001 0000h	0401 0000h	256	CPPI DMA Configuration Registers	ARMSS/DSP	Section 11.13.5.2.3.1
0001 0100h	0401 0100h	128	CPPI DMA Tx Scheduler Registers	ARMSS/DSP	Section 11.13.5.2.3.2
0001 0180h	0401 0180h	3712	Reserved		
0001 1000h	0401 1000h	4096	CPPI DMA Tx Channel Configuration Registers	ARMSS/DSP	Section 11.13.5.2.3.3
0001 2000h	0401 2000h	4096	CPPI DMA Rx Channel Configuration Registers	ARMSS/DSP	Section 11.13.5.2.3.4
0001 3000h	0401 3000h	4096	CPPI DMA Rx Flow Configuration Registers	ARMSS/DSP	Section 11.13.5.2.3.5
0001 4000h	0401 4000h	44 K	Reserved (free)		
0002 0000h	0402 0000h	8192	QM Linking RAM	QM	
0004 0000h	0404 0000h	256	QM Configuration	ARMSS/DSP	Section 11.13.5.2.4.1
0004 0200h	0404 0200h	256	Queue Proxy 0 Configuration	DSP	

Table 11-1969. NSS Top Level Memory Map (continued)

Address Offset	Physical Address	Size (bytes)	Region	Access	Reference
			Queue Proxy 1 Configuration	ARMSS	
0008 0000h	0408 0000h	2048	QM Descriptor Regions	ARMSS/DSP	Section 11.13.5.2.4.2
000C 0000h	040C 0000h	2048	QM Queue/De-queue	ARMSS/DSP	Section 11.13.5.2.4.3
0010 0000h	0410 0000h	2048	QM Queue Status	ARMSS/DSP	Section 11.13.5.2.4.4
0014 0000h	0414 0000h	2048	Queue Proxy 0 Queues	DSP	
			Queue Proxy 1 Queues	ARMSS	
0020 0000h	0420 0000h	256	EMAC SS Configuration Registers	ARMSS/DSP	Section 11.13.5.1
0020 0F00h	0420 0F00h	256	MDIO Registers	ARMSS/DSP	Section 11.13.5.1.5
0022 0000h	0422 0000h	124 K	CPSW_2U Registers (CPSW, Port, ALE, Statistics, and CPTS)	ARMSS/DSP	Section 11.13.5.1.2 Section 11.13.5.1.3 Section 11.13.5.1.4
0023 F000h	0423 F000h	1024	EMAC ECC Aggregator	ARMSS/DSP	Section 11.13.5.1.6
0040 0000h	0440 0000h		SA	ARMSS/DSP	See SA spec.

NOTE: The NSS contains 2K (2048) internal linking RAM entries. The internal linking RAM control in the QM actually only supports 2K-1 entires. In order to use the entire 2K entries, the following procedure must be used:

1. Program linking RAM0 size to 0
2. Program the linking RAM1 address to the internal linking RAM address (0402 0000h)

11.13.1.4 Thread Map

Streaming Switch connects the CDMA with the I/O Peripheral streaming ports (EMAC, SA). The mapping of threads in the streaming switch is contained in Table 11-1970.

Table 11-1970. NAVSS Thread Map

	Slave	CDMA	SA	EMAC
	Slave Threads	15	1	8
Master	**Master Threads**			
CDMA	21	0–3/0–3	4/0	5–12/0–7
SA	2	0–1/4–5		
EMAC	1	0/6		

11.13.1.5 CDMA Receive Flow Mapping

There is not a one-to-one mapping of receive threads and receive flows on the CDMA. Though it is possible to use the default receive flow for a thread (default flow number equals thread number), it is recommended that all flows be explicitly defined, and that all packets received from the streaming domain use a specified flow, not a default flow. This section contains an example for how to assign receive flows. Though it is recommended to use explicit receive flows in each packet, this example does allow for the use of default flows on everything but the Ethernet MAC. The CPSW uses a single thread, but requires a contiguous block of 16 receive flows (8 for priority based classification and 8 for other classifiers), the base of which is specified in CPSW_P0_FLOW_ID_OFFSET register in the CPSW. The mapping of threads is shown in Table 11-1970 *NAVSS Thread Map*. Table 11-1971 shows an example of a receive flow map.

Table 11-1971. CDMA Receive Flow Mapping Example

Source	CDMA Receive Thread(s)	CDMA Receive Flow(s)
CDMA (Infrastructure Channels)	0–3	0–3
SA	4–5	4–5
EMAC	6	15–30
Spare flows		6, 31

11.13.1.6 Network Subsystem High-Level Initialization

High-level initialization of NSS must follow the below sequence.

1. Turn on the NSS power domain
2. Ungate the clocks for all modules used
3. Configure the queue manager
4. Configure the CDMA
   a. Configure the linking RAM
   b. Initialize descriptors
   c. Configure receive flows
   d. Enable transmit channels
   e. Enable receive channels
5. Initialize the Ethernet subsystem
6. Configure the SA.

11.13.1.7 Error Packets Forwarded Through CDMA

The CDMA streaming transmit interface does not have PKT_ERROR bits. Because of this, if a packet is received with error bits set and configured to be forwarded directly to another block without host intervention, that packet may be treated as if it is a good packet. For example, if a packet is received from the Ethernet MAC with error bits set, goes to the CDMA, then is transmited to the SA without host intervention, the SA may treat that packet as a good packet. In order to avoid this, a host process must intervene before transmitting to the SA (or other block) to make sure it wasn't an error packet, or the data flow must be tolerant of processing error packets and be able to check them at some other point.

11.13.2 Navigator Subsystem (NAVSS)

11.13.2.1 Overview

NAVSS specifies the data structures used by Texas Instruments standard communications modules to facilitate direct memory access (DMA) and to provide a consistent application programming interface (API) to the host software in multi-core devices. The data structures and the API used to manipulate them will be jointly referred to as NAVSS.

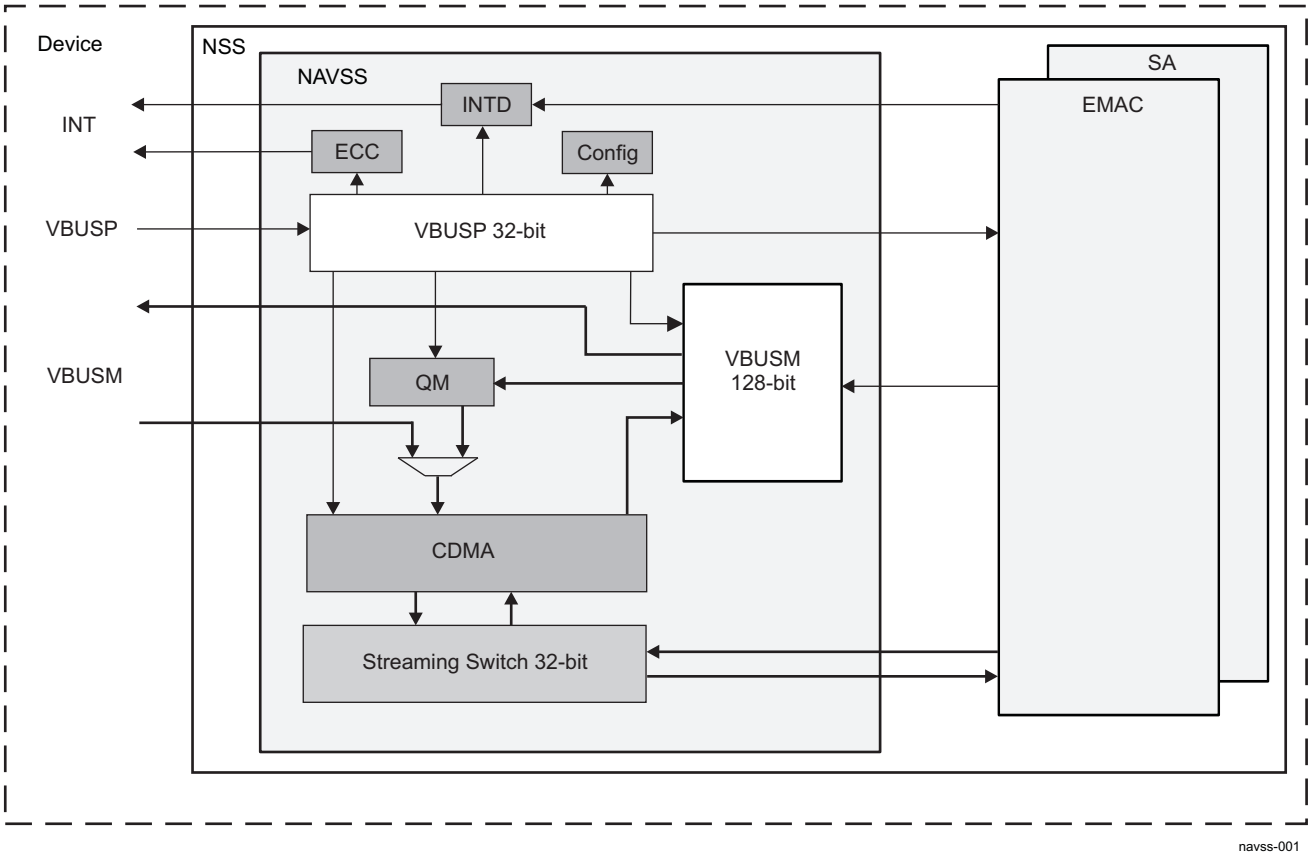
Frequent tasks are commonly offloaded from the host processor to peripheral hardware to increase system performance. Significant performance gains may result from careful design of the host software and communication module interface. In networking systems, packet transmission and reception are critical tasks. Texas Instruments has developed the NAVSS standard, which is aimed at maximizing the efficiency of interaction between the host software and communications modules.

The design goals for NAVSS are as follows:

- Minimize host interaction
- Maximize memory use efficiency
- Maximize bus burst efficiency
- Maximize symmetry between transmit/receive operations
- Maximize scalability for number of connections/buffer sizes/queue sizes/protocols supported
- Minimize protocol specific features
- Minimize complexity

Navigator subsystem hardware components are shows on Figure 11-902. Slave peripherals, Ethernet MAC and Security accelerator, are shown for illustrative purposes.

Figure 11-902. Navigator Subsystem Hardware Block Diagram



11.13.2.1.1 Queue Manager

The queue manager (QM) is a hardware module that is responsible for accelerating management of the packet queues. Packets are added to a packet queue by writing the 32-bit descriptor address to a particular memory mapped location in the queue manager module. Packets are de-queued by reading the same location for that particular queue. NAVSS queue manager modules are capable of queuing only descriptors that have been allocated from the descriptor regions of the associated queue manager.

11.13.2.1.2 Packet DMA (CDMA)

The Packet DMA is a CPPI DMA in which data destination is determined by a destination and free descriptor queue index, not an absolute memory address. In receive mode, the CDMA fetches a free descriptor, traverses the descriptor to find the buffer, CDMA transfers the payload into the buffer, and puts the descriptor into the destination queue. In transmit mode, the CDMA pops the descriptor from the TX queue, traverses the descriptor, reads the payload from the buffer, and DMA transfers the payload to the transmit port.

11.13.2.1.3 Navigator Cloud

A Navigator Cloud is a set of CDMAs and descriptors. Neither CDMAs nor descriptors address the physical Queue Manager(s) directly, but instead use a *queue_manager:queue_number* (qmgr:qnum) notation and registers to create a logical mapping into the physical Queue Manager(s). All CDMAs with the same logical mapping are said to be part of the same Navigator Cloud. A descriptor can be sent to any CDMA in the same cloud, but may or may not transfer correctly through CDMAs in different clouds. A non-compatible logical qmgr:qnum mapping will cause descriptors to arrive in unexpected queues, potentially causing a memory leak.

It is possible to send a descriptor from one cloud to another, but each qmgr:qnum reference must point to the same physical queue for the CDMAs in both clouds. Another way to say this is by example: Let CDMA 1 and 2 have the same base addresses programmed for logical QM0 and QM1 in their respective QMn Base Address registers, but their QM2 and QM3 base addresses are different (so by definition they represent different clouds). Any descriptor traveling between them must reference only QM0 and/or QM1 in every descriptor and RX Flow qmgr:qnum fields. This is especially true if the RX (output) queue for the first CDMA is the same physical queue as the TX (input) queue for the second CDMA.

11.13.2.1.4 Virtualization

Physical memory addresses can be virtualized by using the MPAX units inside the MSMC controller and DSP. It is important to understand how Navigator uses virtualized addresses in a system where both physical and virtual addresses may be used by various components, or when different mappings may address the same physical memory. Here is the summary:

- The Queue Manager does not understand memory translations. It simply converts QM pushed addresses to linking RAM indexes based on the programming of QMSS Memory Regions (see Section 11.13.5.2.4.2, *QM Descriptor Regions*). Virtual addresses can be used, but they must be programmed into the QMSS Memory Regions and used for every push and descriptor reference. Using a physical address when virtual addresses have been programmed into QMSS will result in erroneous behavior.
- The CDMA also does not understand memory translations. It simply makes VBUS transactions using popped descriptor addresses and the memory references inside descriptors. A virtual address used by the CDMA will be translated by MPAX to a physical address. Because the CDMA uses popped addresses, addresses embedded in descriptors, and Rx Flow configurations (containing qmgr:qnum mappings), these must all present a unified view of memory or the result of transfers will be unexpected.

11.13.2.1.5 ARMSS-DSP Shared Use

Most of the time, an ARMSS's memory virtualization will be different than that of the DSPs. Both ARMSSs and DSPs should define their own Navigator Cloud(s). This is an easy and efficient way to keep resources separated. However, Most ARMSS-DSP applications require transferring data between them. There are at least two ways to do this: 1) either use a CDMA to transfer data from one cloud to another, or 2) create a common shared area to be used by both. The common shared area is the preferred approach, because CDMA loading may be reduced, and because configuring two clouds to communicate can sometimes be difficult.

In the common shared area approach, one or more Navigator Clouds are defined specifically for ARMSS-DSP data transfers. This means that memory virtualization is the same (ARMSS, DSP and QMSS use the same address regions whether virtual or physical), they use common descriptor memory regions, and all descriptor references point to memory and queues containing descriptors from one of the common descriptor memory regions.

In this approach, either the ARMSS or DSP writes data to the common shared area and the recipient is notified. This is done without the CDMA because no data transfer is necessary, and notification is accomplished using the QM by itself. Notification occurs with either a queue pend queue and an interrupt on the recipient core, or the recipient core polling on the receive queue.

11.13.2.1.6 NAVSS Definition of Terms

Host— The host is an intelligent system resource that configures and manages each communications control module. The host is responsible for allocating memory, initializing all data structures, and responding to port interrupts. In a typical system, the Host can be a DSP, ARMSS, or other.

Port— A port is the communications module (peripheral hardware) that contains the control logic for Direct Memory Access for a single transmit/receive interface or set of interfaces. Each port may have multiple communication channels that transfer data using homogenous or heterogeneous protocols. A port is usually subdivided into transmit and receive pairs which are independent of each other.

Channel— A channel refers to the sub-division of information (flows) that is transported across a DMA engine. Each channel has associated state information. Channels are used to segregate information flows based on the protocol used, scheduling requirements (for example, CBR, VBR, ABR), or concurrency requirements (that is, blocking avoidance). Information flow within a channel is a stream of strongly ordered information.

Data Buffer— A data buffer is a single data structure that contains payload information for transmission to or reception from a channel.

Buffer Descriptor— A buffer descriptor is a single data structure that contains information about one or more data buffers.

Packet Descriptor— A packet descriptor is another name for the first buffer descriptor within a packet. Some fields within a data buffer descriptor are only valid when it is a packet descriptor including the tags, packet length, packet type, and flags. All Monolithic type descriptors are packet descriptors (and are also a Data Buffer).

Free Descriptor/Buffer Queue— A free descriptor/buffer queue is a hardware managed list of available descriptors with prelinked empty buffers that are to be used by the receive ports for host type descriptors. Free Descriptor/Buffer Queues are implemented by the Queue Manager.

Free Descriptor Queue— A free descriptor queue is a hardware managed list of available descriptors that are not yet linked with buffers that are to be used by the receive ports for monolithic type descriptors. Free Descriptor Queues are implemented by the Queue Manager

Packet Queue— A packet queue is a hardware managed list of valid (that is, populated) packet descriptors that is used for forwarding a packet from one entity to another for any number of purposes.

Queue Manager— The queue manager is a hardware module that is responsible for accelerating management of the packet queues. Packets are added to a packet queue by writing the 32-bit descriptor address to a particular memory mapped location in the Queue Manager module. Packets are de-queued by reading the same location for that particular queue.

Memory— Memory is an area of data storage managed by the host. This area is visible to the port as a 32-bit addressable area.

Device Driver— A device driver is application independent software that runs on the host for purposes abstracting the low level hardware so that upper level software can use the hardware without knowing every bit field location or initialization sequence.. General device driver functions include port initialization, transmit packet queuing, and receive packet processing.

SOP— Start of Packet. This refers to the descriptor/buffer that is the first buffer in a packet.

MOP— Middle of Packet. This refers to the descriptors/buffers that are neither the first or last buffers in a packet.

EOP— End of Packet. This refers to the descriptor/buffer that is the last buffer in a packet.

11.13.2.2 NAVSS Integration

Table 11-1972 through Table 11-2017 summarize the integration of the NAVSS in the device.

Table 11-1972. NAVSS Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
NAVSS	PD2 (NSS)	LPSC3	TeraNet_DMA TeraNet_CFG

Table 11-1973. NAVSS Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
NAVSS	VCLK	CHIP_CLK1/3	PLL Controller	VBUS interface and functional clock
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
NAVSS	NAVSS_MOD_G_RST	MOD_G_RST	LPSC3	Main NAVSS reset. Same as EMAC reset (LPSC3).

Table 11-1974. NAVSS Hardware Requests

Interrupt Requests						
Module Instance	**Event Name**	**Mapped To Input Event [Number]**				**Description**
		ARM GIC	**CIC**	**DSP INTC**	**PMMC INTC**	
NAVSS	CDMA0_STARVE_INTR	[104]	[264]	-	-	CDMA Buffer Starvation Interrupt
	NAVSS_ECC_INTR	[107]	[267]	-	-	Indicates ECC Error in the NAVSS
	PENDING_TXQ_PEND[3:0]	-	-	[31:28]	-	These are queue pending bits representing the non-empty status of transmit queues inside the NAVSS. PENDING_TXQ_PEND[106:0] correspond to queues 127:21. Queues 20:0 are used internally to the NAVSS.
	PENDING_TXQ_PEND[31:0]	[303:272]	[317:286]	-	-	
	PENDING_TXQ_PEND[35:32]	[307:304]	-	-	-	
	PENDING_TXQ_PEND[39:36]	-	-	[27:24]	-	
	PENDING_TXQ_PEND[47:44]	-	-	-	[63:60]	
	PENDING_TXQ_PEND[55:52](1)	-	-	-	-	
	PENDING_TXQ_PEND[63:56]	[315:308]	[225:318]	-	-	
	PENDING_TXQ_PEND[67:64]	-	[229:226]	-	-	
	PENDING_TXQ_PEND[71:68]	-	[333:330]	-	-	
	PENDING_TXQ_PEND[106:72](2)	-	-	-	-	

DMA Requests			
Module Instance	**Event Name**	**Mapped To Input Event [Number]**	**Description**
		EDMACC0	
NAVSS	PENDING_TXQ_PEND[51:48]	[43:40]	These are queue pending bits representing the non-empty status of transmit queues inside the NAVSS.
	PENDING_TXQ_PEND[67:64]	[47:44]	

(1) Not Assigned
(2) Software Use

NOTE: For more information about the device interrupt controllers, see Chapter 9, *Interrupts*.

11.13.2.3 Operational Concepts

This section introduces the data movement concepts and data structures used by NAVSS. Thorough understanding of these concepts is necessary for effective programming of the device. Low-level (bit field) descriptions are provided in later chapters.

11.13.2.3.1 Packets

A packet is the logical grouping of a descriptor and the payload data attached to it. The payload data may be referred to as *packet data* or a *data buffer*, and depending on the descriptor type, may be contiguous with the descriptor fields, or may be somewhere else in memory with a pointer stored in the descriptor.

11.13.2.3.2 Data Buffers

A data buffer is a byte-aligned contiguous block of memory used to store packet payload data. Each buffer is described in an entry in either a packet descriptor or in a buffer descriptor. A data buffer may hold any portion of a packet and may be linked together (via descriptors) with other buffers to form packets. Data buffers may be allocated anywhere within the 32-bit memory space.

The Buffer Length field of the packet/buffer descriptor indicates the number of valid data bytes in the buffer. There may be from 1 to 4M-1 valid data bytes in each buffer.

11.13.2.3.3 Queues

Queues are used to hold pointers to packets while they are being passed between the host and/or any of the ports in the system. Queues are maintained within the Queue Manager module.

11.13.2.3.3.1 Packet Queuing

Queuing of packets onto a packet queue is accomplished by writing a pointer to the Packet Descriptor (and in some cases the length of the packet) into a specific set of addresses within the Queue Manager module. Packets may be queued either onto the head or the tail of the queue and this is selected based on a bit in the QM_QUEUE_REG_C_0. By default, packets will be added to the tail of a Queue if the QM_QUEUE_REG_C_0 has not been written. The Queue Manager provides a unique set of addresses for adding packets for each queue that it manages.

See Section 11.13.2.4.1, *Queue Manager Operation* for more details.

11.13.2.3.3.2 Packet De-queuing

De-queuing of packets from a packet queue is accomplished by reading the head packet pointer from the corresponding address in the Queue Manager. After the head pointer has been read, the Queue Manager will invalidate the head pointer and will replace it with the next packet pointer in the queue. This functionality which is implemented in the Queue Manager prevents the ports from needing to traverse linked lists and allows for certain optimizations to be performed within the Queue Manager.

See Section 11.13.2.4.1, *Queue Manager Operation* for more details.

11.13.2.3.4 Queue Types

This section describes the various types of queues that are used for transmitting and receiving packets through NAVSS.

11.13.2.3.4.1 Transmit Queues

These special queues are called *transmit queues* (the mappings are described in Table 11-1974) used to store the packets that are waiting to be transmitted. Tx ports will require one or more packet queues dedicated to each transmit channel for this purpose. Multiple queues per channel may facilitate Quality of Service (QoS) in some applications.

11.13.2.3.4.2 Transmit Completion Queues

Tx ports also use packet queues referred to as *transmit completion queues* to return packets to the host after they are transmitted. The number of queues required by the port or system for this purpose is driven by the requirements of the garbage collection software within the driver. Tx completion queues are only used when the packet descriptor indicates that the packet should be returned to a queue instead of directly recycling the descriptors to their queues of origin.

11.13.2.3.4.3 Receive Queues

Rx ports use packet queues referred to as *receive queues* to forward completed received packets to the host or another peer port entity. Rx ports may be configured in various ways to forward the received packets onto any number of receive queues. Rx ports may choose to queue their receive packets based strictly on Rx channel, protocol type, priority, direct forwarding requirements (that is, to another ports Tx queue) or any combination of these and this is application specific. In many cases the receive queue is in fact also a transmit queue for another peer entity. It is just a matter of semantics with respect to which port is referenced in the system.

11.13.2.3.4.4 Free Descriptor Queues

Rx ports use queues referred to as *free descriptor queues* to allocate empty monolithic descriptors. The entries on the Free Descriptor Queues do not have any buffers attached as buffers are allocated as part of the descriptor for monolithic type packets.

11.13.2.3.4.5 Free Descriptor/Buffer Queues

Rx ports use queues referred to as *free descriptor/buffer queues* to allocated empty host type descriptors. The entries on the Free Descriptor/Buffer Queues have pre-attached empty buffers whose size and location are described in the *original buffer information* fields in the descriptor. The host is required to allocate both the descriptor and buffer and pre-link them prior to adding a descriptor to a free descriptor/buffer queue.

11.13.2.3.5 Descriptors

Descriptors are small memory areas that describe the packet of data to be transferred through the system.

Descriptor types are discussed and shown in bit-level detail in Section 11.13.2.5, *Descriptor Layouts*, but briefly, the descriptor types are:

11.13.2.3.5.1 Host Packet Descriptor

Host packet descriptors have a fixed size information (or description) area that contains a pointer to a data buffer, and optionally, a pointer to link one or more host buffer descriptors. Host packets are linked in TX by the host application, and by the RX DMA in RX (host packets should not be prelinked when creating an RX FDQ during initialization).

11.13.2.3.5.2 Host Buffer Descriptor

Host buffer descriptors are interchangeable in descriptor size with host packets, but are never placed as the first link of a packet (this is referred to as *start of packet*). They can contain links to other host buffer descriptors.

11.13.2.3.5.3 Monolithic Packet Descriptor

Monolithic packet descriptors differ from host packet descriptors in that the descriptor area also contains the payload data, whereas the host packet contains a pointer to buffer located elsewhere. Monolithic packets are simpler to deal with, but are not as flexible as host packets.

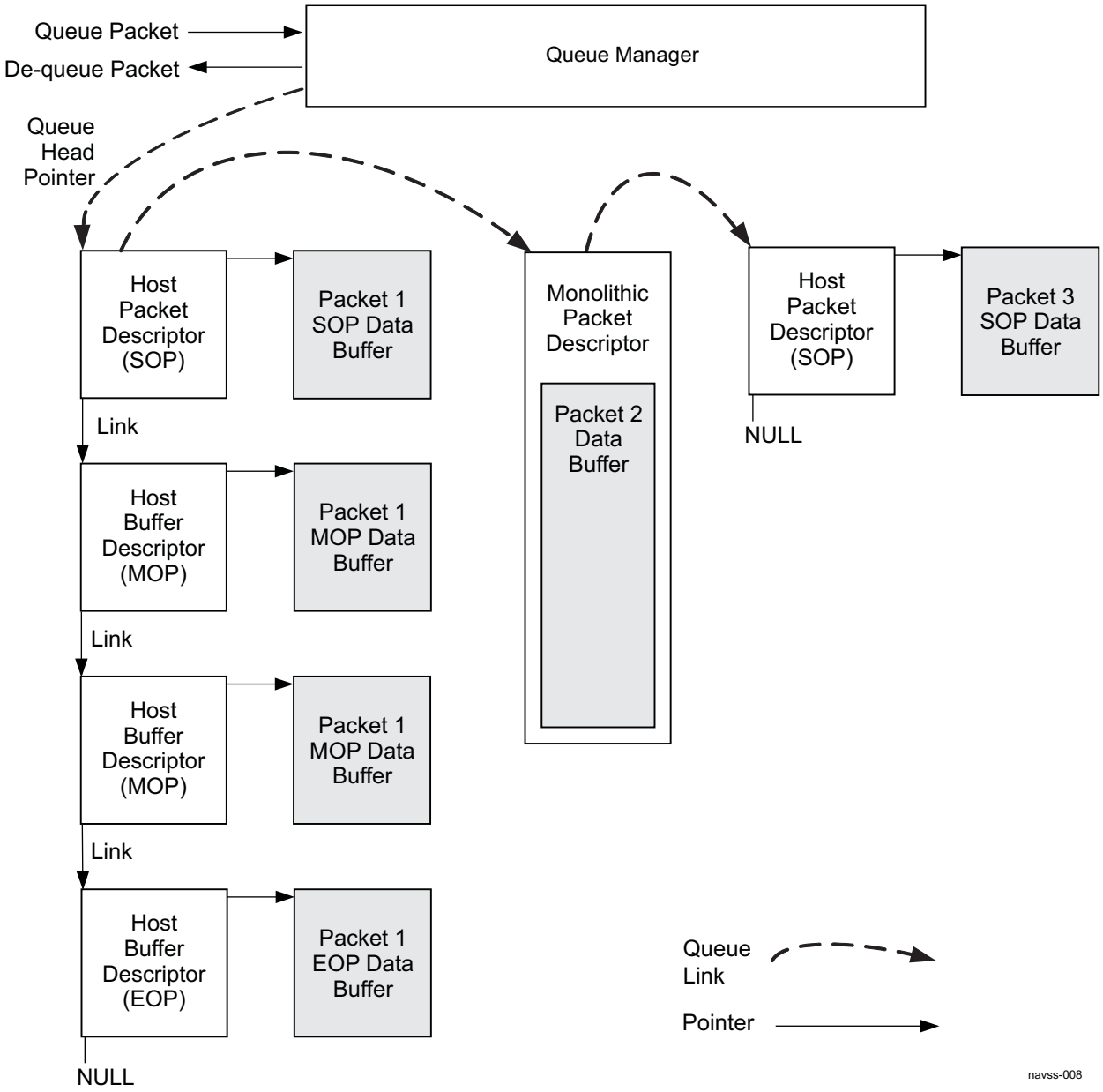
Table 11-1975 shows the different types of descriptors and their characteristics.

Table 11-1975. Descriptor Types and Attributes

Descriptor Type	Includes Valid Packet Info	Provided Number of Slots to Link In External Data Buffers	Provides Local Packet Data Storage	Provides Local Protocol Specific Storage	Provides Slot to Link Additional Descriptors Within Same Packet	Description
Host Packet Descriptor	✓	1		✓	✓	Used to describe packet and SOP buffer in applications that require Host OS compatible data structures (i.e. applications where the descriptors and buffers cannot be managed independently but must instead be pre-linked by the Host software). These applications inherently require a separate descriptor for each buffer.
Host Buffer Descriptor		1			✓	Used to describe non-SOP buffers in applications that require Host OS compatible data structures
Monolithic Packet Descriptor	✓	0	✓	✓		Used to provide descriptor and data information in one contiguous data structure

Figure 11-903 shows how the various types of descriptors are queued. For Host type descriptors, it illustrates how Host Buffers are linked to a Host Packet, while only the Host Packet is pushed and popped from a queue. Both Host and Monolithic descriptors may be pushed into the same queue, though in practice they are usually kept separate.

Figure 11-903. Packet Queuing Data Structure Diagram



11.13.2.3.6 Packet Transmit Operation

Refer to Section 11.13.2.4.3, *Transmit Operation (Host Packet Type)* and Section 11.13.2.4.4, *Transmit Operation (Monolithic Packet)*.

11.13.2.3.7 Packet Receive Operation

Refer to Section 11.13.2.4.8, *Receive Operation (Host Packet)* and Section 11.13.2.4.9, *Receive Operation (Monolithic Packet)*

11.13.2.3.8 CPPI DMA

The CPPI DMA (or CDMA) used within NAVSS is like most DMAs in that it is primarily concerned with moving data from point to point. It is unlike some DMAs in that it is unaware of the payload data's structure. To the CDMA, all payloads are simple one-dimensional byte streams. Programming the CDMA is accomplished through correct initialization of descriptors, CDMA RX/TX channels, and RX flows.

11.13.2.3.8.1 Channels

Each CDMA in the system is configured with a number of receive (RX) and transmit (TX) channels (seeSection 11.13.5.2.3.4 and Section 11.13.5.2.3.3 for details). A channel may be thought of a pathway through the CDMA. Once the CDMA has started a packet on a channel, that channel cannot be used by any other packet until the current packet is completed. Because there are multiple channels for RX and TX, multiple packets may be in-flight simultaneously, and in both directions, because each CDMA contains separate DMA engines for RX and TX.

11.13.2.3.8.2 RX Flows

For transmit, the TX DMA engine uses the information found in the descriptor fields to determine how to process the TX packet. For receive, the RX DMA uses a *flow*. A flow is a set of instructions that tells the RX DMA how to process the RX packet. It is important to note that there is not a correspondence between RX channel and RX flow, but rather between RX packet and RX flow. For example, one peripheral may create a single RX flow for all packets across all channels, and another may create several flows for the packets on each channel.

For loopback CDMA modes (that is, infrastructure cases), the RX flow is specified in the TX descriptor, in the SOURCE_TAG_LO field. The CDMA will pass this value to the streaming I/F as *flow index*. In non-loopback cases, the RX flow is specified in the packet info structure of the Streaming I/F. In the event no RX flow is specified, the RX DMA will use RX flow *N* for RX channel *N*.

11.13.2.3.9 Port Data Structures

11.13.2.3.9.1 Transmit Channel State

The port stores and maintains state information for each transmit channel. The state information is referred to as the Tx Channel State. The Tx Channel State is a collection of persistent variables which are used by the port state machine during the transmit operation. The contents of the Tx Channel State generally needs to contain all of the required fields in the packet descriptor as well as current data pointers, status flags, etc. The Host is not required to initialize the Tx Channel State. On channel setup, the port is required to clear it's own state to a known initialization value.

11.13.2.3.9.2 Transmit Channel Real Time Control/Status

Each Tx DMA channel in the port requires some control and status bits that are allowed to be modified independent of the current processing state of the channel. Enabling and disabling of a channel are key examples of tasks which inherently occur asynchronously to the packet processing on the channel. The following register contains these Real Time Control and Status fields that must be provided in the port for each Tx DMA Channel:

- CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_0 to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_8

11.13.2.3.9.3 Transmit Channel Configuration

Each Tx DMA channel in the port also requires some control fields to be initialized before a channel is enabled. Unlike the Tx Real Time Channel Control fields, these fields cannot be changed while the channel is enabled or active (including the time period where run-out operations may occur after a channel is disabled). The following register contains these Channel Configuration Fields:

- CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_0 to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_8

11.13.2.3.9.4 Receive Port DMA State

The port stores and maintains state information for each receive channel. The state information is referred to as the Rx DMA State. The Rx DMA State is a combination of control fields and protocol specific port scratchpad space used to manipulate data structures in order to receive packets.

11.13.2.3.9.5 Rx DMA Real-Time Control/Status

Each Tx DMA channel in the port requires some control and status bits that are allowed to be modified independent of the current processing state of the channel. Enabling and disabling of a channel are key examples of tasks which inherently occur asynchronously to the packet processing on the channel. The following register contains these Real Time Control and Status fields that must be provided in the port for each Rx DMA Channel:

- CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_0 to CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_25

11.13.2.3.9.6 Receive Flow Configuration

Unlike the Tx channels, each Rx channel does not necessarily have a single dedicated set of configuration registers that configure of the optional channel parameters. Instead, the port provides an array of *flow configurations* that contain the default values for the channel behaviors. Each Rx packet provided by the application provides both a DMA channel ID and a Flow ID to instruct the DMA controller how to properly process the packet. The DMA channel ID is used to select which parallel processing thread is to be used in the DMA controller. The DMA flow ID is used to select which flow configuration should be used during the packet reception.

In a simple use case, the port DMA controller may choose to provide the same number of flow configuration entries as it provides DMA channels. In more complex cases, more flow configuration entries may be provided to enable complex chaining operations between multiple CPPI ports connected in series. Like the Tx static channel control information, the information in the Rx Flow Configuration entries can only be changed with all of the channels that use that particular flow entry are disabled.

The fields contained within an Rx Flow Configuration Entry are described in the following registers:

- CDMA_RX_FLOW_CONFIG_REG_A_0 to CDMA_RX_FLOW_CONFIG_REG_A_31
- CDMA_RX_FLOW_CONFIG_REG_B_0 to CDMA_RX_FLOW_CONFIG_REG_B_31
- CDMA_RX_FLOW_CONFIG_REG_C_0 to CDMA_RX_FLOW_CONFIG_REG_C_31
- CDMA_RX_FLOW_CONFIG_REG_D_0 to CDMA_RX_FLOW_CONFIG_REG_D_31
- CDMA_RX_FLOW_CONFIG_REG_E_0 to CDMA_RX_FLOW_CONFIG_REG_E_31
- CDMA_RX_FLOW_CONFIG_REG_F_0 to CDMA_RX_FLOW_CONFIG_REG_F_31
- CDMA_RX_FLOW_CONFIG_REG_G_0 to CDMA_RX_FLOW_CONFIG_REG_G_31
- CDMA_RX_FLOW_CONFIG_REG_H_0 to CDMA_RX_FLOW_CONFIG_REG_H_31

11.13.2.4 Operational Description

11.13.2.4.1 Queue Manager Operation

11.13.2.4.1.1 Queuing Descriptors

Descriptors are queued onto a logical queue by writing a burst of information to the corresponding Queue N Registers. This burst contains optional control information, an optional descriptor size, and a required pointer to the descriptor that is being added. The control and packet size information (if present) is written to the QM_QUEUE_REG_C_0–63. The descriptor pointer is written to QM_QUEUE_REG_D_0–63. When QM_QUEUE_REG_D_0–63 is written, this kicks off the queue manager causing it to add the descriptor either onto the head or the tail of the queue as specified in QM_QUEUE_REG_C_0–63.

To accomplish this linking, the queue manager will first resolve the 32-bit descriptor pointer into a 16-bit index which is used for linking and queue pointer purposes. This address to index computation is performed by the queue manager in a fixed number of clock cycles. Once the physical index information is determined, the queue manager will link that descriptor onto the descriptor chain that is maintained for that logical queue by writing the linking information out to a linking RAM which is external to the queue manager. The queue manager will also update the queue head and tail pointers appropriately. Since logical queues within the queue manager are maintained using linked lists, queues cannot become full and no check for fullness is required before queuing a packet descriptor.

11.13.2.4.1.2 De-queuing Descriptors

Descriptors are de-queued from a logical queue by reading a descriptor pointer value from the corresponding QM_QUEUE_REG_D_0–63. When QM_QUEUE_REG_D_0–63 is read, if it is not empty it will return the 32-bit descriptor pointer that is on the head of the queue. If the queue is empty it will return a value of 0x0. If other information is desired it must be read from registers A–C in the same burst in which QM_QUEUE_REG_D_0–63 is read.

In order to populate the mailboxes with a 32-bit descriptor address, the queue manager resolve the 16-bit index into a 32-bit descriptor pointer. This index to address lookup is performed by the queue manager in a fixed number of clock cycles. When the 32-bit pointer is resolved and loaded into the mailbox register, the queue manager will assert the qmgr_slv_ready signal until the burst transfer is completed.

11.13.2.4.1.3 Diverting Queued Packets from One Queue to Another

The host can move the entire contents of one queue to another queue by writing the source queue number and the destination queue number to QM_QUEUE_DIVERSION_REG. When diverting packets, the host can choose whether the source queue contents should be pushed onto the tail (default) or head of the destination queue.

11.13.2.4.2 Transmit DMA Controller Channel Setup (All Packet Types)

After a reset or a previous teardown operation but before queuing packets to a channel the host must initialize the channel's Tx Port DMA State. The host initializes the channel Tx Port DMA State by writing to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_0–8. The Host may choose to write the TX_ENABLE bit at the same time or after it has written all of the channel parameters but note that every write will overwrite the channel state for all bytes that are enabled for the write transaction.

After a Transmit channel has been set up, packets can be added to the Queues for the channel to begin the transmit operation. The following sections describe how the transmit operations are performed for the various descriptor types.

11.13.2.4.3 Transmit Operation (Host Packet Type)

After a TX DMA channel has been initialized, it can begin to be used to transmit packets. Packet transmission involves the following steps:

1. The Host is made aware of one or more chunks of data in memory that need to be transmitted as a packet. This may involve directly sourcing data from the Host or it may involve data which has been forwarded from another data source in the system.

2. The Host allocates and populates a host packet descriptor. The host will initialize the following fields within the packet descriptor:
   a. Descriptor Type (set to Host)
   b. Packet Length indicating the total number of bytes that are to be read from all of the buffers for this packet.
   c. Source Tag
   d. Destination Tag (application specific)
   e. Packet Type
   f. Any protocol specific flags for the given packet type
   g. Buffer Pointer with the byte aligned address of the first chunk of buffer data
   h. Buffer Length with the number of bytes in the first chunk of buffer data
   i. Descriptor reclamation policy fields indicating the mode, queue manager, and queue number for recycling the packet after transmission is complete
   j. The Next Descriptor Pointer with the 32-bit aligned address of the next descriptor in this packet. If this is the last chunk of data in the packet, this field must be set to zero
   k. Any protocol specific descriptor sections that are required for the given packet type or system configuration
3. The Host allocates and populates host buffer descriptors as necessary to point to any remaining chunks of data that belong to this packet. The host will initialize the following fields within the host buffer descriptor:
   a. Buffer Descriptor reclamation policy fields (if the intention is to have the DMA automatically recycle the buffer descriptors as they are transmitted).
   b. Buffer Pointer with the byte aligned address of the given chunk of buffer data
   c. Buffer Length with the number of bytes in the given chunk of buffer data
   d. The Next Descriptor Pointer with the 32-bit aligned address of the next descriptor in this packet. If this is the last chunk of data in the packet, this field must be set to zero
4. The Host writes the pointer to the Packet Descriptor into a specific memory mapped location inside the Queue Manager which corresponds to one of the Transmit Queues for the desired DMA channel. Channels may provide more than one Tx Queue and may provide a particular prioritization policy between the queues. This behavior is application specific and is controlled by the DMA controller/scheduler implementation.
5. The Queue Manager provides a level sensitive status signal for the queue which indicates if any packets are currently pending. This level sensitive status line is sent to the hardware block which is responsible for scheduling DMA operations.
6. The DMA controller is eventually brought into context for the corresponding channel and begins to process the packet.
7. The DMA controller reads the packet descriptor pointer and descriptor size hint information from the Queue Manager.
8. The DMA controller reads the packet descriptor from memory
9.
   - If the return policy bit (PD Word 2, bit 15) is cleared– The DMA controller empties each buffer in sequence by transmitting the contents in one or more block data moves. As each buffer is emptied, the DMA will read the next buffer descriptor to obtain the pointer and size of the data buffer as well as the pointer to the next descriptor in the chain.
   - If the return policy bit is set– The DMA controller empties each buffer in sequence by transmitting the contents in one or more block data moves.
10.
    - If the return policy bit (PD Word 2, bit 15) is cleared– When all data for the packet has been transmitted as specified in the packet size field, the DMA will write the pointer to the packet descriptor to the queue specified in the return queue manager/return queue number fields of the packet descriptor.

- If the return policy bit is set– As each buffer is emptied the DMA will write the buffer descriptor pointer to the queue specified in the return queue manager/return queue number fields of the buffer descriptor. The DMA will also read the next buffer descriptor to obtain the pointer and size of the data buffer, the return queue information for the descriptor, as well as the pointer to the next descriptor in the chain.

11. After the Packet Descriptor pointer has been written, the Queue Manager will indicate the status of the Tx Completion Queues to other ports/processors/prefetcher blocks using out-of-band level sensitive status lines. These status lines are set anytime a queue is non-empty.
12.
    - If the return policy bit (PD Word 2, bit 15) is cleared– While most types of peer entities and embedded processors are able to directly and efficiently use these level sensitive status lines, cached processors may require a hardware block to convert the level status into pulsed interrupts and to perform some level of aggregation of the descriptor pointers from the completion queues into lists.
    - If the return policy bit is set– When all data for the packet has been transmitted as specified in the packet size field, if the current buffer is not marked as end of packet (via a null next descriptor pointer), the DMA will continue to walk the list returning buffer descriptors to the appropriate queues as described in step 10.
13.
    - If the return policy bit (PD Word 2, bit 15) is cleared– Host responds to status change from Queue Manager and performs garbage collection as necessary for packet.
    - If the return policy bit is set– When all buffers have been returned, the DMA will write the pointer to the packet descriptor to the queue specified in the return queue manager/return queue number fields of the packet descriptor.

With the return policy bit (PD Word 2, bit 15) cleared, the complete process is shown in Figure 11-904

With the return policy bit set, the complete process is shown in Figure 11-905

Figure 11-904. Host Packet Tx Operation – Complete Packet Return

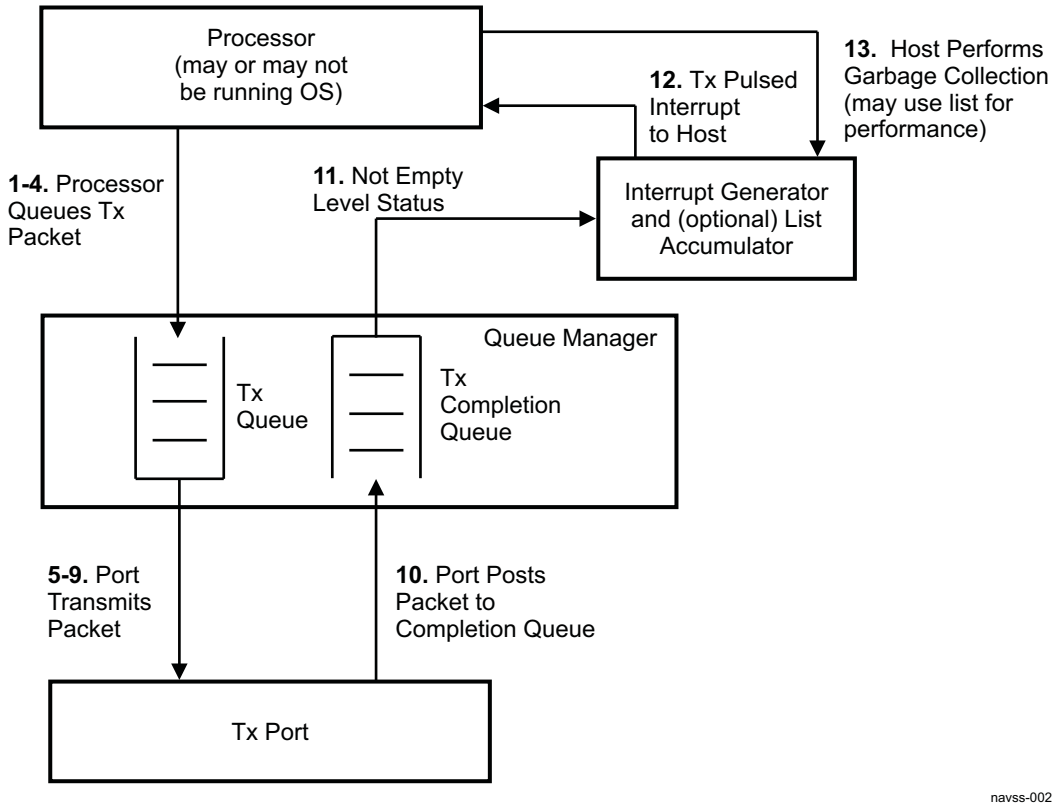
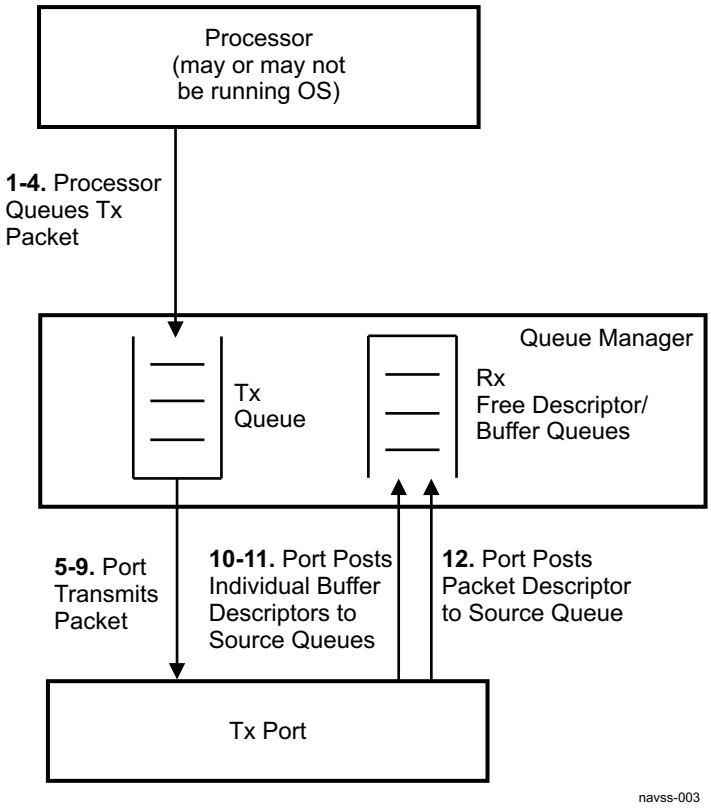


Figure 11-905. Host Packet Tx Operation – Automatic Buffer Recycling Packet Return



11.13.2.4.4 Transmit Operation (Monolithic Packet)

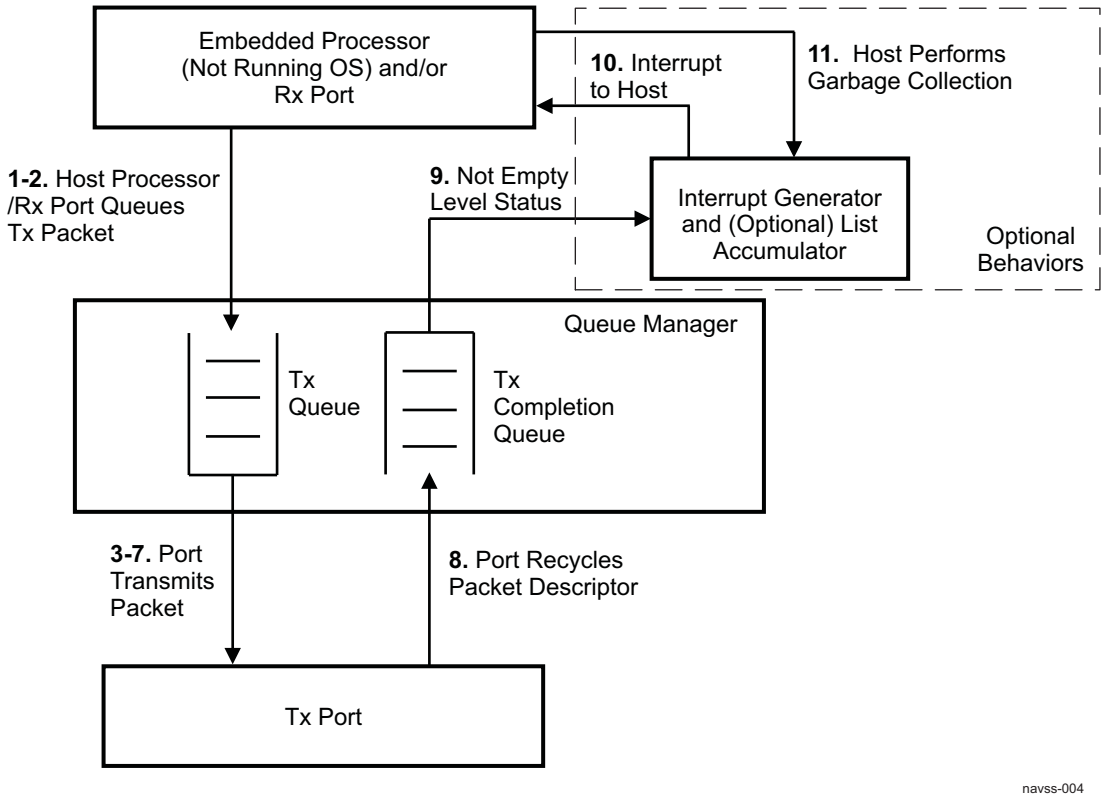
Packet transmission in Monolithic mode involves the following steps:

1. The Host allocates and populates a monolithic packet descriptor. The host will initialize the following fields within the packet descriptor:
   a. Descriptor Type (set to Monolithic)
   b. Packet Length indicating the total number of bytes in the packet
   c. Source Tag
   d. Destination Tag (application specific)
   e. Packet Type
   f. Any protocol specific flags for the given packet type
   g. Offset to payload data
   h. Descriptor reclamation policy fields indicating the mode, queue manager, and queue number for recycling the packet after transmission is complete.
   i. Networking Stack Information (application specific and optional)
   j. Any protocol specific words that are required for the given packet type
2. The Host writes the pointer to the Packet Descriptor into a specific memory mapped location inside the Queue Manager which corresponds to one of the Transmit Queues for the desired DMA channel. Channels may provide more than one Tx Queue and may provide a particular prioritization policy between the queues.
3. The Queue Manager provides a level sensitive status signal for the queue which indicates if any packets are currently pending. This level sensitive status line is sent to the hardware block which is responsible for scheduling DMA operations.
4. The DMA controller is eventually brought into context for the corresponding channel and begins to process the packet.

5. The DMA controller reads the packet descriptor pointer and descriptor size from the Queue Manager.
6. The DMA controller reads the packet descriptor from memory
7. The DMA controller empties the data region in the descriptor by transmitting the contents in one or more block data moves.
8. When all data for the packet has been transmitted as specified in the packet size field, the DMA will write the pointer to the packet descriptor to the queue specified in the return queue manager/return queue number fields of the packet descriptor.
9. After the Packet Descriptor pointer has been written, the Queue Manager indicates the status of the Tx Completion Queues to other ports/processors/prefetcher blocks using out-of-band level sensitive status lines. These status lines are set anytime a queue is non-empty.

   Whether or not any further processing is required on the packet at this point depends on the nature of the queue that the packet was returned to. The most common case is that the monolithic descriptor will be returned to the Rx Free Descriptor queue that it was allocated from originally. If this is the case, the Tx processing is complete at this point. Alternatatively, the queue may also be a Tx Completion Queue in which case, the following optional steps may also be performed:
10. While most types of peer entities and embedded processors are able to directly and efficiently use these level sensitive status lines, cached processors may require a hardware block to convert the level status into pulsed interrupts and to perform some level of aggregation of the descriptor pointers from the completion queues into lists.
11. Host responds to status change from Queue Manager and performs garbage collection as necessary for packet.

Figure 11-906. Monolithic Packet Tx Operation



11.13.2.4.5 Transmit Channel Teardown (All Packet Types)

A channel teardown is commanded by the host to gracefully terminate an active channel from transmitting packets. The host writes the TX_TEARDOWN bit in CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_0–8 in order to initiate channel teardown. Upon reception of a channel teardown command, the port performs the following operations:

1. Gracefully terminates or completes any packet that is in transmission on the channel
2. Clears the channel enable in CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_0–8

The host may issue a teardown command on any channel at any time, regardless of whether the channel has active queue(s) or not. Note that the host should not clear the enable bit when issuing the teardown command. Doing so may prevent the teardown operation from completing since the channel will be removed from transmit scheduling operations.

The Host determines that a teardown is complete by periodically polling the TX_TEARDOWN and TX_ENABLE bits for the channel.

11.13.2.4.6 Receive Channel Setup (All Packet Types)

After a reset or a previous teardown operation but before receiving packets on a channel the host must initialize the channel's Rx Port DMA State. The host initializes the channel Rx Port DMA State by writing to CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_0–25. The Host may choose to write the RX_ENABLE bit at the same time or after it has written all of the channel parameters but note that every write to CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_0–25 will overwrite the channel state for all bytes that are enabled for the write transaction.

11.13.2.4.7 Receive Free Descriptor/Buffer Queue Setup (Host Packets)

The Host is ultimately responsible for providing all of the free descriptors and buffers that the port uses as it receives packets. For Rx Free Descriptor/Buffer queues, descriptors are allocated in a large block by the Host and then initialized and linked with buffers before being given matched pairs where the Host initializes the descriptor to point to its corresponding buffer before being pushed onto a Receive Free Descriptor/Buffer queue. Once the host has allocated both a descriptor and a buffer, it adds them to an Rx Free Descriptor/Buffer Queue by writing the pointer to QM_QUEUE_REG_D_0–63 for the desired queue.

Before the Host adds each Rx buffer descriptor to the queue, it must first initialize the Rx buffer descriptor values as follows:

- Write the Original Buffer Pointer with the byte aligned address of the buffer data
- Write the Original Buffer Size
- Write the Return Queue Number and Return Queue Manager fields in the descriptor to appropriate values (required if auto-recycling is intended)
- Write the On Chip/ Off Chip indicator in the descriptor

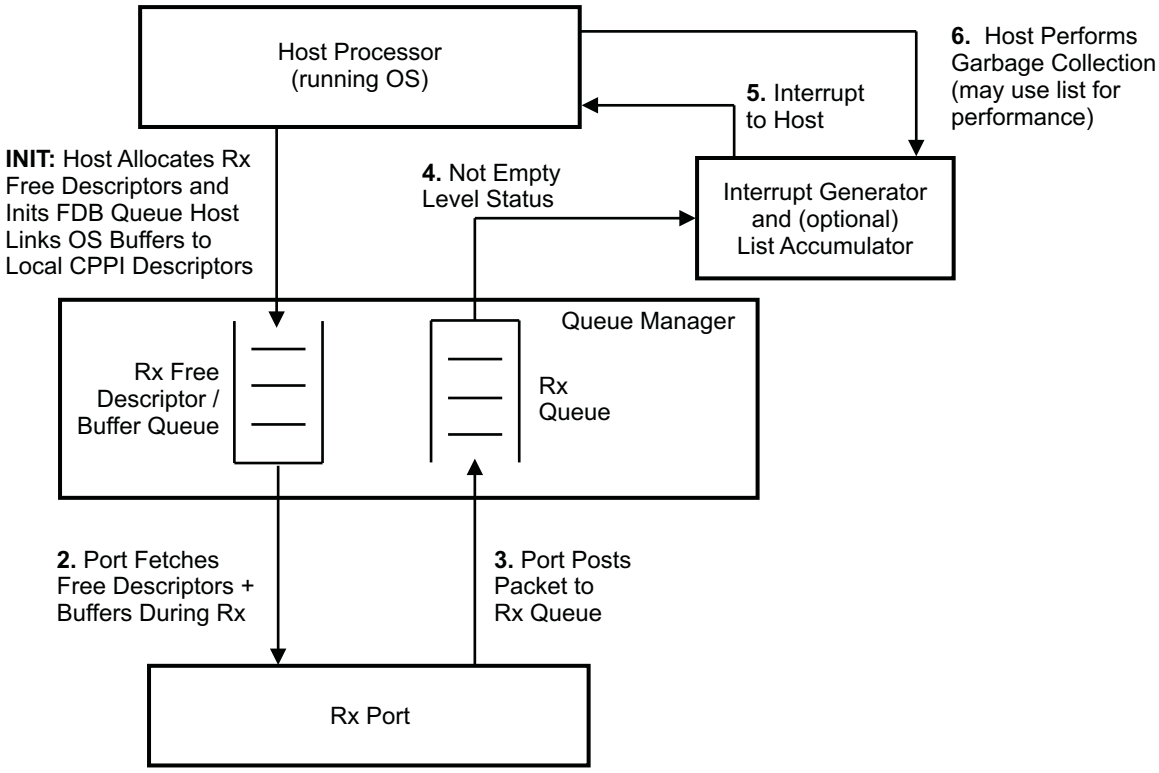
Depending on the software architecture, the Host may also desire to write the return policy bit in the descriptor at this point as well. The DMA will not overwrite any of these listed fields (including return policy) during reception.

All other fields in the Descriptor do not need to be initialized as they will be overwritten by the Port on reception.

11.13.2.4.8 Receive Operation (Host Packet)

Figure 11-907 shows a diagram of the overall receive operation.

Figure 11-907. Packet Receive Operation (Host Packet)



When packet reception begins on a given channel, the port will begin by fetching the first descriptor + buffer from the Queue Manager using the Free Descriptor/Buffer Queue 0 Index that was initialized in the Rx Port DMA State for the channel. If the SOP Buffer Offset in the Rx DMA State is nonzero, then the port will begin writing data after the offset number of bytes in the SOP buffer. The port will then continue filling that buffer and will fetch additional descriptors + buffers as needed using the Free Descriptor/Buffer Queue 1, 2, and 3 indexes for the 2nd, 3rd, and remaining buffers in the packet. The port performs the following operations when each buffer is filled on receive:

1. Fetches next descriptor + buffer (if End of Packet has not been reached)
2. Writes the buffer descriptor to memory. This includes the following fields:
    a. Buffer information
        i. Buffer Pointer with the byte aligned address of the chunk of buffer data
        ii. Buffer Length with the number of bytes in the chunk of buffer data
    b. Pointer to next buffer descriptor in packet or 0 if this is the EOP buffer.

The port performs the following operations when the entire packet has been received:

1. Writes the packet descriptor to memory. This includes the following fields:
    a. Descriptor Type (set to Host)
    b. Packet Length indicating the total number of bytes that are to be read from all of the buffers for this packet.
    c. Source Tag
    d. Destination Tag (application specific)
    e. Packet Type
    f. Any protocol specific flags for the given packet type
    g. SOP buffer information
        i. Buffer Pointer with the byte aligned address of the chunk of buffer data

ii. Buffer Length with the number of bytes in the chunk of buffer data

h. Any protocol specific words that are required for the given packet type

i. Networking Stack Information (application specific and optional)

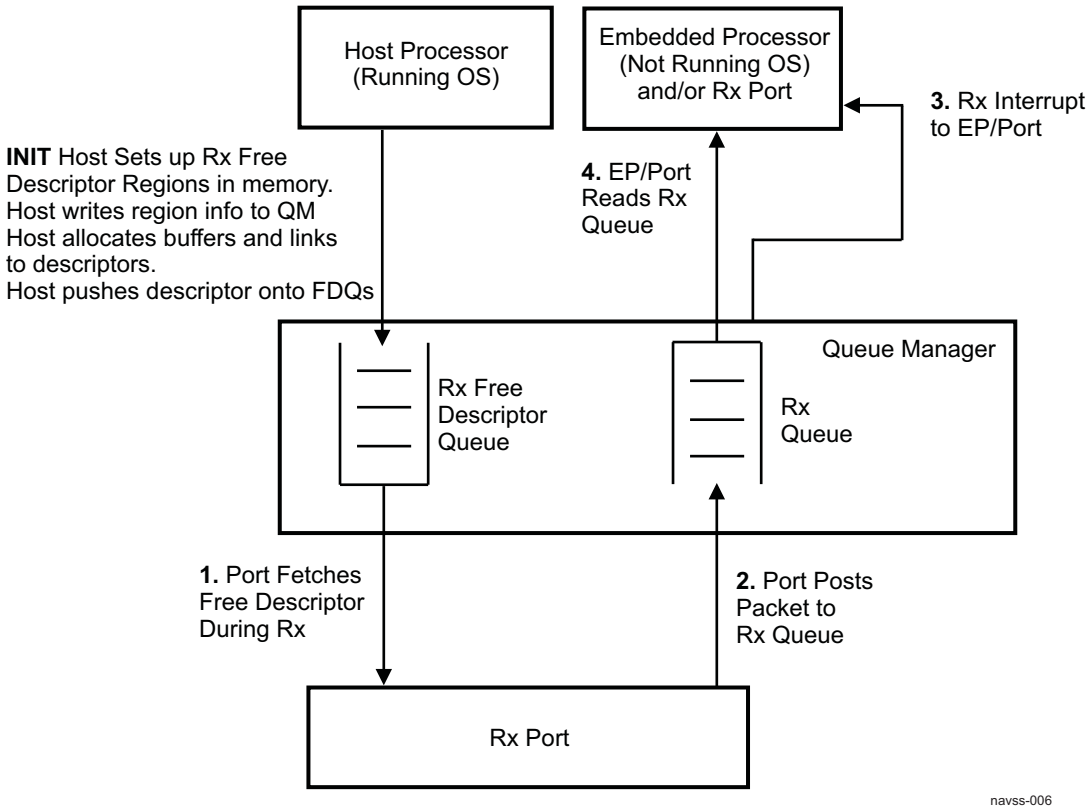
j. Pointer to next buffer descriptor in packet or 0 if this is the EOP buffer.

2. Writes the packet descriptor pointer to the appropriate Rx Queue. The absolute Queue that each packet to be forwarded to on completion of reception will either be the Queue which that was specified in the RX_DEST_QMGR and RX_DEST_QNUM fields in the Rx Flow Entry or can be overridden using an application specific value.

11.13.2.4.9 Receive Operation (Monolithic Packet)

Error! Reference source not found. shows a diagram of the overall Receive operation for a monolithic descriptor type.

Figure 11-908. Monolithic Packet Receive Operation



When packet reception begins on a given channel, the port will begin by fetching a single descriptor for the packet from the Queue Manager using the Monolithic Type Free Descriptor Queue Index that was initialized in the Rx Port DMA State for the channel. If the Monolithic Type Data Offset in the Rx DMA State is non-zero, then the port will begin writing data after the offset number of bytes in the payload portion of the monolithic descriptor. The port will then continue filling the payload portion of the descriptor until the entire packet has been received.

The port performs the following operations when each buffer is filled on receive:

1. Writes the packet descriptor to memory. This includes the following fields:
   a. Descriptor Type (set to Monolithic)
   b. Packet Length indicating the total number of bytes in this packet.
   c. Source Tag
   d. Destination Tag (application specific)

e. Packet Type
f. Any protocol specific flags for the given packet type
g. Networking Stack Information (application specific and optional)
h. Any protocol specific words that are required for the given packet type

2. Writes the packet descriptor pointer to the appropriate Rx Queue. The absolute Queue that each packet to be forwarded to on completion of reception will either be the Queue which that was specified in the RX_DEST_QMGR and RX_DEST_QNUM fields in the Rx Flow Entry or can be overridden using an application specific value.

11.13.2.4.10 Receive Error Processing

Several types of errors/exceptions may be encountered during reception of a host packet. These errors can be broken into these basic types:

- Descriptor Starvation
- Protocol errors
- Dropped packets

11.13.2.4.10.1 Descriptor Starvation

Descriptor starvation occurs when the Port attempts to fetch a free descriptor from an Rx Free Descriptor Queue or Rx Free Descriptor/Buffer Queue and none are available. When no descriptor is available, the Queue Manager will return a value of 0x0 for the descriptor pointer. When the port detects that descriptor starvation has occurred it will react based on the value of the RX_ERROR_HANDLING bit which was programmed into CDMA_RX_FLOW_CONFIG_REG_A_0–31.

- If the RX_ERROR_HANDLING bit is cleared:

The Port will assert a descriptor starvation indicator and will then return any descriptors that it has already used in the reception of the current packet back to the appropriate Free Descriptor or Free Descriptor / Buffer queue that they were fetched from. This is performed by writing the pointers for each descriptor to the Rx Free Descriptor Queue N Register in sequence using the queue indexes in the Rx DMA state. The port may choose to implement a counter that counts the number of packets that were dropped due to descriptor starvation and may also desire to distinguish between whether the starvation occurred on the packet descriptor (SOP) or on a buffer starvation.

- If the RX_ERROR_HANDLING bit is set:

The Port will assert a descriptor starvation indicator (which may cause an interrupt to the Host) and will then behave as if it had not performed the current data block transfer which caused the starvation. The Port is required to pause it's current processing saving the state of the transfer. The next time that the channel comes into context it will again try to allocate a descriptor with the intention being that if buffers/descriptors have been added that the operation will complete successfully on a subsequent retry.

11.13.2.4.10.2 Protocol Errors

specific criteria that was checked during reception of the packet. Examples of protocol errors include packet length errors, CRC errors, or alignment errors. When protocol errors are detected, the port may choose whether or not it will drop the packet. The mechanism that is used to determine which packets to drop and which to forward is application specific. If the port determines that it needs to forward the packet to the Host, it will set the Packet Error bit in the Descriptor indicating that this is a packet which experienced a protocol related error. The type of protocol related error is generally indicated in either the Protocol Specific bits in the descriptor or in the Protocol Specific bytes region. The packet length information and certain portions of the Protocol Specific region may not be correct for packets which encounter errors.

11.13.2.4.10.3 Dropped Packets

Packets can be dropped within an Rx Port for any number for reasons which with the exception of the starvation case which was covered above,are application specific. When a Port determines that it needs to drop a packet, it should return any descriptors to the queues they were allocated from in the appropriate Queue Manager(s). Returning the descriptors to their source queues involves simply performing a write of the appropriate pointer which was allocated from the queue back to the address from which the pointer was originally read.

11.13.2.4.11 Receive Channel Teardown

An Rx channel teardown is commanded by the host to gracefully terminate an active channel from receiving packets. The host initiates a teardown by setting the RX_TEARDOWN bit in the channel's CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_0–25.

Upon reception of a channel teardown command (as indicated by setting the bit in the control register) the port performs the following operations:

1. Completes any packets which may be pending in the ingress port buffers (what the port considers as pending packets is application specific)
2. Clears the channel enable in CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_0–25

The host may issue a teardown command on any channel at any time, regardless of whether the channel is actively receiving a packet or not. Note that the host must not clear the enable bit when issuing the teardown command.

The Host determines that a teardown is complete by periodically polling the TEARDOWN and ENABLE bits for the channel.

11.13.2.5 Descriptor Layouts

Descriptors are memory areas that describe the contents of a packet. This memory may be co-located with the packet data, or may contain pointers to the data.

11.13.2.5.1 Host Packet Descriptor

Host packet descriptors are designed to be used when the application requires support for true, unlimited fragment count scatter/gather-type operations. The host packet descriptor contains the following information:

- Indicator that identifies the descriptor as a host packet descriptor
- Source and destination tags
- Packet type
- Packet length
- Protocol-specific region size
- Protocol-specific control/status bits
- Pointer to the first valid byte in the SOP data buffer
- Length of the SOP data buffer
- Pointer to the next buffer descriptor in the packet
- Software-specific information

Host packet descriptors always contain 32 bytes of required information and may also contain optional software-specific information and protocol-specific information. How much optional information (and therefore the allocated size of the descriptors) is required is application-dependent. The descriptor layout is shown in Table 11-1976.

Table 11-1976. Host Packet Descriptor Layout

Packet info (12 bytes)
Buffer info (8 bytes)
Linking info (4 bytes)
Original buffer info (8 bytes)
Extended packet info block (optional) Includes timestamp and software data (16 bytes)
Protocol-specific data (optional) (0 to *M* bytes where *M* is a multiple of 4)
Other software data (optional and user defined)

Host packet descriptors may be linked with zero or more additional host buffer descriptors in a singly-linked-list fashion to form packets. Each host packet consists of a single host packet descriptor followed by a chain of zero or more host buffer descriptors linked together using the next descriptor pointer fields in the descriptors. The last descriptor in a host packet has a 0 next descriptor pointer.

The other software data portion of the descriptor exists after all of the defined words and is reserved for use by the host software to store completely private data. This region is not used in any way by the DMA or queue manager modules in a NAVSS system and these modules will not modify any bytes within this region.

The contents of the host packet descriptor words are detailed in the following tables:

Table 11-1977. Host Packet Descriptor Packet Information Word 0 (PD Word 0)

Bits	Name	Description	RX Overwrite
31-30	Packet Id	Host packet descriptor type identifier. Value is always 0 (0x0) for Host Packet descriptors.	Yes
29-25	Packet Type	This field indicates the type of this packet and is encoded as follows: 0-31 = To Be Assigned	Yes

Table 11-1977. Host Packet Descriptor Packet Information Word 0 (PD Word 0) (continued)

Bits	Name	Description	RX Overwrite
24-23	Reserved	Unused	Yes
22	Protocol Specific Region Location	This field indicates the location of the protocol-specific words: • 0 = PS words are located in the descriptor • 1 = PS words are located in the SOP Buffer immediately prior to the data.	Yes
21-0	Packet Length	The length of the packet data in bytes. If the packet length is less than the sum of the buffer lengths, then the packet data will be truncated. A packet length greater than the sum of the buffers is an error. The valid range for the packet length is 0 to 4M-1 bytes. If the packet length is set to 0, the port will not actually transmit any information. Instead, the port will perform buffer / descriptor reclamation as instructed in the return information in word 2.	Yes

Table 11-1978. Host Packet Descriptor Packet Information Word 1 (PD Word 1)

Bits	Name	Description	RX Overwrite
31-24	Source Tag - Hi	This field is application-specific. During packet reception, the DMA controller in the port will overwrite this field as specified in the RX_SRC_TAG_HI_SEL field in the flow configuration table entry.	Configurable
23-16	Source Tag - Lo	This field is application-specific. During packet reception, the DMA controller in the port will overwrite this field as specified in the RX_SRC_TAG_LO_SEL field in the flow configuration table entry. For TX, this value supplies the RX flow index to the Streaming I/F for infrastructure use.	Configurable
15-8	Dest Tag – Hi	This field is application specific. During packet reception, the DMA controller in the port will overwrite this field as specified in the RX_DEST_TAG_HI_SEL field in the flow configuration table entry.	Configurable
7-0	Dest Tag - Lo	This field is application specific. During packet reception, the DMA controller in the port will overwrite this field as specified in the RX_DEST_TAG_LO_SEL field in the flow configuration table entry.	Configurable

Table 11-1979. Host Packet Descriptor Packet Information Word 2 (PD Word 2)

Bits	Name	Description	RX Overwrite
31	Extended Packet Info Block Present	This field indicates the presence of the extended packet info block in the descriptor. • 0 = EPIB is not present • 1 = 16 byte EPIB is present	Yes
30	Reserved	Unused	Yes
29-24	Protocol Specific Valid Word Count	This field indicates the valid number of 32-bit words in the protocol-specific region. This is encoded in increments of 4 bytes as follows: • 0 = 0 bytes • 1 = 4 bytes … • 16 = 64 bytes … 32 = 128 bytes 33-63 = Reserved	Yes
23-20	Error Flags	This field contains error flags that can be assigned based on the packet type	Yes
19-16	Protocol Specific Flags	This field contains protocol-specific flags / information that can be assigned based on the packet type.	Yes
15	Return Policy	• This field indicates the return policy for this packet. • 0 = Entire packet (still linked together) should be returned to queue specified in bits 13-0 below. • 1 = Each buffer should be returned to queue specified in bits 13-0 of Word 2 in their respective descriptors. The TX DMA will return each buffer in sequence.	No

Table 11-1979. Host Packet Descriptor Packet Information Word 2 (PD Word 2) (continued)

Bits	Name	Description	RX Overwrite
14	Return Push Policy	This field indicates how a transmit DMA should return the descriptor pointers to the free queues. This field is encoded as follows: • 0 = Descriptor must be returned to tail of queue • 1 = Descriptor must be returned to head of queue This bit is used only when the Return Policy bit is set to 1.	No
13-12	Packet Return Queue Mgr Num	This field indicates which of the four potential queue managers in the system the descriptor is to be returned to after transmission is complete. This field is not altered by the DMA during transmission or reception and should be initialized by the host.	No
11-0	Packet Return Queue Num	This field indicates the queue number within the selected queue manager that the descriptor is to be returned to after transmission is complete. The value 0xFFF is reserved.	No

Table 11-1980. Host Packet Descriptor Buffer 0 Info Word 0 (PD Word 3)

Bits	Name	Description	RX Overwrite
31-22	Reserved	Unused	Yes
21-0	Buffer 0 Length	The buffer length field indicates how many valid data bytes are in the buffer. Unused or protocol-specific bytes at the beginning of the buffer are not counted in the buffer length field. This value will be overwritten during reception.	Yes

Table 11-1981. Host Packet Descriptor Buffer 0 Info Word 1 (PD Word 4)

Bits	Name	Description	RX Overwrite
31-0	Buffer 0 Pointer	The buffer pointer is the byte-aligned memory address of the buffer associated with the buffer descriptor. This value will be written during reception. If the protocol-specific words are placed at the beginning of the SOP buffer, this pointer will point to the PS words. The offset to the data in that case must be calculated by the consumer using the protocol-specific valid word count from word 2. **Usage note**: For TX, it is a good practice to initialize this field and the Original Ptr field in word 7 with the actual buffer address, but this is the field that is used. For RX, this field may be left uninitialized, or set to 0.	Yes

Table 11-1982. Host Packet Descriptor Linking Word (PD Word 5)

Bits	Name	Description	RX Overwrite
31-0	Next Descriptor Pointer	The 32-bit word-aligned memory address of the next buffer descriptor in the packet. If the value of this pointer is 0, then the current buffer is the last buffer in the packet. The host sets the next descriptor pointer.	Yes

Table 11-1983. Host Packet Descriptor Original Buffer Info Word 0 (PD Word 6)

Bits	Name	Description	RX Overwrite
31-28	Original Buffer 0 Pool Index	This field is used to identify which pool the attached buffer was originally allocated from. This is distinct from the descriptor pool/queue index because a single buffer may be referenced by more that one descriptor. This is a software-only field that is not touched by the hardware.	No
27-22	Original Buffer 0 Reference Count	This field is used to indicate how many references have been made to the attached buffer by different descriptors. Multiple buffer references are commonly used to implement broadcast and multicast packet forwarding when zero packet data copies are desired. This is a software-only field that is not touched by the hardware.	No
21-0	Original Buffer 0 Length	The buffer length field indicates the original size of the buffer in bytes. Data bytes are in the buffer. This value will not be overwritten during reception. This value is read by the RX DMA to determine the actual buffer size as allocated by the host at initialization. Because the buffer length in Word 3 is overwritten by the RX port during reception, this field is necessary to permanently store the buffer size information. **Usage Note:** It is good practice to always set this field during initialization.	No

Table 11-1984. Host Packet Descriptor Original Buffer Info Word 1 (PD Word 7)

Bits	Name	Description	RX Overwrite
31-0	Original Buffer 0 Pointer	The buffer pointer is the byte-aligned memory address of the buffer associated with the buffer descriptor. This value will not be overwritten during reception. This value is read by the RX DMA to determine the actual buffer location as allocated by the host at initialization. Because the buffer pointer in word 4 is overwritten by the RX port during reception, this field is necessary to permanently store the buffer pointer information. **Usage Note**: It is good practice to always set this field during initialization, but is used only in RX.	No

Table 11-1985. Host Packet Descriptor Extended Packet Info Block Word 0 (Optional) (1)

Bits	Name	Description	RX Overwrite
31-0	Timestamp Info	This field contains an application-specific timestamp that can be used for traffic shaping in a QoS enabled system.	Configurable

(1) This word is present only if the extended packet info block present bit is set in word 2.

Table 11-1986. Host Packet Descriptor Extended Packet Info Block Word 1 (Optional) (1)

Bits	Name	Description	RX Overwrite
31-0	Software Info 0	This field stores software-centric information that needs to travel with the packet through the stack. This information will be copied from the source descriptor to the destination descriptor whenever a prefetch operation is performed or when transferring through an infrastructure DMA node.	Configurable

(1) This word is present only if the Extended Packet Info Block present bit is set in Word 2.

Table 11-1987. Host Packet Descriptor Extended Packet Info Block Word 2 (Optional) (1)

Bits	Name	Description	RX Overwrite
31-0	Software Info 1	This field stores software centric information that needs to travel with the packet through the stack. This information will be copied from the source descriptor to the destination descriptor whenever a prefetch operation is performed or when transferring through an infrastructure DMA node.	Configurable

(1) This word is present only if the Extended Packet Info Block present bit is set in Word 2.

Table 11-1988. Host Packet Descriptor Extended Packet Info Block Word 3 (Optional) [1]

Bits	Name	Description	RX Overwrite
31-0	Software Info 2	This field stores software centric information that needs to travel with the packet through the stack. This information will be copied from the source descriptor to the destination descriptor whenever a prefetch operation is performed or when transferring through an infrastructure DMA node.	Configurable

(1) This word is present only if the Extended Packet Info Block present bit is set in Word 2.

Table 11-1989. Host Packet Descriptor Protocol Specific Word N (Optional)

Bits	Name	Description	RX Overwrite
31-0	Protocol Specific Data N	This field stores information that varies depending on the block and packet type.	Configurable

11.13.2.5.2 Host Buffer Descriptor

The host buffer descriptor is identical in size and organization to a host packet descriptor but does not include valid information in the packet level fields and does not include a populated region for protocol-specific information. Host buffer descriptors are designed to be linked onto a host packet descriptor or another host buffer descriptor to provide support for unlimited scatter / gather type operations. Host buffer descriptors provide information about a single corresponding data buffer. Every host buffer descriptor stores the following information:

- Pointer to the first valid byte in the data buffer
- Length of the data buffer
- Pointer to the next buffer descriptor in the packet

Host buffer descriptors always contain 32 bytes of required information. Because it is a requirement that it is possible to convert a host descriptor between a buffer descriptor and a packet descriptor (by filling in the appropriate fields), in practice, host buffer descriptors will be allocated using the same sizes as host packet descriptors. The descriptor layout is shown in Table 11-1990.

Table 11-1990. Host Buffer Descriptor Layout

Reserved (10 bytes)
Buffer reclamation info (2 bytes)
Buffer info (8 bytes)
Linking info (4 bytes)
Original buffer info (8 bytes)

A host packet descriptor and zero or more host buffer descriptors may be linked together using the next descriptor pointer fields to form packets. The last descriptor in a packet has a 0 next descriptor pointer. Each host buffer descriptor also points to a single data buffer.

The contents of the host buffer descriptor words are detailed in Table 11-1991 through Table 11-1998. The host buffer descriptor is designed to be interchangeable with host packet descriptors, with common fields residing in the same locations.

Table 11-1991. Host Buffer Descriptor Reserved Word 0 (BD Word 0)

Bits	Name	Description	RX Overwrite
31-0	Reserved	Reserved for host packet fields	No

Table 11-1992. Host Buffer Descriptor Reserved Word 1 (BD Word 1)

Bits	Name	Description	RX Overwrite
31-0	Reserved	Reserved for host packet fields	No

Table 11-1993. Host Buffer Descriptor Buffer Reclamation Info (BD Word 2)

Bits	Name	Description	RX Overwrite
31-15	Reserved	Reserved for host packet fields	No
14	Return Push Policy	This field indicates how a transmit DMA should return the descriptor pointers to the free queues. This field is encoded as follows: • 0 = Descriptor must be returned to tail of queue • 1 = Descriptor must be returned to head of queue This bit is used only when the Return Policy bit is set to 1.	No
13-12	Packet Return Queue Mgr #	This field indicates which of the four potential queue managers in the system the descriptor is to be returned to after transmission is complete. This field is not altered by the DMA during transmission or reception and should be initialized by the host.	No
11-0	Packet Return Queue #	This field indicates the queue number within the selected queue manager that the descriptor is to be returned to after transmission is complete.	No

Table 11-1994. Host Buffer Descriptor Buffer N Info Word 0 (BD Word 3)

Bits	Name	Description	RX Overwrite
31-22	Reserved	Reserved for host packet fields	Yes
21-0	Buffer N Length	The buffer length field indicates how many valid data bytes are in the buffer. Unused or protocol-specific bytes at the beginning of the buffer are not counted in the buffer length field. This value will be overwritten during reception.	Yes

Table 11-1995. Host Buffer Descriptor Buffer N Info Word 1 (BD Word 4)

Bits	Name	Description	RX Overwrite
31-0	Buffer N Pointer	The buffer pointer is the byte-aligned memory address of the buffer associated with the buffer descriptor. This value will not be overwritten during reception.	Yes

Table 11-1996. Host Buffer Descriptor Linking Word (BD Word 5)

Bits	Name	Description	RX Overwrite
31-0	Next Descriptor Pointer	The 32-bit word aligned memory address of the next buffer descriptor in the packet. This is the mechanism used to reference the next buffer descriptor from the current buffer descriptor. If the value of this pointer is 0, then the current buffer is the last buffer in the packet. This value will be overwritten during reception.	Yes

Table 11-1997. Host Buffer Descriptor Original Buffer Info Word 0 (BD Word 6)

Bits	Name	Description	RX Overwrite
31-28	Original Buffer 0 Pool Index	This field is used to identify which pool the attached buffer was originally allocated from. This is distinct from the descriptor pool/queue index because a single buffer may be referenced by more that one descriptor. This is a software-only field that is not touched by the hardware.	No
27-22	Original Buffer 0 Reference Count	This field is used to indicate how many references have been made to the attached buffer by different descriptors. Multiple buffer references are commonly used to implement broadcast and multicast packet forwarding when zero packet data copies are desired. This is a software-only field that is not touched by the hardware.	No

Table 11-1997. Host Buffer Descriptor Original Buffer Info Word 0 (BD Word 6) (continued)

Bits	Name	Description	RX Overwrite
21-0	Original Buffer 0 Length	The buffer length field indicates the original size of the buffer in bytes. Data bytes are in the buffer. This value will not be overwritten during reception. This value is read by the RX DMA to determine the actual buffer size as allocated by the host at initialization. Because the buffer length in word 3 is overwritten by the RX port during reception, this field is necessary to permanently store the buffer size information.	No

Table 11-1998. Host Buffer Descriptor Original Buffer Info Word 1 (BD Word 7)

Bits	Name	Description	RX Overwrite
31-0	Original Buffer 0 Pointer	The buffer pointer is the byte-aligned memory address of the buffer associated with the buffer descriptor. This value will not be overwritten during reception. This value is read by the RX DMA to determine the actual buffer location as allocated by the host at initialization. Because the buffer pointer in word 4 is overwritten by the RX port during reception, this field is necessary to permanently store the buffer pointer information.	No

11.13.2.5.3 Monolithic Descriptor

The monolithic packet descriptor contains the following information:

- Indicator that identifies the descriptor as a monolithic packet descriptor
- Source and destination tags
- Packet type
- Packet length
- Packet error indicator
- Packet return information
- Protocol-specific region size
- Protocol-specific region offset
- Protocol-specific control / status bits
- Packet data

The maximum size of a monolithic packet descriptor is 65535 bytes. Of this, monolithic packet descriptors always contain 12 bytes of required information and may also contain 16 bytes of software-specific tagging information and up to 128 bytes (indicated in 4-byte increments) of protocol-specific information. How much protocol-specific information (and therefore the allocated size of the descriptors) is application dependent. The descriptor layout is shown in Table 11-1999.

Table 11-1999. Monolithic Packet Descriptor Layout

Packet info (12 bytes)
Extended packet info block (optional) Includes PS bits, timestamp, and software data words (20 bytes)
Protocol-Specific data (optional) (0 to *M* bytes where *M* is a multiple of 4)
Null region (0 to (511-12) bytes)
Packet data (0 to 64K – 1)
Other software data (optional and user defined)

The other software data portion of the descriptor exists after all of the defined words and is reserved for use by the host software to store private data. This region is not used in any way by the DMA or queue manager modules in a NAVSS system and these modules will not modify any bytes within this region.

A note on the placement of data with respect to the optional EPIB block: If EPIB is present, the 16 bytes of EPIB data begins at byte offset 16, and PS or packet data may begin at byte offset 32 (from the descriptor address). If EPIB is not present, PS or packet data may begin at byte offset 12 (from the descriptor address).

The contents of the monolithic packet descriptor words are detailed in the following tables:

Table 11-2000. Monolithic Packet Descriptor Word 0

Bits	Name	Description	RX Overwrite
31-30	Packet Id	Monolithic packet descriptor type. Value is always 2 (0x02) for monolithic descriptors.	Yes
29-25	Packet Type	This field indicates the type of this packet and is encoded as follows: • 0-31 = To Be Assigned	Yes
24-16	Data Offset	This field indicates the byte offset from byte 0 of this descriptor to the location where the valid data begins. On RX, this value is set equal to the value for the SOP offset given in the RX DMA channel's monolithic control register. When a monolithic packet is processed, this value may be modified in order to add or remove bytes to or from the beginning of the packet. The value for this field can range from 0-511 bytes, which means that the maximum NULL region can be 511-12 bytes, because byte 0 is the start of the 12 byte packet info area. Note that the value of this field must always be greater than or equal to 4 times the value given in the Protocol Specific Valid Word Count field.	Yes
15-0	Packet Length	The length of the packet data in bytes. The valid range is from 0 to 65535 bytes. NOTE: The sum of the data offset field and the packet length must not exceed 64KB or the defined size of the descriptor. To do so is an error, and may cause transmission problems through the Streaming Interface.	Yes

Table 11-2001. Monolithic Packet Descriptor Word 1

Bits	Name	Description	RX Overwrite
31-24	Source Tag - Hi	This field is application-specific. During packet reception, the DMA controller in the port will overwrite this field as specified in the RX_SRC_TAG_HI_SEL field in the flow configuration table entry.	Configurable
23-16	Source Tag - Lo	This field is application-specific. During packet reception, the DMA controller in the port will overwrite this field as specified in the RX_SRC_TAG_LO_SEL field in the flow configuration table entry. For TX, this value supplies the RX flow index to the streaming I/F for infrastructure use.	Configurable
15-8	Dest Tag – Hi	This field is application-specific. During packet reception, the DMA controller in the port will overwrite this field as specified in the RX_DEST_TAG_HI_SEL field in the flow configuration table entry.	Configurable
7-0	Dest Tag - Lo	This field is application-specific. During packet reception, the DMA controller in the port will overwrite this field as specified in the RX_DEST_TAG_LO_SEL field in the flow configuration table entry.	Configurable

Table 11-2002. Monolithic Packet Descriptor Word 2

Bits	Name	Description	RX Overwrite
31	Extended Packet Info Block Present	This field indicates the presence of the extended packet info block in the descriptor. • 0 = EPIB is not present • 1 = 16 byte EPIB is present	Yes
30	Reserved	Unused	Yes

Table 11-2002. Monolithic Packet Descriptor Word 2 (continued)

Bits	Name	Description	RX Overwrite
29-24	Protocol Specific Valid Word Count	This field indicates the valid number of 32-bit words in the protocol-specific region. This is encoded in increments of 4 bytes as follows: • 0 = 0 bytes • 1 = 4 bytes • … • 16 = 64 bytes • … • 32 = 128 bytes • 33-63 = Reserved	Yes
23-20	Error Flags	This field contains error flags that can be assigned based on the packet type.	Yes
19-16	Protocol Specific Flags	This field contains protocol-specific flags / information that can be assigned based on the packet type.	Yes
15	Reserved	Unused	No
14	Return Push Policy	This field indicates how a transmit DMA should return the descriptor pointers to the free queues. This field is encoded as follows: • 0 = Descriptor must be returned to tail of queue • 1 = Descriptor must be returned to head of queue	No
13-12	Packet Return Queue Mgr #	This field indicates which of the four potential queue managers in the system the descriptor is to be returned to after transmission is complete. This field is not altered by the DMA during transmission or reception and should be initialized by the host.	No
11-0	Packet Return Queue #	This field indicates the queue number within the selected queue manager that the descriptor is to be returned to after transmission is complete.	No

Table 11-2003. Monolithic Extended Packet NULL Word (Optional) (1)

Bits	Name	Description	RX Overwrite
31-0	Reserved	This field is present only to align the extended packet words to a 128-bit boundary in memory. This word can be used for host software scratchpad because it will not be copied or overwritten by the DMA components.	No

(1) This word is present only if the Extended Packet Info Block present bit is set in word 2.

Table 11-2004. Monolithic Extended Packet Info Word 0 (Optional) (1)

Bits	Name	Description	RX Overwrite
31-0	Timestamp Info	This field contains an application-specific timestamp that can be used for traffic shaping in a QoS-enabled system.	Configurable

(1) This word is present only if the Extended Packet Info Block present bit is set in word 2.

Table 11-2005. Monolithic Extended Packet Info Word 1 (Optional) (1)

Bits	Name	Description	RX Overwrite
31-0	Software Info 0	This field stores software-centric information that needs to travel with the packet through the stack. This information will be copied from the source descriptor to the destination descriptor whenever a prefetch operation is performed or when transferring through an infrastructure DMA node.	Configurable

(1) This word is present only if the Extended Packet Info Block present bit is set in word 2.

Table 11-2006. Monolithic Extended Packet Info Word 2 (Optional) (1)

Bits	Name	Description	RX Overwrite
31-0	Software Info 1	This field stores software-centric information that needs to travel with the packet through the stack. This information will be copied from the source descriptor to the destination descriptor whenever a prefetch operation is performed or when transferring through an infrastructure DMA node.	Configurable

(1) This word is present only if the Extended Packet Info Block present bit is set in word 2.

Table 11-2007. Monolithic Extended Packet Info Word 3 (Optional) (1)

Bits	Name	Description	RX Overwrite
31-0	Software Info 2	This field stores software-centric information that needs to travel with the packet through the stack. This information will be copied from the source descriptor to the destination descriptor whenever a prefetch operation is performed or when transferring through an infrastructure DMA node.	Configurable

(1) This word is present only if the Extended Packet Info Block present bit is set in word 2.

Table 11-2008. Monolithic Packet Descriptor Protocol Specific Word M (Optional) (1)

Bits	Name	Description	RX Overwrite
31-0	Protocol Specific Data N	This field stores information that varies depending on the packet type.	Configurable

(1) These words, if present, immediately follow the software data block information.

Table 11-2009. Monolithic Packet Descriptor Payload Data Words 0-N (1) (2)

Bits	Name	Description	RX Overwrite
31-0	Packet Data N	These words store the packet payload data.	Yes

(1) The payload data follows the protocol-specific words at an offset specified in the data offset field of word 0.
(2) This field is endian-specific. In other words, this is the only field in the descriptor that changes based on the endianess of the system.

11.13.3 Memory Error Detection and Correction

The CPSW_2U error detection and correction logic uses the ECC Aggregator Module (Section 11.14.4.19.1).

The ECC_VECTOR register is used to select which ECC RAM's status and control registers are currently being read or written as shown in Table 11-2010. The ECC_ENABLE bit in the ECC_CONTROL register is not used (ECC is always enabled). The CPSW_2U FIFO RAMs implement ECC only on packet headers. The packet data is protected by Castignoli (regardless of the input packet CRC type or output CRC type).

Table 11-2010. ECC RAM to EMAC RAM Mapping

ECC RAM Number	EMAC RAM
0	Reserved
1	FIFO RAM

11.13.3.1 Packet Header ECC

Only packet headers bits are protected by ECC in the FIFO RAMs. The ECC_ERROR_CONTROL1 register ECC_ROW is not implemented – ECC_BIT1 is implemented to determine which bit of the header is flipped for an SEC error when the ECC_CRC_MODE bit is cleared in the CPSW_CONTROL register. The ECC status registers return the RAM address that was flipped (ECC_ROW) along with the ECC_BIT1 value. Forcing double bit errors in testing can cause indeterminate operation if multiple used packet header bits are flipped given that only single bit errors are fixed by the ECC logic. Header bits 207 down to 200 are not currently used in the CPSW_2U and may be used to test double bit errors without the possibility of requiring a reset for the switch to recover from the double bit error. No header bits are flipped when ECC_CRC_MODE is set to one.

The header ECC code is stored in bits 255 down to 208. If any bit is flipped in the ECC code, the flipped bit will be corrected, but the index of the flipped bit will be reported as bit zero. This implies that when the aggregator reports that there is a SEC on bit 0, it can mean two things: either SEC on data bit 0 or SEC somewhere inside the ECC code.

11.13.3.2 Packet Protect CRC

Each packet received without error is passed through the CPSW_2U memories with a generated Castignoli protect CRC. The protect CRC is checked on egress for correctness and removed. If the CRC is correct (no RAM bit errors), then the packet is output with the selected port CRC type. If a protect CRC error is detected on host egress then the TXST_DROP signal will be asserted so that the packet is dropped to the host. If a protect CRC error is detected on Ethernet egress then the egress CRC will be generated on the packet and at least one byte of the CRC will be inverted on output. CRC memory protect errors do not assert the ECC_INTR signal. CRC memory protect errors are counted in the associated port statistics registers and issue an interrupt on STAT_PEND if any CRC memory protect error occurs (and the statistics for that port are enabled). When the ECC_CRC_MODE bit in the CPSW_CONTROL register is set, the ECC_BIT1 aggregator bits will flip the associated column bit in any FIFO memory read operation, inducing a CRC protect error when the protect CRC is checked. No header bits are flipped when ECC_CRC_MODE is set.

11.13.3.3 Aggregator RAM Control

The ECC logic for each FIFO RAM (receive and transmit) is divided into eight separate ECC encoders/decoders that encode/decode 26-bits of data each. Each of the 8 encoders (0 to 7) generates 6-bits of ECC code (48 code bits total), and each of the eight decoders (0 to 7) checks 6-bits of ECC code across the 26-bits of data (208 data bits total). The 48-bits of ECC code are passed through the RAM in the upper 48 unused bits in the header word. The header data bits and ECC code bits are shown in Table 11-2011. The ECC_BIT1[15:0] value returned on error is a 16-bit value that is the concatenation of 5 bits of zero, 3 bits of the encoder/decoder number (0 to 7), 3 bits of zero, and 5 bits of index into the indicated 26-bit encoder/decoder.

For example, an ECC_BIT1 value of 0x0308 is bit 8 of encoder/decoder 3, which is header bit 86 (that is, (26×3) + 8).

Table 11-2011. ECC Submodule Header Data Bit to Encoder/Decoder Mapping

Header Data Bits	Encoder/Decoder
25:0	Encoder/Decoder 0 Data
51:26	Encoder/Decoder 1 Data
77:52	Encoder/Decoder 2 Data
103:52	Encoder/Decoder 3 Data
129:104	Encoder/Decoder 4 Data
155:130	Encoder/Decoder 5 Data
181:156	Encoder/Decoder 6 Data
207:182	Encoder/Decoder 7 Data
213:208	Encoder/Decoder 0 ECC
219:214	Encoder/Decoder 1 ECC
225:220	Encoder/Decoder 2 ECC
231:226	Encoder/Decoder 3 ECC
237:232	Encoder/Decoder 4 ECC
243:238	Encoder/Decoder 5 ECC
249:244	Encoder/Decoder 6 ECC
255:250	Encoder/Decoder 7 ECC

11.13.3.4 ECC Wrapper and ECC Aggregator

The ECC wrapper implements SECDED code for single bit error detection and correction and double bit error detection. ECC Aggregator module gathers level pending status from the ECC RAMs into a single interrupt to the host. Software interface is provided for status and control.

There are three types of ECC registers:

1. *Global registers* that are common to all ECC RAMs. This includes the ECC_REVISION, ECC_VECTOR, and ECC_MISC_STATUS registers. Every ECC RAM serviced by the aggregator module is assigned a unique ID by the module. Writing this ID to the ECC_VECTOR register selects the ECC RAM to be controlled or read the ECC status from.
2. *ECC control and status registers.* These are specific to each ECC RAM and selected by the RAM_ID written to the ECC_VECTOR register. The accesses to these registers are done via the serial VBUSP protocol. Note that due to the long latencies on the serial VBUS, the reads to the ECC control and status registers are triggered by a write to the ECC_VECTOR register. This is described in detail in Section 11.14.4.19.1.1.
3. *Interrupt registers.* These include the standard interrupt status, interrupt enable, interrupt clear and EOI registers that are part of a standard interrupt module.

   ECC registers are described in Section 11.13.5.1.6, *NSS_0_CFG_ECC Registers*.

11.13.3.4.1 Reads to ECC Control and Status Registers

Due to the long latencies on the serial VBUS, the reads to the ECC control and status registers for each ECC RAM are triggered by writing a *read* message to the ECC_VECTOR register as described below:

1. Software writes the RAM_ID in ECC_VECTOR[10-0], sets bit [15] TRIGGER_READ, and writes [23-16] READ_ADDRESS. The READ_ADDRESS corresponds to any of the ECC control or status registers (at offset 0x10, that is, ECC_WRAPPER_REVISION through 0x24, that is, ECC_ERROR_STATUS2) which require serial VBUS access to the ECC RAM(s).
2. Software then polls the ECC_VECTOR[24] READ_DONE bit waiting for setting to 1. This indicates that the read operation has completed on the serial VBUS.

3. Software reads the data from the ECC control or status register. Read data is returned on the following clock cycle.

11.13.3.4.2 ECC Interrupts

The ECC aggregator module aggregates all the level pending status from the ECC RAMs to a single EOI-handshake based interrupt to the host. Software is expected to follow the sequence described below:

1. Software enables the interrupts for the ECC RAM(s) by writing to the Interrupt Enable register (ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15).
2. On receiving an interrupt, software checks the Interrupt Status Register (ECC_INT_STATUS_0 to ECC_INT_STATUS_15) to see which ECC RAM(s) caused the error. Note that this status read is not a serial VBUS operation.
3. Software writes the RAM_ID in the ECC_VECTOR register along with the read message that includes setting bit [15] TRIGGER_READ, writing the ECC_ERROR_STATUS1 register's address (that is, 0x20) to bits [23-16] READ_ADDRESS. Software needs to load the read message in the ECC_VECTOR register again if it needs to read also the ECC_ERROR_STATUS2.
4. Software polls the ECC_VECTOR[24] READ_DONE bit. When this is set to 1, it can read the ECC_ERROR_STATUS1 and ECC_ERROR_STATUS2 registers.
5. After the interrupt has been serviced, software must clear the interrupt status by writing to bits 8, 9, or 10 depending on the type of ECC error in the ECC_ERROR_STATUS1 register.
6. Software must poll the ECC_ERROR_STATUS1 register to guarantee that the status bit has been cleared. Otherwise there is no indication that the write has actually completed over the serial VBUS
7. Software must write to the ECC_EOI register to clear the aggregated interrupt output.

11.13.4 Gigabit Ethernet MAC (EMAC) Subsystem

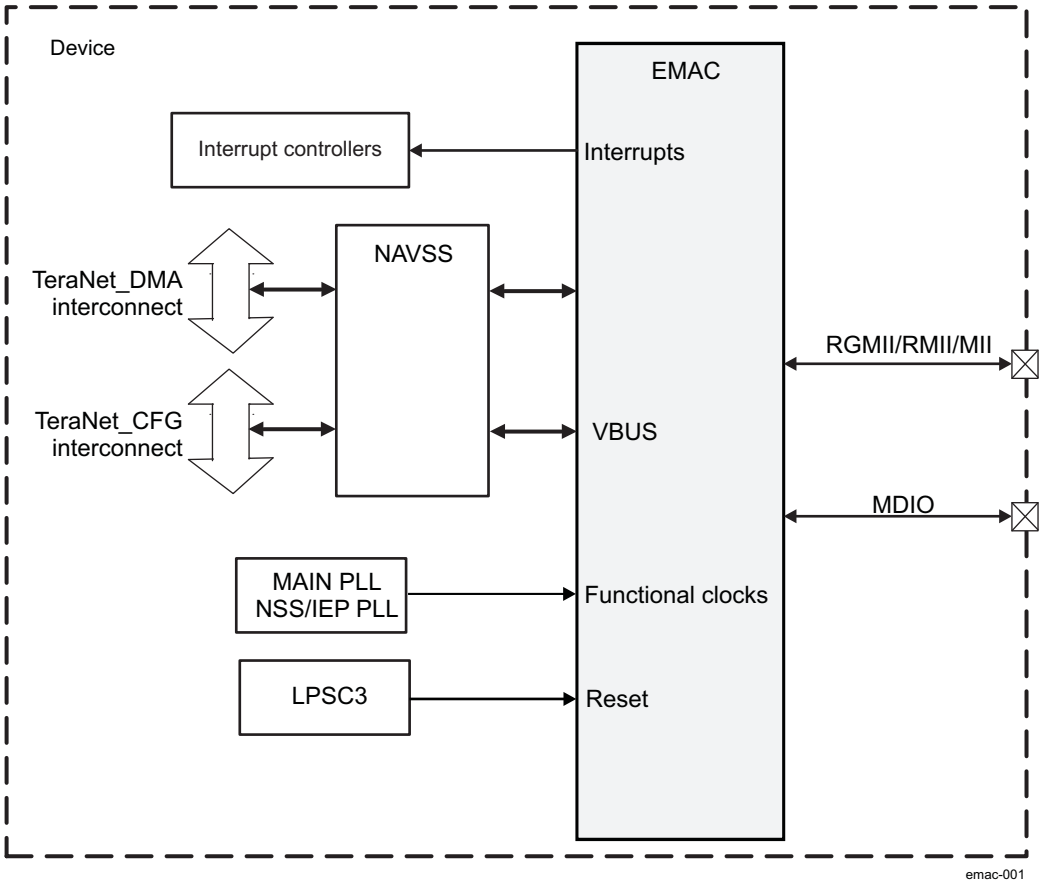
This section describes the gigabit ethernet subsystem in the device.

11.13.4.1 EMAC Overview

The three-port Gigabit Ethernet MAC subsystem (EMAC) provides ethernet packet communication for the device. EMAC features the Gigabit Media Independent Interface (G/MII), Reduced Gigabit Media Independent Interface (RGMII), Reduced Media Independent Interface (RMII), and the Management Data Input/Output (MDIO) interface for physical layer device (PHY) management.

Figure 11-909 shows the EMAC subsystem overview.

Figure 11-909. EMAC Overview



11.13.4.1.1 Features

The EMAC subsystem provides the following features:

- One Ethernet port (port 1) with selectable RGMII, RMII, and G/MII interfaces and an internal (CPPI) port (port 0)
- Synchronous 10/100/1000 Mbit operation
- Flexible logical FIFO-based packet buffer structure
- Eight priority level Quality Of Service (QOS) support (802.1p)
- Support for Audio/Video Bridging (P802.1Qav/D6.0)
- Support for IEEE 1588 Clock Synchronization (2008 Annex D, Annex E and Annex F)
  - Timestamp module capable of time stamping external timesync events like Pulse-Per-Second and also generating Pulse-Per-Second outputs
  - CPTS module that supports time stamping for IEEE1588 with support for 4 hardware push events

and generation of compare output pulses

- Energy Efficient Ethernet (EEE) support (802.3az)
- Flow Control Support (802.3x)
- Address Lookup Engine (ALE)
  - 64 total address entries plus VLANs
  - Wire rate lookup
  - Host controlled time-based aging and/or auto-aging
  - Multiple spanning tree support
  - L2 address lock and L2 filtering support
  - MAC authentication (802.1x)
  - Receive-based or destination-based multicast and broadcast rate limits
  - MAC address blocking
  - Source port locking
  - OUI (Vendor ID) host accept/deny feature
- VLAN support
  - 802.1Q compliant
    - Auto add port VLAN for untagged frames on ingress
    - Auto VLAN removal on egress and auto pad to minimum frame size
- Ethernet Statistics:
  - EtherStats and 802.3Stats Remote network Monitoring (RMON) statistics gathering (shared)
  - Programmable statistics interrupt mask when a statistic is above one half its 32-bit value
- Flow Control Support (802.3x)
- Digital loopback and FIFO loopback modes supported
- Maximum frame size 2016 bytes (2020 with VLAN)
- Management Data Input/Output (MDIO) module for PHY Management
- Emulation support
- Full duplex mode supported in 10/100/1000 Mbps. Half-duplex mode supported only in 10/100 Mbps.
- RAM Error Detection and Correction (SECDED)

Terminology:

AVB— Audio Video Bridging

AVBTP— Audio Video Bridging Trasport Protocol

BMCA— Best Master Clock Algorithm

CFI— Canonical Format Indicator

CPPI— Communications Port Programming Interface

DLR— Device Level Ring

DSCP— Differentiated Services Code Point

EEE— Energy Efficient Ethernet

EMAC— Ethernet Media Access Control

EOP— End of Packet

EOQ— End of Queue

IPG— Inter-Packet Gap

LPI— Low Power Indicator

MDIO— Management Data Input/Output

MOF— Middle of Frame

OUI— Organizationally Unique Identifier

PTP— Precision Time Protocol

RMON— Remote Monitoring

RTCP— RTP Control Protocol

RTP— Real-time Transport Protocol

SCR— Switched Central Resource

SRP— Stream Reservation Protocol

TOS— Type of Service

VLAN— Virtual Local Area Network

11.13.4.2 EMAC Environment

11.13.4.2.1 G/MII Interface

NOTE: G/MII interface in this device functions only in MII mode. In MII Mode (100/10 Mbps) EMAC operates in full-duplex and half-duplex.

The pin connections of the G/MII Interface is shown in Figure 11-910. The detailed description of the signals are listed in the following tables.

Figure 11-910. MII Interface Typical Application

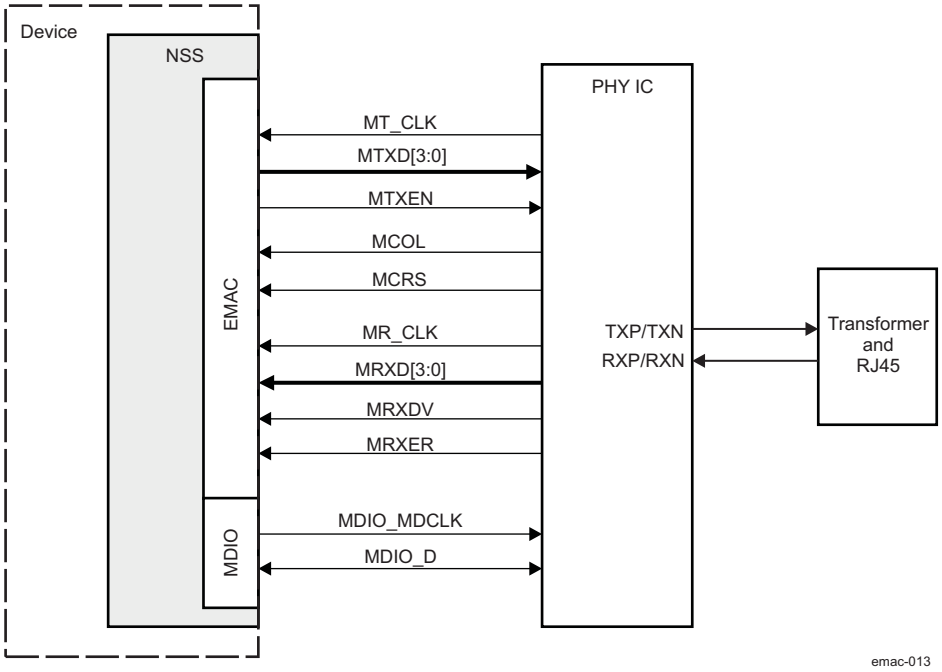


Table 11-2012. G/MII I/O Description in MII Mode

Signal	Device Pin	I/O[1]	Description
MT_CLK	MII_TXCLK	I	The transmit clock is a continuous clock that provides the timing reference for transmit operations. The MTXD and MTXEN signals are tied to this clock. The clock is generated by the PHY and is 2.5 MHz at 10 Mbps operation and 25 MHz at 100 Mbps operation.
MTXD[3:0]	MII_TXD[3:0]	O	The transmit data pins are a collection of 4 data signals MTXD[3:0] comprising 4 bits of data. MTXD0 is the least-significant bit (LSB). The signals are synchronized by MT_CLK and valid only when MTXEN is asserted.
MTXEN	MII_TXEN	O	The transmit enable signal indicates that the MTXD pins are generating 4bit data for use by the PHY. It is driven synchronously by MT_CLK.
MTXER	MII_TXER	O	Transmit data error. Used only for EEE to request PHY for low power transition.
MCOL	MII_COL	I	In half-duplex operation, the MCOL pin is asserted by the PHY when it detects a collision on the network. It remains asserted while the collision condition persists. This signal is not necessarily synchronous to MT_CLK nor MR_CLK. In full-duplex operation, the MCOL pin is used for hardware transmit flow control. Asserting the MCOL pin will stop packet transmissions; packets in the process of being transmitted when MCOL is asserted will complete transmission. The MCOL pin should be held low if hardware transmit flow control is not used.

(1) I = Input; O = Output

Table 11-2012. G/MII I/O Description in MII Mode (continued)

Signal	Device Pin	I/O[1]	Description
MCRS	MII_CRS	I	In half-duplex operation, the MCRS pin is asserted by the PHY when the network is not idle in either transmit or receive. The pin is de-asserted when both transmit and receive are idle. This signal is not necessarily synchronous to MT_CLK nor MR_CLK. In full-duplex operation, the MCRS pin should be held low.
MR_CLK	MII_RXCLK	I	The receive clock is a continuous clock that provides the timing reference for receive operations. The MRXD, MRXDV, and MRXER signals are tied to this clock. The clock is generated by the PHY and is 2.5 MHz at 10 Mbps operation and 25 MHz at 100 Mbps operation.
MRXD[3:0]	MII_RXD[3:0]	I	The receive data pins are a collection of 4 data signals comprising 4 bits of data. MRXD0 is the least-significant bit (LSB).The signals are synchronized by MR_CLK and valid only when MRXDV is asserted.
MRXDV	MII_RXDV	I	The receive data valid signal indicates that the MRXD pins are generating nibble data for use by the EMAC. It is driven synchronously to MR_CLK.
MRXER	MII_RXER	I	Receive Data Error input
MDIO_MDCLK	MDIO_CLK	O	Management data clock. The MDIO data clock is sourced by the MDIO module on the system. It is used to synchronize MDIO data access operations done on the MDIO_D pin.
MDIO_D	MDIO_DATA	I/O	MDIO data pin drives PHY management data into and out of the PHY by way of an access frame consisting of start of frame, read/write indication, PHY address, register address, and data bit cycles. The MDIO_D pin acts as an output for all but the data bit cycles at which time it is an input for read operations.

11.13.4.2.2 RMII Interface

Figure 11-911 shows a device with integrated EMAC and MDIO interfaced via a RMII connection in a typical system. The individual CPSW and MDIO signals for the RMII interface are summarized in Table 11-2013.

For more information, refer to either the IEEE 802.3 standard or ISO/IEC 8802-3:2000(E).

Figure 11-911. RMII Interface Typical Application

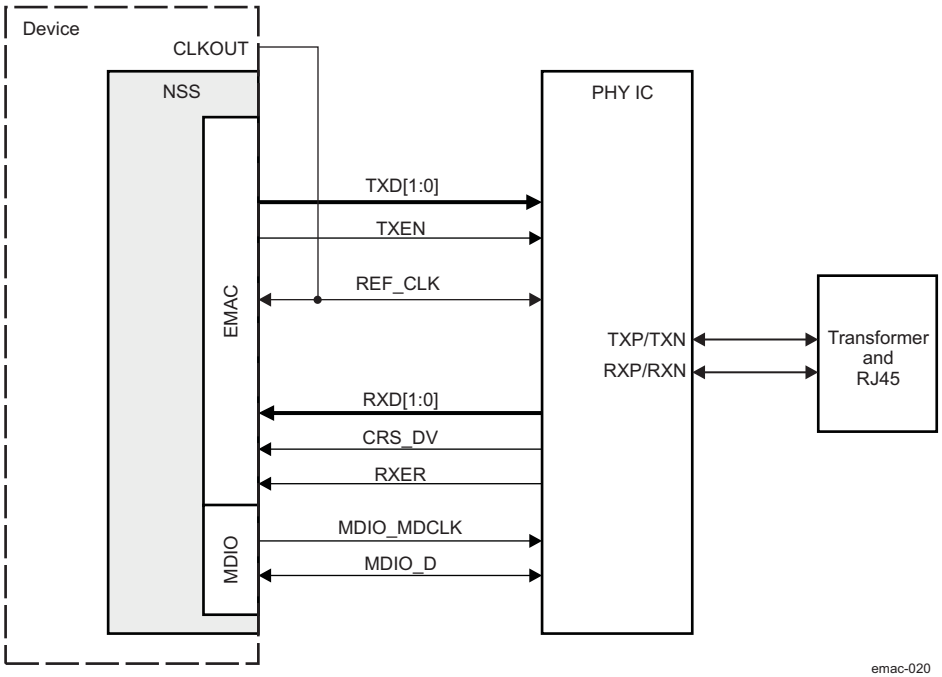


Table 11-2013. RMII I/O Description

Signal	Device Pin(s)	I/O[1]	Description
TXD[1:0]	RMII_TXD[1:0]	O	Transmit data . The transmit data pins are a collection of 2 bits of data. TXD0 is the least-significant bit (LSB). The signals are synchronized by RMII_MHZ_50_CLK and valid only when TXEN is asserted.
TXEN	RMII_TXEN	O	Transmit enable. The transmit enable signal indicates that the RMII_TXD pins are generating data for use by the PHY. RMII_TXEN is synchronous to RMII_MHZ_50_CLK
RMII0_MHZ_50_CLK	RMII_REFCLK	O	RMII reference clock. The reference clock is used to synchronize all RMII signals. RMII_MHZ_50_CLK must be continuous and fixed at 50 MHz.
RXD[1:0]	RMII_RXD[1:0]	I	Receive data. The receive data pins are a collection of 2 bits of data. RXD0 is the least-significant bit (LSB). The signals are synchronized by RMII_MHZ_50_CLK and valid only when CRS_DV is asserted and RXER is de-asserted.
CRS_DV	RMII_CRS_DV	I	Carrier sense/receive data valid. Multiplexed signal between carrier sense and receive data valid.
RXER	RMII_RXER	I	Receive error. The receive error signal is asserted to indicate that an error was detected in the received frame.
MDIO_MDCLK	MDIO_CLK	O	Management data clock (MDIO_MCLK). The MDIO data clock is sourced by the MDIO module on the system. It is used to synchronize MDIO data access operations done on the MDIO_D pin.
MDIO_D	MDIO_DATA	I/O	MDIO data pin drives PHY management data into and out of the PHY by way of an access frame consisting of start of frame, read/write indication, PHY address, register address, and data bit cycles. The MDIO_D pin acts as an output for all but the data bit cycles at which time it is an input for read operations.

(1) I = Input; O = Output

11.13.4.2.3 RGMII Interface

Figure 11-912 shows a device with integrated CPSW and MDIO interfaced via a RGMII connection in a typical system. The individual CPSW and MDIO signals for the RGMII interface are summarized in Table 11-2014.

Figure 11-912. RGMII Interface Typical Application

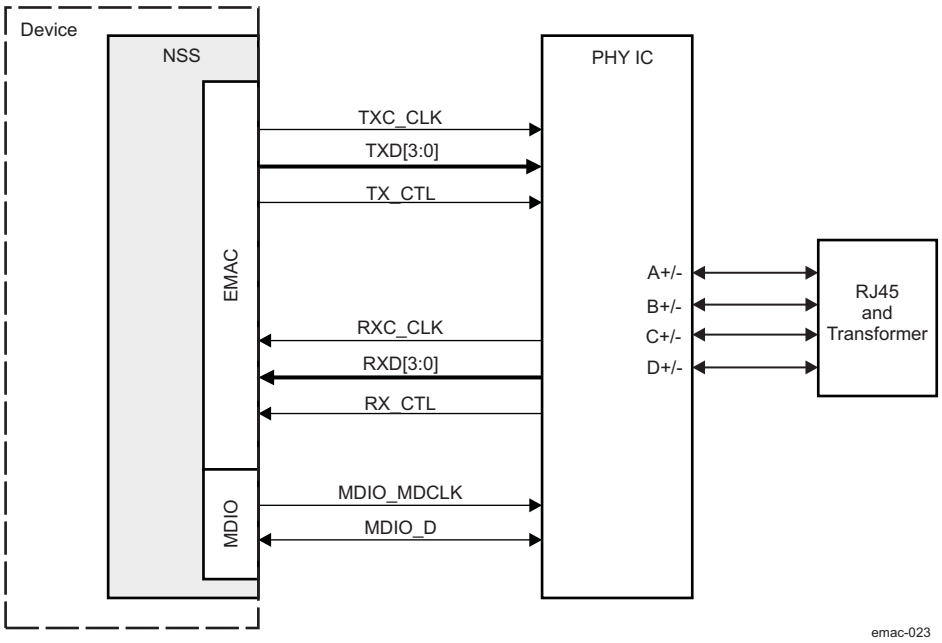


Table 11-2014. RGMII I/O Description

Signal	Device Pin(s)	I/O[1]	Description
TXD[3:0]	RGMII_TXD[3:0]	O	The transmit data pins are a collection of 4 bits of data. TXD0 is the least-significant bit (LSB). The signals are valid only when TX_CTL is asserted.
TX_CTL	RGMII_TXCTL	O	Transmit Control/enable. The transmit enable signal indicates that the TXD pins are generating data for use by the PHY.
TXC_CLK	RGMII_TXC	O	The transmit reference clock. The clock is 2.5 MHz at 10 Mbps operation, 25 MHz at 100 Mbps operation, and 125 MHz at 1000 Mbps of operation.
RXD[3:0]	RGMII_RXD[3:0]	I	The receive data pins are a collection of 4 bits of data. RXD0 is the least-significant bit (LSB). The signals are valid only when RX_CTL is asserted
RX_CTL	RGMII_RXCTL	I	The receive data valid/control signal indicates that the RXD pins are nibble data for use by the EMAC.
RXC_CLK	RGMII_RXC	I	The receive clock is a continuous clock that provides the timing reference for receive operations. The clock is generated by the PHY and is 2.5 MHz at 10 Mbps operation, 25 MHz at 100 Mbps operation, 125 MHz at 1000 Mbps of operation.
MDIO_MDCLK	MDIO_CLK	O	Management data clock (MDIO_MCLK). The MDIO data clock is sourced by the MDIO module on the system. It is used to synchronize MDIO data access operations done on the MDIO pin.
MDIO_D	MDIO_DATA	I/O	MDIO data pin drives PHY management data into and out of the PHY by way of an access frame consisting of start of frame, read/write indication, PHY address, register address, and data bit cycles. The MDIO_D pin acts as an output for all but the data bit cycles at which time it is an input for read operations.

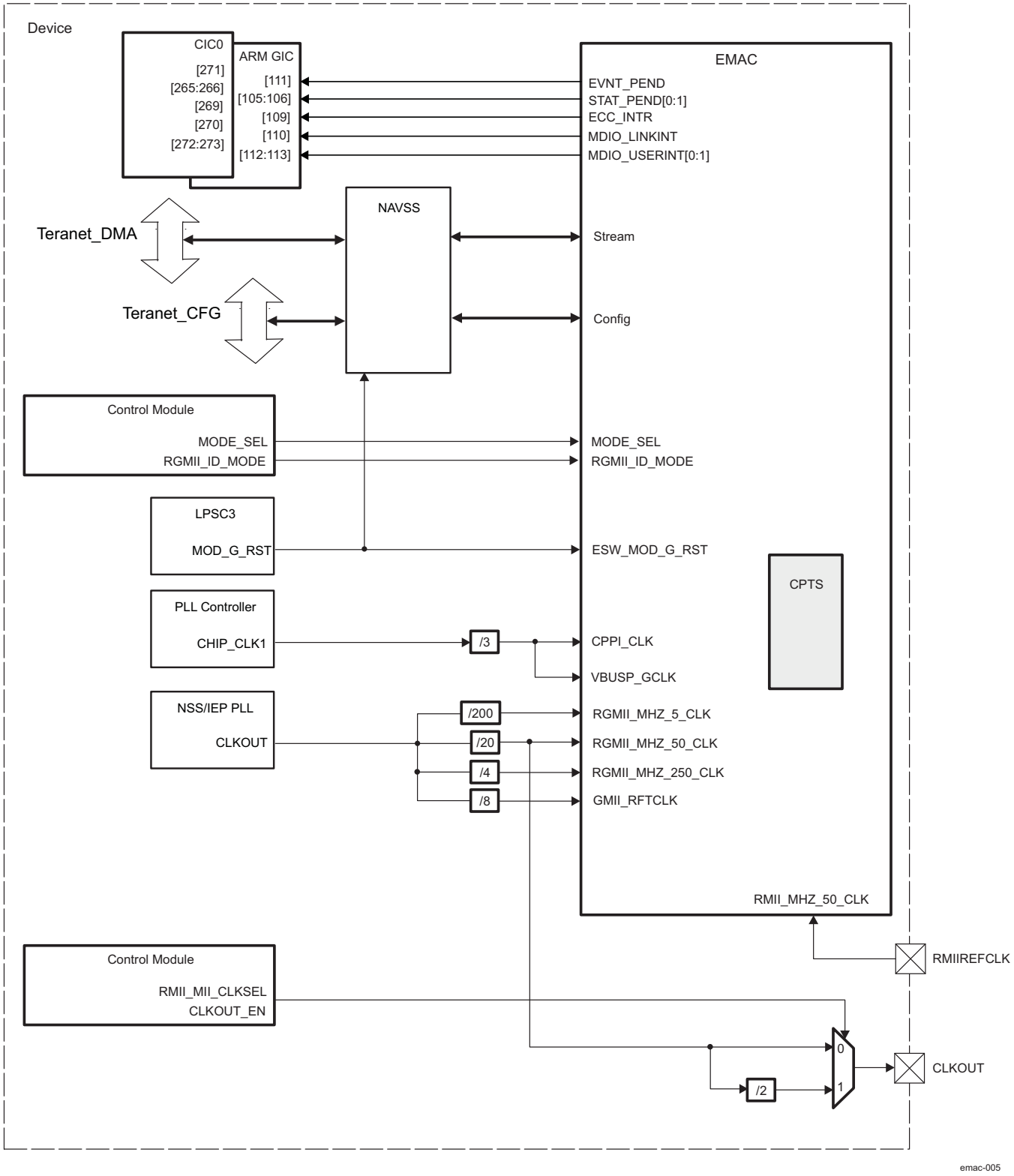
(1) I = Input; O = Output

NOTE: The control module registers assign the specific function to the device pads. For more information on control module settings, see Section 5.1.3.1.1, *Pad Configuration Registers* in *Section 5.1, Control Module (BOOT_CFG)* and Device *Data Manual*.

11.13.4.3 EMAC Integration

Figure 11-913 shows the integration of the EMAC module in the device.

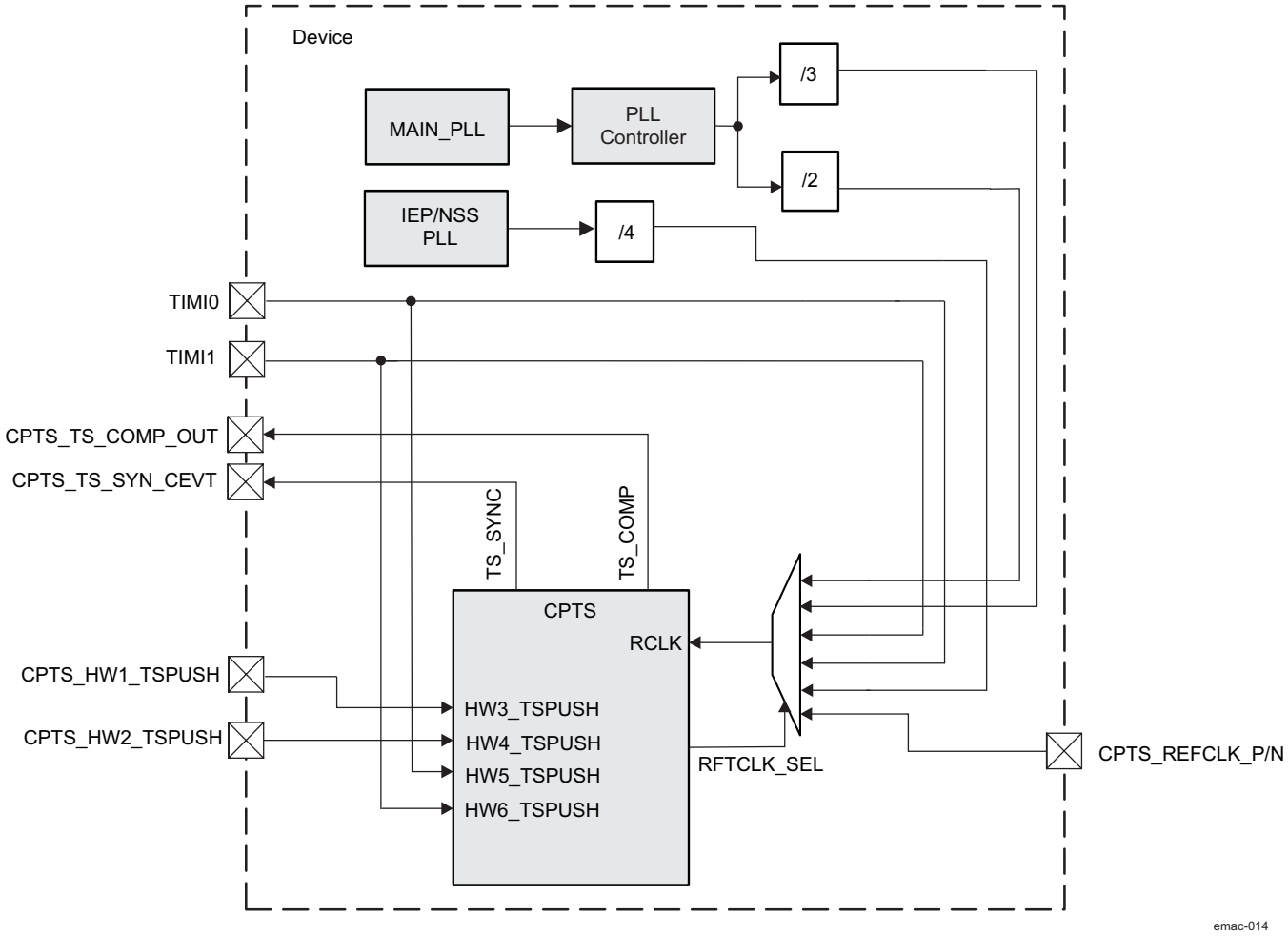
Figure 11-913. EMAC Integration



BOOT_CFG register references: ETHERNET_CFG, ETHERNET_CLKCTL, MACID0, MACID1.

Figure 11-914 shows CPTS integration in the device.

Figure 11-914. CPTS Integration



CPTS IEEE 1588 clock (RCLK) is selected through the CPTS_RFTCLK_SEL register.

NOTE: For CPTS functional description, see Section 11.13.4.4.7, *Common Platform Time Sync (CPTS)*.

Table 11-2015 through Table 11-2017 summarize the integration of the EMAC module in the device.

Table 11-2015. EMAC Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
EMAC	PD2 (NSS)	LPSC3	TeraNet_DMA TeraNet_CFG

Table 11-2016. EMAC Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
EMAC	CPPI_CLK	CHIP_CLK1/3	PLL Controller	CPPI packet streaming interface clock. Main clock for EMAC.
	VBUSP_GCLK	CHIP_CLK1/3	PLL Controller	VBUS interface clock. Min. 125 MHz for Gigabit mode.
	G/MII_RFTCLK	NSS/IEP PLLOUT/8	NSS/IEP PLL	125-MHz G/MII Gigabit mode clock
	RGMII_MHZ_5_CLK	NSS/IEP PLLOUT/200	NSS/IEP PLL	5-MHz RGMII clock
	RGMII_MHZ_50_CLK	NSS/IEP PLLOUT/20	NSS/IEP PLL	50-MHz RGMII clock
	RGMII_MHZ_250_CLK	NSS/IEP PLLOUT/4	NSS/IEP PLL	250-MHz RGMII clock
	RMII_MHZ_50_CLK	RMII_REFCLK	RMII_REFCLK pin	50-MHz RMII clock. Typically sourced from the CLKOUT pin
	CPTS_RFTCLK	CHIP_CLK1/2	PLL Controller	CPTS IEEE 1588 clock. Selected through the CPTS_RFTCLK_SEL register
		CHIP_CLK1/3	PLL Controller	
		TIMI0	TIMI0 pin	
		TIMI1	TIMI1 pin	
		NSS/IEP PLLOUT/4	NSS/IEP PLL	
		CPTS_REFCLK	CPTS_REFCLK_P/N pins	
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
EMAC	ESW_MOD_G_RST	MOD_G_RST	LPSC3	EMAC main reset

Table 11-2017. EMAC Hardware Requests

Interrupt Requests				
Module Instance	**Event Name**	**Mapped To Input Event [Number]**		**Description**
		ARM GIC	**CIC**	
EMAC	EVNT_PEND	[111]	[271]	CPTS Event Pending Interrupt
	STAT_PEND[0:1]	[105:106]	[265:266]	Statistics Pending Interrupts
	ECC_INTR	[109]	[269]	ECC Interrupt
	MDIO_LINKINT	[110]	[270]	MDIO Link Interrupt
	MDIO_USERINT[0:1]	[112:113]	[272:273]	MDIO User Interrupts

NOTE: For more information about the device interrupt controllers, see Chapter 9, *Interrupts*.

NOTE: For the description of the interrupt source, see Section 11.13.4.4.5, *Interrupt Functionality*.

11.13.4.4 EMAC Functional Description

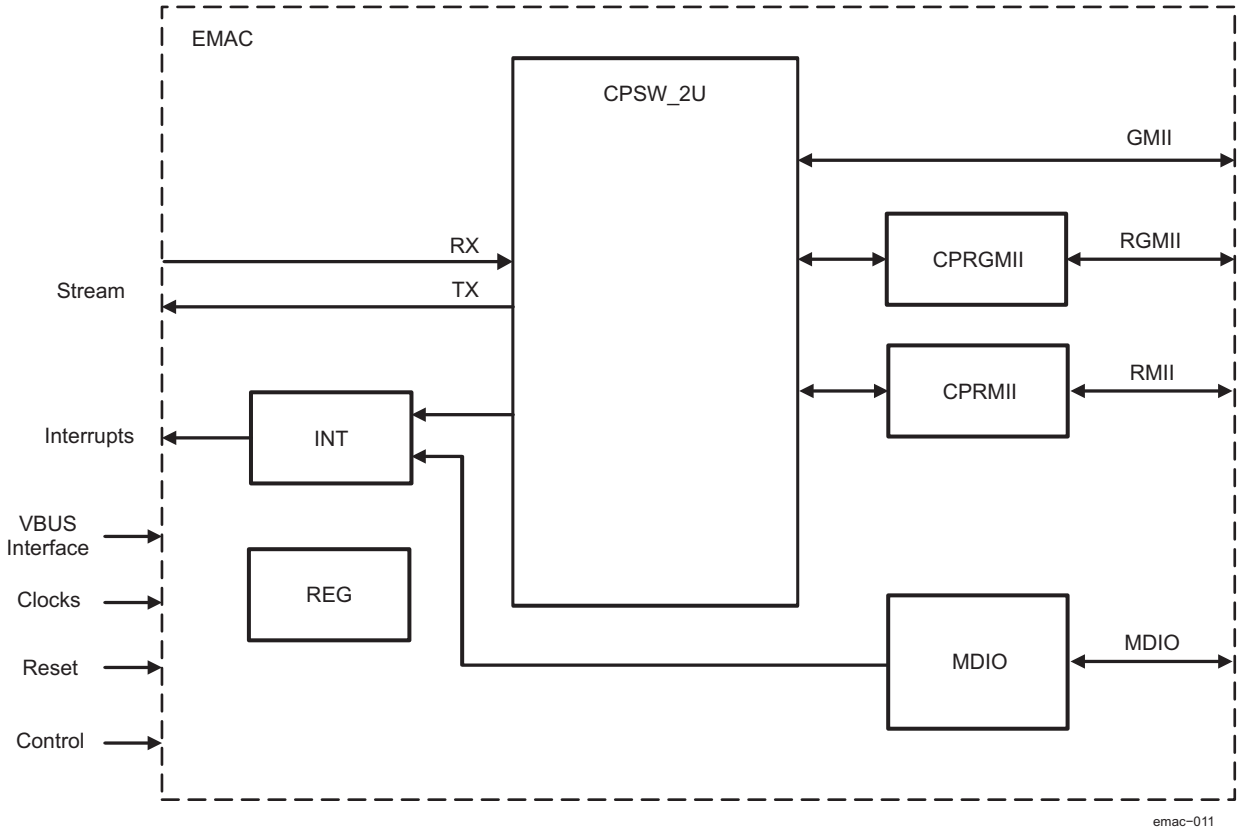
The EMAC subsystem modules are compliant to the IEEE Std 802.3 Specification. EMAC is shown in Figure 11-915.

11.13.4.4.1 Functional Block Diagram

The EMAC subsystem consists of:

- CPSW_2U which features one G/MII interface
- One RGMII interface module
- One RMII interface module
- One Host Port 0 CPPI Packet Streaming Interface
- One MDIO interface module
- One Interrupt Controller module

Figure 11-915. EMAC Top Level Block Diagram



11.13.4.4.2 EMAC Ports

Ethernet Subsystem has two ports. Port 0 is the Host port (internal to the Subsystem). Port 1 is the external ports connected to G/MII, RGMII, or RMII interfaces as per the interface selected.

Naming conventions followed in this section:

- Port0 is referred to the CPPI Host Port
- Port1 is referred to the interfaces GMII/RGMII/RMII

11.13.4.4.2.1 Interface Mode Selection

The EMAC Ethernet Subsystem has one 10/100/1000 Ethernet port with selectable MII, RMII, and RGMII interfaces.

The interface mode is selected by configuring the ethernet interface mode selection bitfield (MODE_SEL) in the ETHERNET_CFG register.

See the device Data Manual for configuring the pin mux mode as per the interface selected.

11.13.4.4.3 Clocking

11.13.4.4.3.1 Subsystem Clocking

EMAC clocking summary is shown in Section 11.13.4.3, *EMAC Integration*.

11.13.4.4.3.2 Interface Clocking

Data is transmitted and received with respect to the reference clocks of the interface pins.

11.13.4.4.3.2.1 G/MII Interface Clocking

GMII1_MR_CLK and GMII1_MT_CLK frequencies are fixed by the 802.3 specification.

- 2.5 MHz at 10 Mbps
- 25 MHz at 100 Mbps
- 125 MHz at 1000 Mbps

NOTE: G/MII interface functions only in MII mode (10/100 Mbps).

11.13.4.4.3.2.2 RGMII Interface Clocking

RGMII_RXC, RGMII_TXC frequencies are:

- 2.5 MHz at 10 Mbps
- 25 MHz at 100 Mbps
- 125 MHz at 1000 Mbps

11.13.4.4.3.2.3 RMII Interface Clocking

RMII interface clock RMII_50MHZ_CLK frequency is:

- 50 MHz at 10 Mbps
- 50 MHz at 100 Mbps

RMII_MII_CLKOUT_EN and RMII_MII_CLKSEL bits in the ETHERNET_CLKCTL enable the clocking of RMII interface. See Section 5.1, *Control Module (BOOT_CFG).*

11.13.4.4.3.2.4 MDIO Clocking

The MDIO clock is based on a divide-down of the interface (VBUSP_GCLK) clock, running at 125 MHz. The application software or driver must control the divide-down value.

See the MDIO_CONTROL register for configuring the Clock Divider (CLKDIV) value.

11.13.4.4.4 Software IDLE

The submodule software idle register bits enable CPSW_2U operation to be completely or partially suspended by software control. The idle state is entered at packet boundaries, and no further packet operations will occur on an idled submodule until the idle command is removed. The CPSW_SOFT_IDLE[0] SOFT_IDLE bit may be set if desired. The CPSW SOFT_IDLE bit causes packets to not be transferred from one FIFO to another FIFO internal to the switch. Idle status is determined by reading or polling the idle bit.

11.13.4.4.5 Interrupt Functionality

EMAC Ethernet Subsystem has four Interrupt outputs:

- CPTS_PEND – CPTS Event Pending Interrupt
- STAT_PEND[0:1] – Statistics Pending Interrupts
- ECC_INTR – ECC Interrupt
- MDIO_LINKINT - MDIO Link Interrupt
- MDIO_USERINT[0:1] – MDIO User Interrupts

11.13.4.4.5.1 EVNT_PEND (CPTS_PEND) Interrupt

See Section 11.13.4.4.7, *Common Platform Time Sync (CPTS)* for more details on this interrupt.

11.13.4.4.5.2 Statistics Interrupt

The statistics level interrupt (STAT_PEND) will be asserted, if enabled when any statistics value is greater than or equal to 8000 0000h. The statistics interrupt is cleared by writing to decrement all statistics values greater than 8000 0000h (such that their new values are less than 8000 0000h). The raw and masked statistics interrupt status may be read by reading the CPTS_INTSTAT_RAW and CPTS_INTSTAT_MASKED registers, respectively.

The Statistics interrupt is enabled by setting to 1 the TS_PEND_EN bit in the CPTS_INT_ENABLE register

11.13.4.4.5.3 MDIO Interrupts

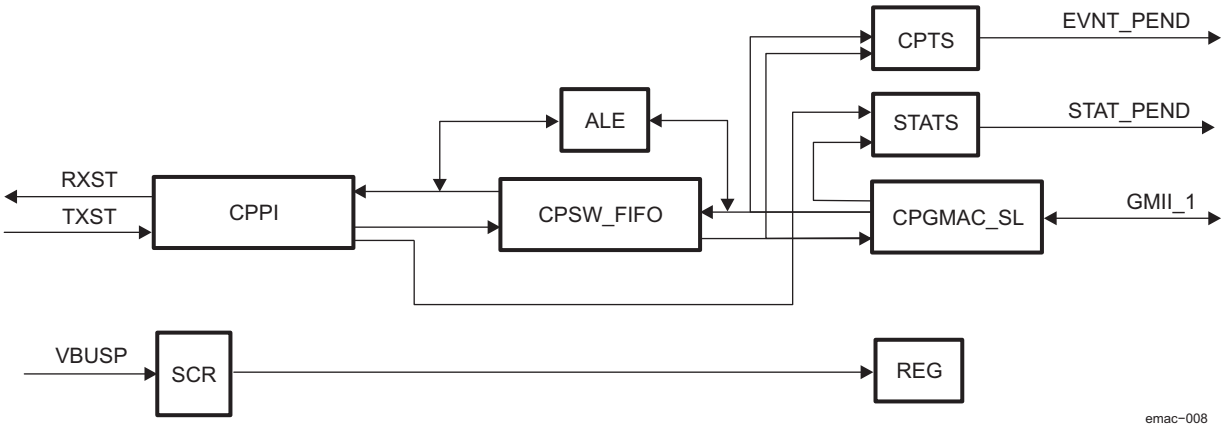
MDIO_LINKINT is set if there is a change in the link state of the PHY corresponding to the address in the PHYADR_MON field of the MDIO_USER_PHY_SEL_0 register and the corresponding LINKINT_ENABLE bit is set. The MDIO_LINKINT event is also captured in the MDIO_LINK_INT_MASKED register. When the GO bit in the MDIO_USER_ACCESS_0 register transitions from 1 to 0, indicating the completion of a user access, and the corresponding USERINTMASKSET bit in the MDIO_USER_INT_MASK_SET register is set, the MDIO_USERINT signal is asserted. The MDIO_USERINT event is also captured in the MDIO_USER_INT_MASKED register.

11.13.4.4.6 CPSW_2U

The CPSW_2U G/MII interface is compliant to the IEEE Std 802.3 Specification.

The CPSW_2U contains one CPGMAC_SL interface (Ethernet port 1), one CPPI interface Host Port (port 0), Common Platform Time Sync (CPTS), ALE Engine and Statistics (STATS). A top-level block diagram of the CPSW_2U is shown in Figure 11-916.

Figure 11-916. CPSW_2U Block Diagram



11.13.4.4.6.1 Address Lookup Engine (ALE)

The address lookup engine (ALE) processes all received packets to determine which port(s) if any that the packet should the forwarded to. The ALE uses the incoming packet received port number, destination address, source address, length/type, and VLAN information to determine how the packet should be forwarded. The ALE outputs the port mask to the switch fabric that indicates the port(s) the packet should be forwarded to. The ALE is enabled when the ENABLE_ALE bit in the ALE_CONTROL register is set. All packets are dropped when the ENABLE_ALE bit is cleared to 0.

In normal operation, the CPGMAC_SL modules are configured to issue an abort, instead of an end of packet, at the end of a packet that contains an error (runt, frag, oversize, jabber, crc, alignment, code etc.) or at the end of a mac control packet. However, when the CPSW_P1_MAC_CONTROL configuration bit(s) RX_CEF_EN, RX_CSF_EN, or RX_CMF_EN are set, error frames, short frames or mac control frames have a normal end of packet instead of an abort at the end of the packet. When the ALE receives a packet that contains errors (due to a set header error bit), or a mac control frame and does not receive an abort, the packet will be forwarded only to the host port (port 0). Packets with errors that are forwarded to the host have no vlan untagging or drop due to rate limiting. No ALE learning occurs on packets with errors or mac control frames. Learning is based on source address and lookup is based on destination address. Directed packets from the host are not learned, updated, or touched.

The ALE may be configured to operate in bypass mode by setting the ALE_BYPASS bit in the ALE_CONTROL register. When in bypass mode, all CPGMAC_SL received packets are forwarded only to the host port (port 0). In bypass mode, the ALE processes host port transmit packets the same as in normal mode. In general, packets would be directed by the host in bypass mode.

The ALE may be configured to operate in OUI deny mode by setting the ENABLE_OUI_DENY bit in the ALE_CONTROL register. When in OUI deny mode, a packet with a non-matching OUI source address will be dropped unless the destination address matches a multicast table entry with the super bit set. Broadcast packets will be dropped unless the broadcast address is entered into the table with the SUPER bit set. Unicast packets will be dropped unless the unicast address is in the table with BLOCK and SECURE both set (supervisory unicast packet).

Multicast supervisory packets are designated by the super bit in the table entry. Unicast supervisory packets are indicated when block and secure are both set. Supervisory packets are not dropped due to rate limiting, OUI, or VLAN processing.

11.13.4.4.6.1.1 Address Table Entry

The ALE table contains multiple table entry types. Each table entry represents a free entry, an address, a VLAN, an address/VLAN pair, or an OUI address. Software should ensure that there are not double address entries in the table. The double entry used would be indeterminate. Reserved table bits must be written with zeroes.

Source Address learning occurs for packets with a unicast, multicast or broadcast destination address and a unicast or multicast (including broadcast) source address. Multicast source addresses have the group bit (bit 40) cleared before ALE processing begins, changing the multicast source address to a unicast source address. A multicast address of all ones is the broadcast address which may be added to the table. A learned unicast source address is added to the table with the following control bits:

Table 11-2018. Learned Address Control Bits

Bit(s)	Value
unicast_type	11
block	0
secure	0

If a received packet has a source address that is equal to the destination address then the following occurs:

- The address is learned if the address is not found in the table.
- The address is updated if the address is found.
- The packet is dropped.

Table Entry Type

00 - Free Entry

01 - Address Entry : unicast or multicast determined by destination **address bit 40.**

10 - VLAN entry

11 - VLAN Address Entry : unicast or multicast determined by **address bit 40.**

11.13.4.4.6.1.1.1 Free Table Entry

Table 11-2019. Free (Unused) Address Table Entry Bit Values

70:62	61:60	59:0
Reserved	ENTRY_TYPE (00)	Reserved

11.13.4.4.6.1.1.2 Multicast Address Table Entry

Table 11-2020. Multicast Address Table Entry Bit Values

70:68	67:66	65	64	63:62	61:60	59:48	47:0
Reserved	PORT_MASK	SUPER	Reserved	MCAST_FWD_STATE	ENTRY_TYPE (01)	Reserved	MULTICAST_ADDRESS

Supervisory Packet (SUPER)

When set, this field indicates that the packet with a matching multicast destination address is a supervisory packet.

0: Non-supervisory packet

1: Supervisory packet

Port Mask(1:0) (PORT_MASK)

This 2-bit field is the port bit mask that is returned with a found multicast destination address. There may be multiple bits set indicating that the multicast packet may be forwarded to multiple ports (but not the receiving port).

Multicast Forward State (MCAST_FWD_STATE)

Indicates the port state(s) required for the received port on a destination address lookup in order for the multicast packet to be forwarded to the transmit port(s). A transmit port must be in the Forwarding state in order to forward the packet. If the transmit PORT_MASK has multiple set bits then each forward decision is independent of the other transmit port(s) forward decision.

00 - Forwarding

01 - Blocking/Forwarding/Learning

10 - Forwarding/Learning

11 - Forwarding

The forward state test returns a true value if both the RX and TX ports are in the required state.

Table Entry Type (ENTRY_TYPE)

Address entry type. Unicast or multicast determined by address bit 40.

01: Address entry. Unicast or multicast determined by address bit 40.

Packet Address (MULTICAST_ADDRESS)

This is the 48-bit packet MAC address. For an OUI address, only the upper 24-bits of the address are used in the source or destination address lookup. Otherwise, all 48-bits are used in the lookup.

11.13.4.4.6.1.1.3 VLAN/Multicast Address Table Entry

Table 11-2021. VLAN/Multicast Address Table Entry Bit Values

70:68	67:66	65	64	63:62	61:60	59:48	47:0
Reseved	PORT_MASK	SUPER	Reserved	MCAST_FWD_STATE	ENTRY_TYPE (11)	VLAN_ID	MULTICAST_ADDRESS

Supervisory Packet (SUPER)

When set, this field indicates that the packet with a matching multicast destination address is a supervisory packet.

0: Non-supervisory packet

1: Supervisory packet

Port Mask(1:0) (PORT_MASK)

This 2-bit field is the port bit mask that is returned with a found multicast destination address. There may be multiple bits set indicating that the multicast packet may be forwarded to multiple ports (but not the receiving port).

Multicast Forward State (MCAST_FWD_STATE)

Indicates the port state(s) required for the received port on a destination address lookup in order for the multicast packet to be forwarded to the transmit port(s). A transmit port must be in the Forwarding state in order to forward the packet. If the transmit PORT_MASk has multiple set bits then each forward decision is independent of the other transmit port(s) forward decision.

00 - Forwarding

01 - Blocking/Forwarding/Learning

10 - Forwarding/Learning

11 - Forwarding

The forward state test returns a true value if both the RX and TX ports are in the required state.

Table Entry Type (ENTRY_TYPE)

Address entry type. Unicast or multicast determined by address bit 40.

11: VLAN address entry. Unicast or multicast determined by address bit 40.

VLAN ID (VLAN_ID)

The unique identifier for VLAN identification. This is the 12-bit VLAN ID.

Packet Address (MULTICAST_ADDRESS)

This is the 48-bit packet MAC address. For an OUI address, only the upper 24-bits of the address are used in the source or destination address lookup. Otherwise, all 48-bits are used in the lookup.

11.13.4.4.6.1.1.4 Unicast Address Table Entry

Table 11-2022. Unicast Address Table Entry Bit Values

70:68	67:66	65	64	63:62	61:60	59:48	47:0
Reseved	PORT_NUMBER	BLOCK	SECURE	UNICAST_TYPE (00) or (X1)	ENTRY_TYPE (01)	Reserved	UNICAST_ADDRESS

Port Number (PORT_NUMBER)

This field indicates the port number (not port mask) that the packet with a unicast destination address may be forwarded to. Packets with unicast destination addresses are forwarded only to a single port (but not the receiving port).

Block (BLOCK)

The block bit indicates that a packet with a matching source or destination address should be dropped (block the address).

0 - Address is not blocked.

1 - Drop a packet with a matching source or destination address (secure

must be zero)

If block and secure are both set, then they no longer mean block and secure. When both are set, the block and secure bits indicate that the packet is a unicast supervisory (super) packet and they determine the unicast forward state test criteria. If both bits are set then the packet is forwarded if the receive port is in the Forwarding/Blocking/Learning state. If both bits are not set then the packet is forwarded if the receive port is in the Forwarding state.

Secure (SECURE)

This bit indicates that a packet with a matching source address should be dropped if the received port number is not equal to the table entry port_number.

0 - Received port number is a don't care.

1 - Drop the packet if the received port is not the secure port for the source

address and do not update the address (block must be zero)

Unicast Type (UNICAST_TYPE)

This field indicates the type of unicast address the table entry contains.

00 - Unicast address that is not ageable.

01 - Ageable unicast address that has not been touched.

10 - OUI address - lower 24-bits are don't cares (not ageable).

11 - Ageable unicast address that has been touched.

Table Entry Type (ENTRY_TYPE)

Address entry. Unicast or multicast determined by address bit 40.

01: Address entry. Unicast or multicast determined by address bit 40.

Packet Address (UNICAST_ADDRESS)

This is the 48-bit packet MAC address. All 48-bits are used in the lookup.

11.13.4.4.6.1.1.5 OUI Unicast Address Table Entry

Table 11-2023. OUI Unicast Address Table Entry Bit Values

70:64	63:62	61:60	59:48	47:24	23:0
Reserved	UNICAST_TYPE (10)	ENTRY_TYPE (01)	Reserved	UNICAST_OUI	Reserved

Unicast Type (UNICAST_TYPE)

This field indicates the type of unicast address the table entry contains.

00 - Unicast address that is not ageable.

01 - Ageable unicast address that has not been touched.

10 - OUI address - lower 24-bits are don't cares (not ageable).

11 - Ageable unicast address that has been touched.

Table Entry Type (ENTRY_TYPE)

Address entry. Unicast or multicast determined by address bit 40.

01: Address entry. Unicast or multicast determined by address bit 40.

Packet Address (UNICAST_OUI)

For an OUI address, only the upper 24-bits of the address are used in the source or destination address lookup.

11.13.4.4.6.1.1.6 VLAN/Unicast Address Table Entry

Table 11-2024. Unicast Address Table Entry Bit Values

70:68	67:66	65	64	63:62	61:60	59:48	47:0
Reseved	PORT_NUMBER	BLOCK	SECURE	UNICAST_TYPE (00) or (X1)	ENTRY_TYPE (11)	VLAN_ID	UNICAST_ADDRESS

Port Number (PORT_NUMBER)

This field indicates the port number (not port mask) that the packet with a unicast destination address may be forwarded to. Packets with unicast destination addresses are forwarded only to a single port (but not the receiving port).]

Block (BLOCK)

The block bit indicates that a packet with a matching source or destination address should be dropped (block the address).

0 - Address is not blocked.

1 - Drop a packet with a matching source or destination address (secure must be zero)

If block and secure are both set, then they no longer mean block and secure. When both are set, the block and secure bits indicate that the packet is a unicast supervisory (super) packet and they determine the unicast forward state test criteria. If both bits are set then the packet is forwarded if the receive port is in the Forwarding/Blocking/Learning state. If both bits are not set then the packet is forwarded if the receive port is in the Forwarding state.

Secure (SECURE)

This bit indicates that a packet with a matching source address should be dropped if the received port number is not equal to the table entry PORT_NUMBER.

0 - Received port number is a don't care.

1 - Drop the packet if the received port is not the secure port for the source address and do not update the address (block must be zero)

Unicast Type (UNICAST_TYPE)

This field indicates the type of unicast address the table entry contains.

00 - Unicast address that is not ageable.

01 - Ageable unicast address that has not been touched.

10 - OUI address - lower 24-bits are don't cares (not ageable).

11 - Ageable unicast address that has been touched.

Table Entry Type (ENTRY_TYPE)

Address entry. Unicast or multicast determined by address bit 40.

11: VLAN address entry. Unicast or multicast determined by address bit 40.

VLAN ID (VLAN_ID)

The unique identifier for VLAN identification. This is the 12-bit VLAN ID.

Packet Address (UNICAST_ADDRESS)

This is the 48-bit packet MAC address. All 48-bits are used in the lookup.

11.13.4.4.6.1.1.7 VLAN Table Entry

Table 11-2025. VLAN Table Entry

70:68	67	66	65	64:62	61:60	59:48	47:46	45:44
Reserved	NO_LEARN_MASK	Reserved	VLAN_FORCE_INGRESS_CHECK	Reserved (000)	ENTRY_TYPE (10)	VLAN_ID	Reserved	REG_MCAST_FLOOD_INDEX

43:26	25:24	23:22	21:20	19:2	1:0
Reserved	FORCE_UNTAGGED_EGRESS	Reserved	UNREG_MCAST_FLOOD_INDEX	Reserved	VLAN_MEMBER_LIST

No Learn Mask (NO_LEARN_MASK)

When a bit is set in this mask, a packet with an unknown source address received on the associated port will not be learned (i.e. When a VLAN packet is received and the source address is not in the table, the source address will not be added to the table).

VLAN Force Ingress Check (VLAN_FORCE_INGRESS_CHECK)

If the receive port is not a member of this VLAN then the packet is dropped. This is similar to the VID_INGRESS_CHECK bit in the ALE_PORT_CONTROL_0 registers except this check is for this VLAN only (not all VLANs).

Table Entry Type (ENTRY_TYPE)

10: VLAN entry

VLAN ID (VLAN_ID)

The unique identifier for VLAN identification. This is the 12-bit VLAN ID.

Registered Multicast Flood Index (REG_MCAST_FLOOD_INDEX)

Index into ALE_VLAN_MASK_MUX_0 to ALE_VLAN_MASK_MUX_3 register array that is used to create the registered multicast flood mask.

Force Untagged Packet Egress (FORCE_UNTAGGED_EGRESS)

This field causes the packet VLAN tag to be removed on egress (except on port 0).

Unregistered Multicast Flood Mask (UNREG_MCAST_FLOOD_INDEX)

Index into ALE_VLAN_MASK_MUX_0 to ALE_VLAN_MASK_MUX_3 register array that is used to create the unregistered multicast flood mask.

VLAN Member List (VLAN_MEMBER_LIST)

This field indicates which port(s) are members of the associated VLAN. One bit per port.

11.13.4.4.6.1.2 ALE Policing and Classification

The ALE has a number of configurable classifier engines (policers) that can be used for classification. Classification is a subset of the policing function and uses a policer without the color marking or rate limiting functions. A policer is a hardware engine that is used for policing. The POLICERS_DIV_8 field in the ALE_STATUS register indicates the number of policers available to be used for classification. Each policer can be enabled to match on one or more of any of the below packet fields for classification. All but Port and Priority are index references to the ALE table entries.

- Port Number
- Priority extracted from VLAN, mapped from DSCP if enabled, or Default Port Priority
- Organization Network Unique identifier - OUI
- Destination Address - DA
- Source Address - SA
- VLANID
- Ether Type
- IP Source Address - IPSA with full CIDR masking
- IP Destination Address - IPSA with full CIDR masking
- Support Host Thread/Flow ID mapping based on any packet classification above

11.13.4.4.6.1.2.1 ALE Classification

When the policers are configured as classifiers, the color marking and policing functions of the policing/classifier engines are not used. One or multiple classifiers can be configured to match on a single packet. For example, a classifier can be enabled to match on priority while another classifier could match IP address.

11.13.4.4.6.1.2.2 Classifier to CPPI Transmit Flow ID Mapping

The ALE can generate a 6-bit transmit CPPI Flow ID based on classifier matches that can be used instead of the switch default transmit Flow ID mapping. The switch default flow ID is the remapped received packet priority (0 to 7). Thread and flow ID are used interchangeably for this since there is a single hardware thread (TXST_THREAD_MREADY) but there are 6-bits of FLOW_ID in the transmit CPPI INFO word 0. When enabled, the highest classifier match can map to a particular 6-bit flow ID value that is associated with the classifier. The ALE also supports an optional ALE default thread/flow ID value in the event that no classifiers match. Each thread/flow ID, including the ALE default thread/flow ID, has an enable such that the ALE default thread/Flow ID is used if enabled and if no matches occur (instead of the remapped received packet priority). If the ALE default is not enabled and no matches occur then the switch default value will be used. If multiple classifier matches occur, the highest match with a thread enable bit set will be used. The resultant flow ID has the CPSW_P0_FLOW_ID_OFFSET register value added to it to determine the actual value in the INFO 0 Flow ID field.

Three registers are used for ALE classification thread/flow ID mapping configuration (ALE_THREAD_DEF, ALE_THREAD_CTL, and ALE_THREAD_VAL). The three thread mapping registers are used independently and are separate from the other ALE policing registers. The ALE_THREAD_CTL register allows the ALE_THREAD_VAL register contents to be written to the selected classifier. There is a single ALE_THREAD_DEF that is used for all classifiers. The thread mapping registers can be written or changed at any time but any packets that are already processed will not have their thread altered.

11.13.4.4.6.1.3 DSCP

The ALE can map DSCP field to priority prior to classification matching. When enabled the DSCP is mapped via 64 priority entries such that any DSCP value can be mapped to any of the eight priorities. When a packet is received without a VLAN priority this remapped priority can be used instead of the default Port VLAN priority field. See CPSW_P0_RX_DSCP_MAP_0 and CPSW_P1_RX_DSCP_MAP_0 registers in the Register Manual section for DSCP mapping.

11.13.4.4.6.1.4 Packet Forwarding Processes

There are four processes that an incoming received packet may go through to determine packet forwarding. The processes are *Ingress Filtering*, *VLAN_Aware Lookup*, *VLAN_Unaware Lookup*, and *Egress.*

Packet processing begins in the Ingress Filtering process. Each port has an associated packet forwarding state that can be one of four values (Disabled, Blocked, Learning, or Forwarding). The default state for all ports is Disabled. The host sets the packet forwarding state for each port.

In the packet ingress process (receive packet process), there is a forward state test for unicast destination addresses and a forward state test for multicast addresses. The multicast forward state test indicates the port states required for the receiving port in order for the multicast packet to be forwarded to the transmit port(s). A transmit port must be in the Forwarding state for the packet to be forwarded for transmission. The mcast_fwd_state indicates the required port state for the receiving port as indicated in the preceding table. The unicast forward state test indicates the port state required for the receiving port in order to forward the unicast packet. The transmit port must be in the Forwarding state in order to forward the packet. The block and secure bits determine the unicast forward state test criteria. If both bits are set then the packet is forwarded if the receive port is in the Forwarding/Blocking/Learning state. If both bits are not set then the packet is forwarded if the receive port is in the Forwarding state. The transmit port must be in the Forwarding state regardless. The forward state test used in the ingress process is determined by the destination address packet type (multicast/unicast).

In general, packets received with errors are dropped by the address lookup engine without learning, updating, or touching the address. The error condition and the abort are indicated by the CPGMAC_SL to the ALE. Packets with errors may be passed to the host (not aborted) by a CPGMAC_SL port, if the port has the RX_CMF_EN, RX_CEF_EN, or RX_CSF_EN bit(s) set. Error packets that are passed to the host by the CPGMAC_SL are considered to be bypass packets by the ALE and are sent only to the host. Error packets do not learn, update, or touch addresses regardless of whether they are aborted or sent to the host. Packets with errors received by the host are forwarded as normal.

The following control bits are in the CPSW_P1_MAC_CONTROL register:

- [22] RX_CEF_EN - enables frames that are fragments, long, jabber, CRC, code, and alignment errors to be forwarded
- [23] RX_CSF_EN - enables short frames to be forwarded
- [24] RX_CMF_EN - enables MAC control frames to be forwarded.

11.13.4.4.6.1.5 Learning Process

The learning process is applied to each receive packet that is not aborted. The learning process is a concurrent process with the packet forwarding process.

11.13.4.4.6.1.6 VLAN Aware Mode

The CPSW_2U is in VLAN aware mode when the VLAN_AWARE bit is set in the CPSW_CONTROL register.

In VLAN aware mode, transmitted packet data is changed depending on the packet type (PKT_TYPE), packet priority (PKT_PRI), and VLAN information.

11.13.4.4.6.1.7 VLAN Unaware Mode

An egress port is operating in the VLAN unaware mode when the VLAN_AWARE bit in the CPSW_CONTROL register is cleared to zero. In VLAN unaware mode, transmit (egress) packets are not modified on egress.

11.13.4.4.6.2 Packet Priority Handling

Packets are received on two ports, one CPGMAC_SL Ethernet port and one CPPI host port. Received packets have a received packet priority (0 to 7, with 7 being the highest priority).

The received packet priority is determined as follows:

1. If the first packet LTYPE = 0x8100 then the received packet priority is the packet priority (VLAN tagged and priority tagged packets).
2. Else if the first packet LTYPE = 0x0800 and byte 14 (following the LTYPE) is equal to 0x4X, and DSCP_IPV4_EN is set in CPSW_P0_CONTROL or CPSW_P1_CONTROL, then the received packet priority is the 6-bit TOS field in byte 15 (upper 6 bits) mapped through the port's DSCP priority mapping registers (IPv4 packet).
3. Else if the first packet LTYPE = 0x86DD and the most significant nibble of byte 14 (following the LTYPE) is equal to 0x6, and DSCP_IPV6_EN is set in CPSW_P0_CONTROL or

CPSW_P1_CONTROL, then the received packet priority is the 6-bit priority (in the 6-bits following the upper nibble 0x6) mapped through the port's DSCP priority mapping registers (IPv6 packet).

4. Else the received packet priority is the source (ingress) port priority.

The received packet priority is mapped through the receive ports associated packet-priority-to-header-packet-priority-mapping register to obtain the header packet priority. The header packet priority is the hardware switch priority. The header packet priority is also used as the actual transmit packet priority if the VLAN information is to be sent on egress.

The CPPI Port (port 0) also has a received packet thread. The received packet thread is determined exactly as the received packet priority except for untagged packets. For untagged packets, the received packet thread is the packet streaming interface thread that the packet was received on (instead of the port VLAN priority or DSCP priority). The received packet thread is the hardware switch priority. The received packet thread only determines which hardware switch priority the packet should be sent to, the egress VLAN rules are identical to packets that were received on Ethernet ports (as determined by the header packet priority).

11.13.4.4.6.3 Packet CRC Handling

11.13.4.4.6.3.1 Ethernet Port Ingress Packet CRC

All Ethernet ports check the ingress packet CRC in all modes/speeds. The receive port can check either Ethernet CRC or Castagnoli CRC as determined by the CRC_TYPE bit in the CPSW_P1_MAC_CONTROL register.

11.13.4.4.6.3.2 Ethernet Port Egress Packet CRC

Ethernet ports transmit each egress packet with the CRC selected by the CRC_TYPE bit in the CPSW_P1_MAC_CONTROL register, regardless of the type of CRC that the packet had on ingress to the switch. At the egress port after passing through the switch, the packet CRC is checked for correctness and if the CRC is correct then the packet is output with the generated selected output CRC. If the packet CRC is incorrect, due either to a bit flip in a memory or an error CRC passed in on host ingress, then the generated egress CRC type is used with at least a single byte of the CRC inverted to indicate the error. If the packet length including CRC is divisible by 4 then all 4 CRC bytes will be inverted on error. If there are three bytes remainder after dividing the packet length by 4 then three bytes will be inverted (and so on down to one byte remainder).

11.13.4.4.6.3.3 CPPI Port Ingress Packet CRC

CPPI port ingress packets can be passed in with or without a CRC. The host port is Ethernet CRC only. CPPI ingress packets are not checked for CRC correctness on CPPI ingress, however they are checked for correctness on Ethernet egress and are output with a CRC error if they came in with a CRC error. If a CPPI ingress packet does not have the PASSED_CRC bit set then a CRC will be generated for the packet on CPPI ingress. If the PASSED_CRC bit is set then the packet is received and forwarded unchanged. The PASSED_CRC bit is set via bit 3 of "Protocol Specific Flags" (ps_flags) in the packet descriptor. The CRC_TYPE is set via bit 2 of "Protocol Specific Flags" (ps_flags) in the packet descriptor and must be Ethernet CRC (CRC_TYPE=0).

11.13.4.4.6.3.4 CPPI Port Egress Packet CRC

The P0_TX_CRC_REMOVE bit in the CPSW_CONTROL register determines if CPPI egress packet have an Ethernet CRC included or not.

11.13.4.4.6.4 Audio Video Bridging

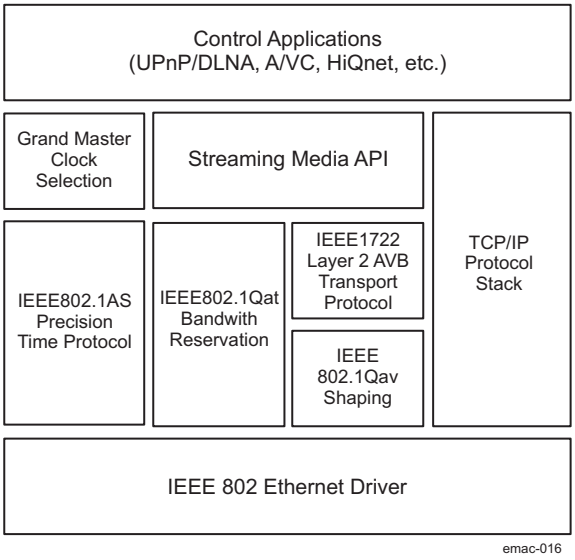
Audio Video Bridging is a project of IEEE 802.1 concerned with enabling low-latency streaming of time-sensitive audiovisual data over networks. Devices are designated as talkers (transmitters), bridges, or listeners (receivers). It is suggested that the maximum latency could be 2 ms over 7 hops for Class A devices and 20 ms over 7 hops for Class B devices. A hop is essentially a single local area network stage in the journey of a packet. Every time a bridge is encountered between one network section and another a hop is involved. One of the performance goals is that AVB streams will not use more than 75 percent of a link's bandwidth, leaving the remaining capacity for non-AVB streams.

The goal of developing AVB is simply--extend Ethernet's data-networking capabilities to the realm of reliable real-time audio/video networking.

An "Audio Video Bridging" network is one that implements a set of protocols being developed by the IEEE 802.1 Audio/Video Bridging Task Group. There are four primary differences between the proposed Audio Video Bridging architecture and existing 802 architectures (from now on the term "AVB" will be used instead of "Audio Video Bridging"):

1. Precise synchronization - IEEE 802.1AS: "*Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks.* "a.k.a Precision Time Protocol (PTP).
2. Traffic shaping for media streams - IEEE 802.1Qav: "*Virtual Bridged Local Area Networks:* Forwarding and Queuing for Time-Sensitive Streams."
3. Admission controls - IEEE 802.1Qat: "*Virtual Bridged Local Area Networks - Amendment 9:* Stream Reservation Protocol (SRP)."
4. Identification of non-participating devices - IEEE 802.1BA: "Audio/Video Bridging (AVB) Systems"

Figure 11-917. The Network Static with AVB



The following sections describe the media transport protocols that work within the AVB framework.

11.13.4.4.6.4.1 IEEE 802.1AS: Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks (Precision Time Protocol (PTP))

The protocol defined by 802.1AS automatically selects a device to be the master clock, and then distributes this clock throughout the bridged LAN / IP subnet to all other network devices using link-specific transmit/receive time-stamping. However, we only use a two-step solution only on transmit. That is, we do not modify a packet with the timestamp on the way out. The timestamp packet is sent out and then a separate message with the timestamp is sent by the host afterward. Receive can be one or two step.

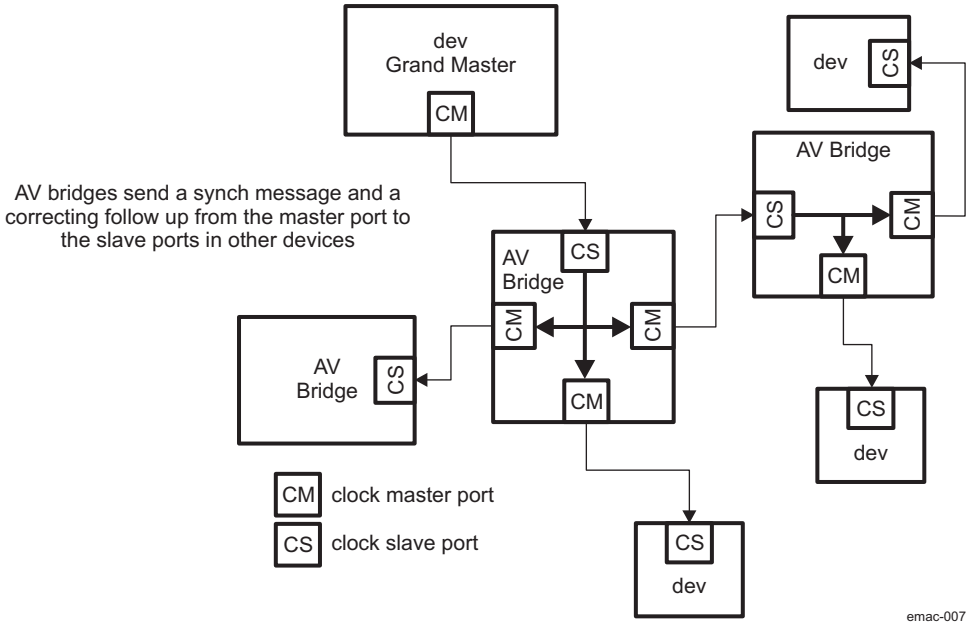
NOTE: The 802.1AS-distributed clock is not used as a media clock. Rather, the shared 802.1AS clock reference is used to regenerate the media clock at the listener/renderer. Such a reference removes the need to force the latency of the network to be constant, or compute long running averages in order to estimate the actual media rate of the transmitter in the presence of substantial network jitter. IEEE 802.1AS is based on the ratified IEEE 1588 standard.

Based on IEEE 1588:2002,Â PTP devices exchange standard Ethernet messages that synchronize network nodes to a common time reference by defining clock master selection and negotiation algorithms, link delay measurement and compensation, and clock rate matching and adjustment mechanisms.

Designed as a simplified profile of IEEE 1588, a primary difference between 1588 and IEEE 802.1AS is that PTP is a layer 2--in other words, a non-IP routable protocol. Like IEEE 1588, PTP defines an automatic method for negotiating the network clock master, the Best Master Clock Algorithm (BMCA). PTP nodes can be assigned one of eight priority levels, presumably based on clock quality. BMCA defines the underlying negotiation and signaling mechanism whose purpose is to identify the AVB LAN Grandmaster. Once a Grandmaster has been selected, synchronization automatically begins.

At the core of 802.1AS synchronization is time-stamping. In short, during PTP message ingress/egress from the 802.1AS-capable MAC, the PTP Ether type triggers the sampling of the value of a local real-time counter (RTC). Slave nodes compare the value of their RTC against the PTP Grandmaster and, by use of link delay measurement and compensation techniques, match their RTC value to the time of the AVB LAN PTP domain. After network time throughout the AVB LAN has converged, periodic SYNC and FOLLOW_UP messages provide the information that enables the PTP rate matching adjustment algorithms. The result is all PTP nodes are then synchronized to the same "Wall Clock" time. PTP assures 1-µs accuracy over seven network hops.

Figure 11-918. AVB Network & PTP Clock Entities

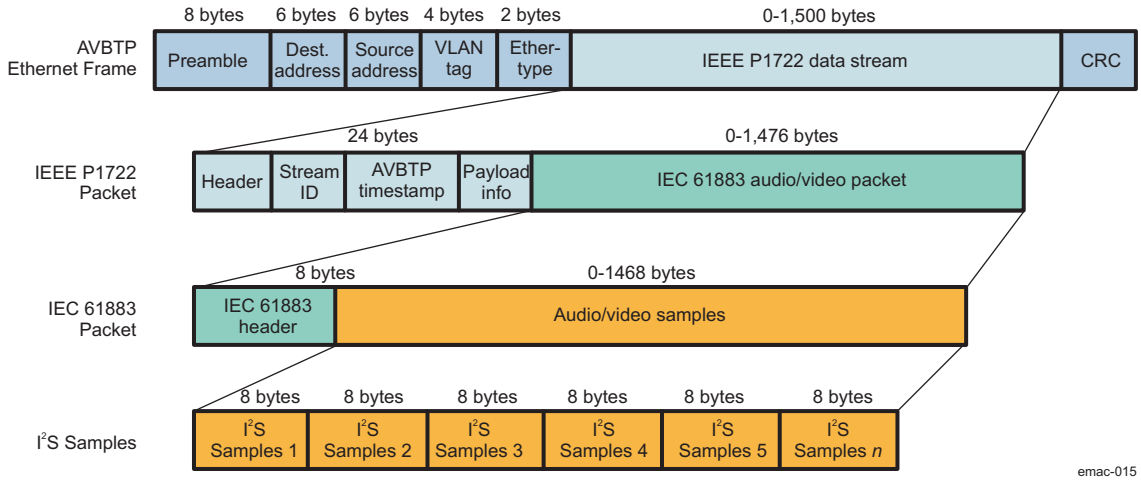


The media transport protocols that work within the AVB framework are:

11.13.4.4.6.4.1.1 IEEE 1722: "Layer 2 Transport Protocol for Time-Sensitive Streams"

AVBTP or 1722 sits above the IEEE 802.1 AVB plumbing and below the application layer. It acts as the conduit between an Ethernet MAC and a streaming application. AVBTP abstracts the underlying network transmission channel to enable the virtual connection of distributed audio and video CODECs over reliable Ethernet networks. A complete AVBTP Ethernet packet is shown in Figure 11-919 and illustrates how IEC 61883-6 AM824 uncompressed audio samples are encapsulated in an Ethernet frame.

Figure 11-919. IEEE 1722 Packets



1722 or AVBTP Presentation Time and Synchronization:

Synchronization in an AVB network starts with the Precision Time Protocol but ends with synchronized media clocks. PTP is responsible for synchronizing all nodes in an AVB network to identical wall clock time; not for synchronizing media clocks. In other words, PTP does not actually transport synchronized media clocks but instead provides a low-level building block crucial for managing a distributed media synchronization system.

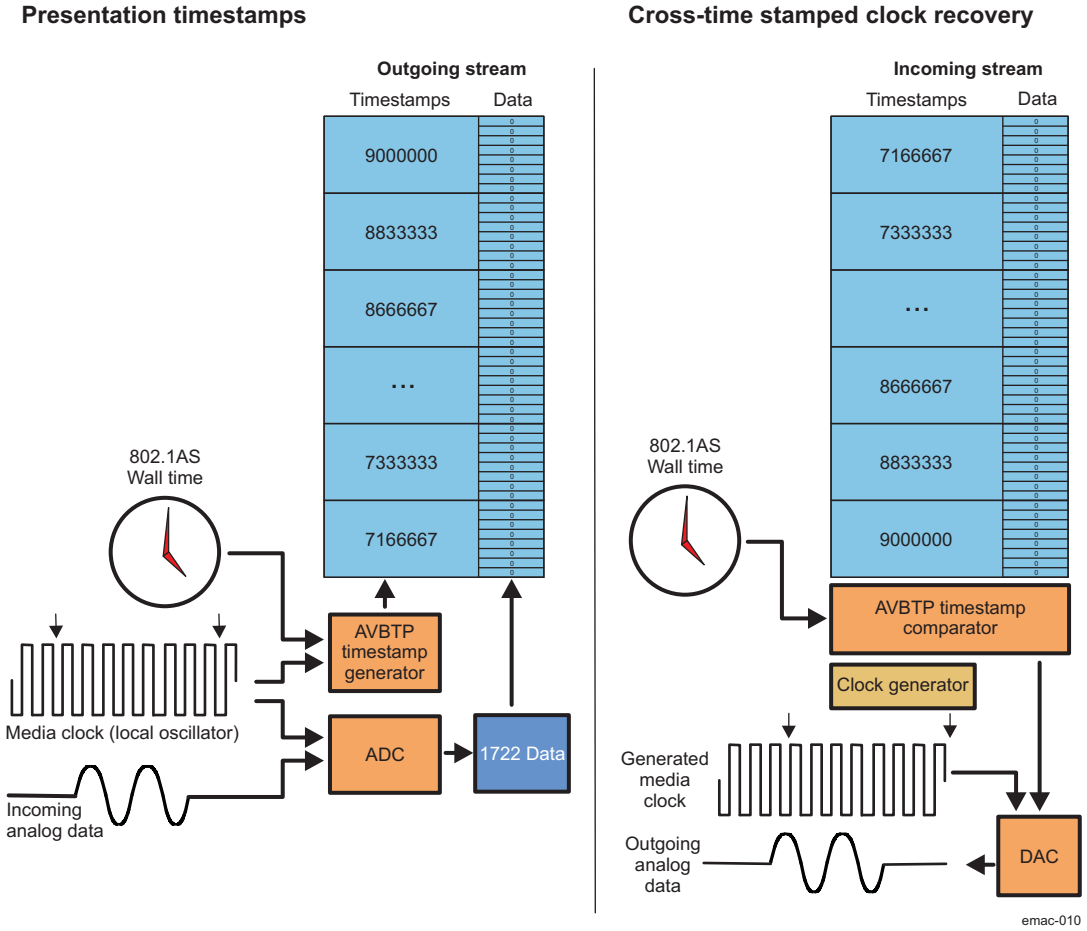
A crucial benefit of this approach is coexistence of multiple, independent media clock domains on an AVB network. Unrelated audio and video streams can simultaneously exist in the same LAN.

AVBTP assumes that AVB node media clocks are clocked by free-running oscillators. It is also assumed that the node's internal concept of wall clock time has been synchronized to the PTP Grandmaster. AVBTP media clock sources embed "AVBTP Presentation Timestamps" in AVBTP streaming packets. Figure 11-920 illustrates the relationship between PTP network time and AVBTP Presentation Timestamps.

11.13.4.4.6.4.1.2 IEEE 1733: Extends RTCP for RTP Streaming over AVB-supported Networks

This standard specifies the protocol, data encapsulations, connection management and presentation time procedures used to ensure interoperability between audio and video based end stations that use standard networking services provided by all IEEE 802 networks meeting QoS requirements for time-sensitive applications by leveraging the Real-time Transport Protocol (RTP) family of protocols and IEEE 802.1 Audio/Video Bridging (AVB) protocols.

Figure 11-920. Cross Time Stamping and Presentation Timestamps



11.13.4.4.6.4.2 IEEE 802.1Qav: "Virtual Bridged Local Area Networks: Forwarding and Queuing for Time-Sensitive Streams"

This standard allows bridges to provide guarantees for time-sensitive (that is, bounded latency and delivery variation), loss-sensitive real-time audio video (AV) data transmission (AV traffic). It specifies per priority ingress metering, priority regeneration, and timing-aware queue draining algorithms. This standard uses the timing derived from IEEE 802.1AS. Virtual Local Area Network (VLAN) tag encoded priority values are allocated, in aggregate, to segregate frames among controlled and non-controlled queues, allowing simultaneous support of both AV traffic and other bridged traffic over and between wired and wireless Local Area Networks (LANs).

Such a guarantee in bandwidth is provided by two functional entities:

- A registration protocol, which registers the service and its maximum network utilization with a device or switch (IEEE 802.1Qat: "Virtual Bridged Local Area Networks - Amendment 9: Stream Reservation Protocol (SRP)")
- A hardware bandwidth management service.
  - Receive policing and
  - Transmit rate control.

End Station Behavior

In order for an end station to successfully participate in the transmission and reception of time-sensitive streams, it is necessary for their behavior to be compatible with the operation of the forwarding and queuing mechanisms employed in bridges.

The requirements for end stations that participate as "talkers" i.e., sources of time-sensitive streams are different from the requirements that apply to "listeners", the destination station(s) for the streams.

Talker Behavior

In order for Talker-originated data streams to make use of the credit-based shaper behavior in Bridges, it is a requirement for a Talker to use the priorities that the Bridges in the network recognize as being associated with SR classes exclusively for transmitting stream data.

It is also necessary for the Talker and the Bridges in the path to the Listener(s), to have a common view of the bandwidth required in order to transmit the Talker's streams, and for that bandwidth to be reserved along the path to the Listener(s). This latter requirement can be met by means of stream reservation mechanisms, such as defined in SRP, or by other management means.

End stations that are Talkers shall exhibit transmission behavior for frames that are part of "time-sensitive streams" that is consistent with the operation of the credit-based shaper algorithm, both in terms of the way they transmit frames that are part of an individual data stream, and in terms of the way they transmit stream data frames from a Port.

In effect, the queuing model for a Talker Port (and a Listener port), and for given priorities, can be considered to look like Figure 11-921.

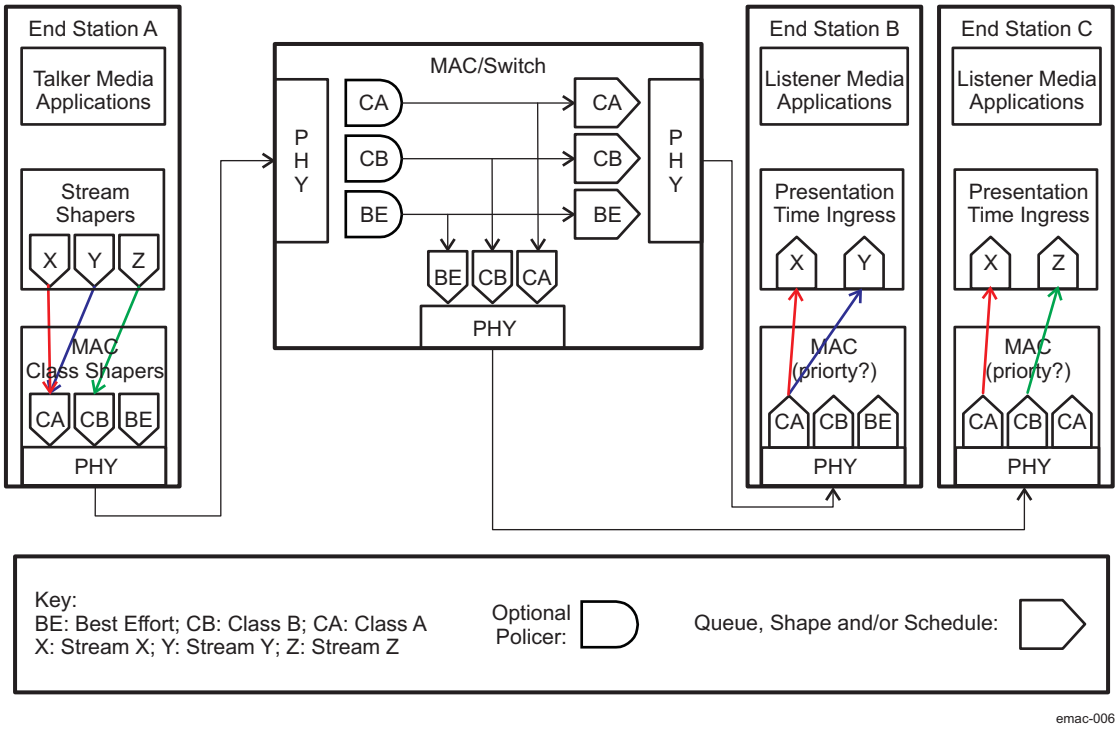
Listener Behavior

The primary requirement for a listener station is that it is capable of buffering the amount of data that could be transmitted for a stream during a time period equivalent to the accumulated maximum jitter that could be experienced by stream data frames in transmission between Talker and Listener.

From the point of view of the specification of the forwarding and queuing requirements for time-sensitive streams, it is assumed that the listener will assess the buffering required for a stream as part of the stream bandwidth reservation mechanisms employed by the implementation.

The credit-based shaper's operation details are beyond the scope of this document.

Figure 11-921. AV Stream Queuing/Policing



11.13.4.4.6.4.2.1 Configuring the Device for 802.1Qav Operation:

There is no dedicated register-set to be configured for the time-sensitive stream handling. The list of functional features of CPSW_2U that will have to be configured are:

- DESCRIPTORS and CHANNEL CONFIGURATIONS:
  - CPPI TX and RX descriptors
  - VLAN and Priority tags

Table 11-2026. Example of TX Configuration

TX DMA CHANNEL	Packet Priority	Switch Queue Priority
7	7	3
6	5	2
5	3	1
4	1	0

Table 11-2027. Example of RX Configuration

RX DMA CHANNEL	Packet Priority	Switch Queue Priority
0	7	0
0	5	0
0	3	0
0	1	0

- ALE Configuration:
  - ALE in VLAN-ware mode, Non-ALE in bypass mode.

11.13.4.4.6.5 Ethernet MAC Sliver (CPGMAC_SL)

The CPGMAC_SL peripheral is compliant to the IEEE Std 802.3 Specification. Half-duplex mode is supported in 10/100 Mbps mode, but not in 1000 Mbps (gigabit) mode.

Features:

- Synchronous 10/100/1000 Mbit operation
- G/MII Interface
- Hardware Error handling including CRC
- Full-Duplex Gigabit operation (half-duplex gigabit is not supported)
- EtherStats and 802.3Stats RMON statistics gathering support for external statistics collection module
- Transmit CRC generation selectable on a per channel basis
- Emulation Support
- VLAN Aware Mode Support
- Hardware flow control
- Programmable Inter Packet Gap (IPG).

11.13.4.4.6.5.1 G/MII Media Independent Interface

The following sections cover operation of the Media Independent Interface in 10/100/1000 Mbps modes. An IEEE 802.3 compliant Ethernet MAC controls the interface.

11.13.4.4.6.5.1.1 Data Reception

11.13.4.4.6.5.1.2 Receive Control

Data received from the PHY is interpreted and output. Interpretation involves detection and removal of the preamble and start of frame delimiter, extraction of the address and frame length, data handling, error checking and reporting, cyclic redundancy checking (CRC), and statistics control signal generation.

11.13.4.4.6.5.1.3 Receive Inter-Frame Interval

The 802.3 required inter-packet gap (IPG) is 24 G/MII clocks (96 bit times) for 10/100 Mbit modes, and 12 G/MII clocks (96 bit times) for 1000 Mbit mode. However, the MAC can tolerate a reduced IPG (2 G/MII clocks in 10/100 mode and 5 G/MII clocks in 1000 mode) with a correct preamble and start frame delimiter.

This interval between frames must comprise (in the following order):

- An Inter-Packet Gap (IPG).
- A seven octet preamble (all octets 0x55).
- A one octet start frame delimiter (0x5D).

11.13.4.4.6.5.1.4 Data Transmission

The Gigabit Ethernet Mac Sliver (G/MII) passes data to the PHY when enabled. Data is synchronized to the transmit clock rate. The smallest frame that can be sent is two bytes of data with four bytes of CRC (6 byte frame).

11.13.4.4.6.5.1.5 Transmit Control

A jam sequence is output if a collision is detected on a transmit packet. If the collision was late (after the first 64 bytes have been transmitted) the collision is ignored. If the collision is not late, the controller will back off before retrying the frame transmission. When operating in full duplex mode the carrier sense (CRS) and collision sensing modes are disabled.

11.13.4.4.6.5.1.6 CRC Insertion

The MAC generates and appends a 32-bit Ethernet CRC onto the transmitted data, if the transmit packet header PASS_CRC bit is 0. For the CPMAC_SL generated CRC case, a CRC at the end of the input packet data is not allowed.

If the header word PASS_CRC bit is set, then the last four bytes of the TX data are transmitted as the frame CRC. The four CRC data bytes should be the last four bytes of the frame and should be included in the packet byte count value. The MAC performs no error checking on the outgoing CRC when the PASS_CRC bit is set.

11.13.4.4.6.5.1.7 MTXER

The MTXER signal is only used for EEE. If an underflow condition occurs on a transmitted frame, the frame CRC will be inverted to indicate the error to the network. Underflow is a hardware error.

11.13.4.4.6.5.1.8 Adaptive Performance Optimization (APO)

The Ethernet MAC port incorporates Adaptive Performance Optimization (APO) logic that may be enabled by setting the TX_PACE bit in the CPSW_P1_MAC_CONTROL register. Transmission pacing to enhance performance is enabled when set. Adaptive performance pacing introduces delays into the normal transmission of frames, delaying transmission attempts between stations, reducing the probability of collisions occurring during heavy traffic (as indicated by frame deferrals and collisions) thereby increasing the chance of successful transmission.

When a frame is deferred, suffers a single collision, multiple collisions or excessive collisions, the pacing counter is loaded with an initial value of 31. When a frame is transmitted successfully (without experiencing a deferral, single collision, multiple collision or excessive collision) the pacing counter is decremented by one, down to zero.

With pacing enabled, a new frame is permitted to immediately (after one IPG) attempt transmission only if the pacing counter is zero. If the pacing counter is non zero, the frame is delayed by the pacing delay, a delay of approximately four inter-packet gap delays. APO only affects the IPG preceding the first attempt at transmitting a frame. It does not affect the back-off algorithm for re-transmitted frames.

11.13.4.4.6.5.1.9 Inter-Packet-Gap Enforcement

The measurement reference for the IPG of 96 bit times is changed depending on frame traffic conditions. If a frame is successfully transmitted without collision, and MCRS is de-asserted within approximately 48 bit times of MTXEN being de-asserted, then 96 bit times is measured from MTXEN. If the frame suffered a collision, or if MCRS is not de-asserted until more than approximately 48 bit times after MTXEN is de-asserted, then 96 bit times (approximately, but not less) is measured from MCRS.

11.13.4.4.6.5.1.10 Back Off

The Gigabit Ethernet Mac Sliver (G/MII) implements the 802.3 binary exponential back-off algorithm.

11.13.4.4.6.5.1.11 Programmable Transmit Inter-Packet Gap

The transmit inter-packet gap (IPG) is programmable through the CPSW_P1_MAC_TX_GAP register. The default value is decimal 12. The transmit IPG may be increased to the maximum value of 1FFh. Increasing the IPG is not compatible with transmit pacing. The short gap feature will override the increased gap value, so the short gap feature may not be compatible with an increased IPG.

11.13.4.4.6.5.1.12 Speed, Duplex and Pause Frame Support Negotiation

The CPMAC_SL can operate in half duplex or full duplex in 10/100 Mbit modes, and can operate in full duplex only in 1000 Mbit mode. Pause frame support is included in 10/100/1000 Mbit modes as configured by the host.

11.13.4.4.6.5.2 RMII Interface

The CPRMII peripheral is compliant to the RMII specification document.

11.13.4.4.6.5.2.1 Features

- Source Synchronous 10/100 Mbit operation
- Full and Half Duplex support

11.13.4.4.6.5.2.2 RMII Receive (RX)

The CPRMII receive (RX) interface converts the input data from the external RMII PHY (or switch) into the required MII (CPGMAC) signals. The carrier sense and collision signals are determined from the RMII input data stream and transmit inputs as defined in the RMII specification.

An asserted RMRXER on any di-bit in the received packet will cause an MRXER assertion to the CPGMAC during the packet. In 10Mbps mode, the error is not required to be duplicated on 10 successive clocks. Any di-bit which has an asserted RMII_RXER during any of the 10 replications of the data will cause the error to be propagated.

Any received packet that ends with an improper nibble boundary aligned RMCRSDV toggle will issue an MRXER during the packet to the CPGMAC. Also, a change in speed or duplex mode during packet operations will cause packet corruption.

The CPRMII can accept receive packets with shortened preambles, but 0x55 followed by a 0x5D is the shortest preamble that will be recognized (1 preamble byte with the start of frame byte). At least one byte of preamble with the start of frame indicator is required to begin a packet. An asserted RMCRSDV without at least a single correct preamble byte followed by the start of frame indicator will be ignored.

11.13.4.4.6.5.2.3 RMII Transmit (TX)

The CPRMII transmit (TX) interface converts the EMAC MII input data into the RMII transmit format. The data is then output to the external RMII PHY.

The EMAC does not source the transmit error (MII TXERR) signal. Any transmit frame from the CPGMAC with an error (underrun) will be indicated as an error by an error CRC. Transmit error is assumed to be de-asserted at all times and is not an input into the CPRMII module. Zeroes are output on RMTXD[1:0] for each clock that RMTXEN is de-asserted.

11.13.4.4.6.5.3 RGMII Interface

The CPRGMII peripheral is compliant to the RGMII specification document.

11.13.4.4.6.5.3.1 Features

- Supports 1000/100/10 Mbps speed
- Internal TXC delay on transmit (enabled/disabled in BOOT_CFG)
- MII mode is not supported

11.13.4.4.6.5.3.2 RGMII Receive (RX)

The CPRGMII receive (RX) interface converts the source synchronous DDR input data from the external RGMII PHY into the required G/MII (CPGMAC) signals.

11.13.4.4.6.5.3.3 RGMII Transmit (TX)

The CPRGMII transmit (TX) interface converts the CPGMAC G/MII input data into the DDR RGMII format. The DDR data is then output to the external PHY.

The CPGMAC does not source the transmit error (TXERR) signal. Any transmit frame from the CPGMAC with an error (underrun) will be indicated as an error by an error CRC. Transmit error is assumed to be de-asserted at all times and is not an input into the CPRGMII module.

The TXD[7:0] data bus uses only the lower nibble. The CPRGMII will output the lower nibble twice in 10/100 mode to avoid unnecessary signal switching.

Packets will be precluded from transmission through the CPRGMII module for 4096 transmit clocks after the rising edge of RGMII_LINK. Packet transmission will begin on the first TX_CTL rising edge after the 4096 transmit clock count has expired.

11.13.4.4.6.5.4 Frame Classification

Received frames are proper (good) frames if they are between 64 and RX_MAXLEN in length (inclusive) and contain no errors (code/align/CRC).

Received frames are long frames if their frame count exceeds the value in the CPSW_P0_RX_MAXLEN/CPSW_P1_RX_MAXLEN register. The register reset (default) value is 1518 (decimal). Long received frames are either oversized or jabber frames. Long frames with no errors are oversized frames. Long frames with CRC, code, or alignment errors are jabber frames.

Received frames are short frames if their frame count is less than 64 bytes. Short frames that contain no errors are undersized frames. Short frames with CRC, code, or alignment errors are fragment frames.

A received long packet will always contain RX_MAXLEN number of bytes transferred to memory (if RX_CEF_EN = 1). An example with RX_MAXLEN = 1518 is:

- If the frame length is 1518, then the packet is not a long packet and there will be 1518 bytes transferred to memory.

- If the frame length is 1519, there will be 1518 bytes transferred to memory. The last three bytes will be the first three CRC bytes.
- If the frame length is 1520, there will be 1518 bytes transferred to memory. The last two bytes will be the first two CRC bytes.
- If the frame length is 1521, there will be 1518 bytes transferred to memory. The last byte will be the first CRC byte.

If the frame length is 1522, there will be 1518 bytes transferred to memory. The last byte will be the last data byte.

11.13.4.4.6.6 Embedded Memories

Table 11-2028. Embedded Memories

Memory Type Description	Number of Instances	
Single-port 512-word × 256-bit RAM	1	(CPPI)

11.13.4.4.6.7 Flow Control

There are two types of switch flow control: CPPI port flow control and Ethernet port flow control. The CPPI and Ethernet port naming conventions for data flow into and out of the switch are reversed. For the CPPI port (port 0), transmit operations move packets from external memory into the switch and then out to either or both Ethernet transmit ports (ports 1 and 2). CPPI receive operations move packets that were received on either or both Ethernet receive ports to external memory.

11.13.4.4.6.7.1 CPPI Port Flow Control

The CPPI FIFO port has flow control for receive operations (packet ingress). CPPI receive flow control is always enabled and is initiated when triggered. CPPI Receive flow control is accomplished through pushback. The pushback will cause the CPPI port logical receive FIFO to fill up with packet data. Flow control is accomplished with a de-asserted RXST_THREAD_SREADY on the receive packet streaming interface.

11.13.4.4.6.7.2 Ethernet Port Flow Control

The Ethernet ports have flow control available for transmit and receive. Transmit flow control stops the Ethernet port from transmitting packets to the wire (switch egress) in response to a received pause frame. Transmit flow control does not depend on FIFO usage.

The ethernet ports have flow control available for receive operations (packet ingress). Ethernet port receive flow control is initiated when enabled and triggered. Packets received on an ethernet port can be sent to the other ethernet port or the CPPI port (or both). Each destination port can trigger the receive ethernet port flow control. An ethernet destination port triggers another ethernet receive flow control when the destination port is full.

When a packet is received on an ethernet port interface with enabled flow control the below occurs:

- The packet will be sent to all ports that currently have room to take the entire packet.
- The packet will be retried until successful to all ports that indicate they don't have room for the packet.

The flow control trigger to the CPGMAC_SL will be asserted until the packet has been sent, and there is room in the logical receive FIFO for packet runout from another flow control trigger (RX_PKT_CNT = 0). Ethernet port receive flow control is disabled by default on reset. Ethernet port receive flow control requires that the RX_FLOW_EN bit in CPSW_P1_MAC_CONTROL be set to 1. When receive flow control is enabled on a port, the port's associated FIFO block allocation must be adjusted. The port RX allocation must increase from the default three blocks to accommodate the flow control runout. A corresponding decrease in the TX block allocation is required. If a sending port ignores a pause frame then packets may overrun on receive (and be dropped) but will not be dropped on transmit. If flow control is disabled for G/MII ports, then any packets that are dropped are dropped on transmit and not on receive.

11.13.4.4.6.7.2.1 Receive Flow Control

When enabled and triggered, receive flow control is initiated to limit the CPGMAC_SL from further frame reception. Half-duplex mode receive flow control is collision based while full duplex mode issues 802.3X pause frames. In either case, receive flow control prevents frame reception by issuing the flow control appropriate for the current mode of operation. Receive flow control is enabled by the RX_FLOW_EN bit in the CPSW_P1_MAC_CONTROL register. Receive flow control is triggered (when enabled) when the RX_FLOW_TRIGGER input is asserted. The CPGMAC_SL is configured for collision or IEEE 802.3X flow control via the FULLDUPLEX bit in the CPSW_P1_MAC_CONTROL register.

11.13.4.4.6.7.2.2 Collision Based Receive Buffer Flow Control

Collision-based receive buffer flow control provides a means of preventing frame reception when the port is operating in half-duplex mode (FULLDUPLEX is cleared in CPSW_P1_MAC_CONTROL). When receive flow control is enabled and triggered, the port will generate collisions for received frames. The jam sequence transmitted will be the twelve byte sequence C3.C3.C3.C3.C3.C3.C3.C3.C3.C3.C3.C3 (hex). The jam sequence will begin no later than approximately as the source address starts to be received. Note that these forced collisions will not be limited to a maximum of 16 consecutive collisions, and are independent of the normal back-off algorithm. Receive flow control does not depend on the value of the incoming frame destination address. A collision will be generated for any incoming packet, regardless of the destination address.

11.13.4.4.6.7.2.3 IEEE 802.3X Based Receive Flow Control

IEEE 802.3x based receive flow control provides a means of preventing frame reception when the port is operating in full-duplex mode (FULLDUPLEX is set in CPSW_P1_MAC_CONTROL). When receive flow control is enabled and triggered, the port will transmit a pause frame to request that the sending station stop transmitting for the period indicated within the transmitted pause frame.

The CPGMAC_SL will transmit a pause frame to the reserved multicast address at the first available opportunity (immediately if currently idle, or following the completion of the frame currently being transmitted). The pause frame will contain the maximum possible value for the pause time (FFFFh). The MAC will count the receive pause frame time (decrements FF00h down to 0) and retransmit an outgoing pause frame if the count reaches zero. When the flow control request is removed, the MAC will transmit a pause frame with a zero pause time to cancel the pause request.

Note that transmitted pause frames are only a request to the other end station to stop transmitting. Frames that are received during the pause interval will be received normally (provided the RX FIFO is not full).

Pause frames will be transmitted if enabled and triggered regardless of whether or not the port is observing the pause time period from an incoming pause frame.

The CPGMAC_SL will transmit pause frames as:

- The 48-bit reserved multicast destination address 01.80.C2.00.00.01.
- The 48-bit source address - from SL_SA[47:0] input.
- The 16-bit length/type field containing the value 88.08
- The 16-bit pause opcode equal to 00.01
- The 16-bit pause time value FF.FF. A pause-quantum is 512 bit-times. Pause frames sent to cancel a pause request will have a pause time value of 00.00.
- Zero padding to 64-byte data length (The CPGMAC_SL will transmit only 64 byte pause frames).
- The 32-bit frame-check sequence (CRC word).

All quantities above are hexadecimal and are transmitted most-significant byte first. The least-significant bit is transferred first in each byte.

If RX_FLOW_EN is cleared to 0 while the pause time is nonzero, then the pause time will be cleared to 0 and a zero count pause frame will be sent.

11.13.4.4.6.7.2.4 Transmit Flow Control

Incoming pause frames are acted upon, when enabled, to prevent the CPGMAC_SL from transmitting any further frames. Incoming pause frames are only acted upon when the FULLDUPLEX and TX_FLOW_EN bits in the CPSW_P1_MAC_CONTROL register are set. Pause frames are not acted upon in half-duplex mode. Pause frame action will be taken if enabled, but normally the frame will be filtered and not transferred to memory. MAC control frames will be transferred to memory if the RX_CMF_EN (copy MAC frames) bit in the CPSW_P1_MAC_CONTROL register is set. The TX_FLOW_EN and FULLDUPLEX bits effect whether or not MAC control frames are acted upon, but they have no effect upon whether or not MAC control frames are transferred to memory or filtered.

Pause frames are a subset of MAC Control Frames with an opcode field = 0001h. Incoming pause frames will only be acted upon by the port if:

- TX_FLOW_EN is set in CPSW_P1_MAC_CONTROL register, and
- the frame's length is 64 to SL_RX_MAXLEN bytes inclusive, and
- the frame contains no CRC error or align/code errors.

The pause time value from valid frames will be extracted from the two bytes following the opcode. The pause time will be loaded into the port's transmit pause timer and the transmit pause time period will begin.

If a valid pause frame is received during the transmit pause time period of a previous transmit pause frame then:

- if the destination address is not equal to the reserved multicast address or any enabled or disabled unicast address, then the transmit pause timer will immediately expire, or
- if the new pause time value is zero then the transmit pause timer will immediately expire, else
- the port transmit pause timer will immediately be set to the new pause frame pause time value. (Any remaining pause time from the previous pause frame will be discarded).

If TX_FLOW_EN in CPSW_P1_MAC_CONTROL register is cleared, then the pause-timer will immediately expire.

The port will not start the transmission of a new data frame any sooner than 512-bit times after a pause frame with a non-zero pause time has finished being received (MRXDV going inactive). No transmission will begin until the pause timer has expired (the port may transmit pause frames in order to initiate outgoing flow control). Any frame already in transmission when a pause frame is received will be completed and unaffected.

Incoming pause frames consist of the below:

- A 48-bit destination address equal to:
  - The reserved multicast destination address 01.80.C2.00.00.01
  - , or the CPGMAC_SL SL_SA [47:0] input.
- The 48-bit source address of the transmitting device.
- The 16-bit length/type field containing the value 88.08
- The 16-bit pause opcode equal to 00.01
- The 16-bit pause_time. A pause-quantum is 512 bit-times.
- Padding to 64-byte data length.
- The 32-bit frame-check sequence (CRC word).

All quantities above are hexadecimal and are transmitted most-significant byte first. The least-significant bit is transferred first in each byte.

The padding is required to make up the frame to a minimum of 64 bytes. The standard allows pause frames longer than 64 bytes to be discarded or interpreted as valid pause frames. The CPGMAC_SL will recognize any pause frame between 64 bytes and RX_MAXLEN bytes in length.

11.13.4.4.6.8 Latency

When CPSW_2U is configured as a store and forward switch, the switch latency is defined as the amount of time between the end of packet reception of the received packet to the start of the output packet transmit.

Table 11-2029. Switch Latency

Mode	Latency
Gig (1000)	880 ns
100	1.3 µs
10	6.5 µs

11.13.4.4.6.9 Emulation Control

The emulation control input (EMUSUSP) and submodule emulation control registers allow CPSW_2U operation to be completely or partially suspended. There are two CPSW_2U submodules that contain emulation control registers (CPGMAC_SL and CPDMA). The submodule emulation control registers must be accessed to facilitate CPSW_2U emulation control. The CPSW_2U module enters the emulation suspend state if all three submodules are configured for emulation suspend and the emulation suspend input is asserted. A partial emulation suspend state is entered if one or two submodules is configured for emulation suspend and the emulation suspend input is asserted. Emulation suspend occurs at packet boundaries. The emulation control feature is implemented for compatibility with other peripherals.

CPGMAC_SL Emulation Control

The emulation control input (TBEMUSUP) and register bits (SOFT and FREE bits in the CPSW_P1_MAC_EMCONTROL register) allow CPGMAC_SL operation to be suspended. When the emulation suspend state is entered, the CPGMAC_SL will stop processing receive and transmit frames at the next frame boundary. Any frame currently in reception or transmission will be completed normally without suspension. For receive, frames that are detected by the CPGMAC_SL after the suspend state is entered are ignored. Emulation control is implemented for compatibility with other peripherals.

CDMA Emulation Control

The emulation control input (TBEMUSUP) and register bits (SOFT and FREE bits in the CDMA_EMULATION_CONTROL_REG register) allow CDMA operation to be suspended. When the emulation suspend state is entered, the CDMA will stop processing receive and transmit frames at the next frame boundary. Any frame currently in reception or transmission will be completed normally without suspension. For transmission, any complete or partial frame in the tx cell FIFO will be transmitted. For receive, frames that are detected by the PDMA after the suspend state is entered are ignored. No statistics will be kept for ignored frames. Emulation control is implemented for compatibility with other peripherals

Table 11-2030 shows the operations of the emulation control input and register bits.

Table 11-2030. Emulation Control Input

EMUSUSP	SOFT	FREE	Description
0	X	X	Normal Operation
1	0	0	Normal Operation
1	1	0	Emulation Suspend
1	X	1	Normal Operation

11.13.4.4.6.10 Energy Efficient Ethernet Support (802.3az)

Energy Efficient Ethernet (EEE) allows the PSC to turn off the module clock during inactive periods as determined by network and host traffic. The module can then be awakened by host queued transmit packet(s) or by a port's external Ethernet PHY. EEE operations are configured as shown below:

1. The 12-bit EEE clock pre-scale value is written to the CPSW_EEE_PRESCALE register. The pre-

scaler is used to clock all EEE-related counters

2. The port Idle to LPI count values (Px_IDLE2LPI) are written with the desired values
3. The port LPI to Wake count values (Px_LPI2WAKE) are written with the desired values
4. The EEE_EN bit is set in the switch CPSW_CONTROL register
5. Set the SS_CONTROL[8] SS_EEE_EN bit to enable energy efficient operations at subsystem level.

EEE operation can begin after configuration. The host allows (through PSC) the CPSW_2U to enter a low power state by asserting the EEE_CLKSTOP_REQ signal. There are no requirements on host queues or traffic in order for the host to assert or de-assert EEE_CLKSTOP_REQ to the CPSW_2U.

Each ethernet port has a transmit and a receive LPI (low power indicate) state. The receive LPI state is entered when the port's corresponding PHY indicates the LPI state via the CPSW_2U G/MII interface. The PHY indicates LPI by asserting MRXER with a MRXD[7:0] value of 0x01 while MRXDV is deasserted (inter-packet gap). The Ethernet transmit port indicates LPI after the Px_IDLE2LPI value has been counted (the transmit port has gone idle for the configured amount of time). If another packet is received for transmit during the count then the count is restarted. When the transmit port has been idle for the Idle to LPI time, the transmit port enters the LPI state and indicates LPI to the associated PHY. The LPI is indicated to the external PHY by an asserted MTXER with a MTXD[7:0] while MTXEN is deasserted (inter-packet gap). The CPDMA LPI state includes transmit and receive. The CPDMA LPI state is entered when the CPDMA transmit and receive have both been idle for the Idle to LPI time (P0_IDLE2LPI). The Idle to LPI time value for all ports must be large relative to the switch latency to ensure that the count is not able to complete between successive packets.

NOTE: The procedure above is described for the G/MII interfaces at the CPSW_2U boundary. External PHY signaling has the following conditions:

- G/MII interface is used only in MII mode. Therefore, only MRXD[3:0] and MTXD[3:0] are pinned out
- RGMII is a DDR interface. TXEN and TXER are the sampled values of TX_CTL at the rising and the falling TXC_CLK edges, respectively. RXDV and RXER are the sampled values of RX_CTL at the rising and the falling RXC_CLK edges, respectively
- In RMII mode, EEE is not supported.

When all transmit and receive ports are in the LPI state (CPSW LPI state), the EEE_CLKSTOP_ACK signal is asserted, and the PSC is allowed to stop the module clock. When EEE_CLKSTOP_ACK is asserted, the clock may be turned on and off as desired by the host. The host is allowed to restart the clock, perform slave read/write operations to the CPSW_2U memory address space, and then turn off the clock again while EEE_CLKSTOP_ACK is asserted.

The software can remove and disable from re-entering the CPSW LPI state by restarting the module clock and then de-asserting EEE_CLKSTOP_REQ. The host may queue transmit packets at any time including without regard to the CPSW_2U LPI state (the clock must be restarted in order to write the CPSW_2U slave address space as described above). Host writes to transmit head descriptor pointers will cause the CLKSTOP_WAKEUP signal to be asserted if the CPSW_2U is in the low power state (if CLKSTOP_ACK is asserted).

The external Ethernet PHY's can also wakeup the PSC by removing the Ethernet receive LPI indication. If the module is in the CPSW Idle state with CLKSTOP_ACK asserted and the receive LPI indication is removed, the CLKSTOP_WAKEUP signal will be asynchronously asserted. On wakeup, the PSC restarts the clock and de-assert the EEE_CLKSTOP_REQ signal. The CLKSTOP_WAKEUP signal will be synchronously deasserted with CLKSTOP_ACK. Upon the deassertion of CLKSTOP_REQ, the Ethernet ports will count the Px_LPI2WAKE time for each port at which time the port is available for transmit. Upon the de-assertion of CLKSTOP_REQ, the CPDMA transmit will count the P0_LPI2WAKE count at which time the CPDMA will begin to send any packets that the host has queued (switch ingress). The wait time on CPDMA transmit is included to preclude the host from filling up the Ethernet port transmit FIFO's while the Ethernet ports are in the LPI to wake time. There is no LPI to wake time on CPDMA receive (switch egress).

11.13.4.4.6.11 CPSW_2U Network Statistics

The CPSW_2U has a set of statistics that record events associated with frame traffic on selected switch ports. The statistics values are cleared to zero 38 clocks after the rising edge of GMAC_RST. When port enable (P0_STAT_EN or/and P1_STAT_EN) bits in the CPSW_STAT_PORT_EN register are set, all statistics registers are write to decrement. The value written will be subtracted from the register value with the result being stored in the register. If a value greater than the statistics value is written, then zero will be written to the register (writing 0xFFFF FFFF clears a statistics location). When all port enable bits are cleared to zero, all statistics registers are read/write (normal write direct, so writing 0x0000 0000 clears a statistics location). All write accesses must be 32-bit accesses.

The statistics interrupt (STAT_PEND) will be issued if enabled when any statistics value is greater than or equal to 0x8000 0000. The statistics interrupt is removed by writing to decrement any statistics value greater than 0x8000 0000. The statistics are mapped into internal memory space and are 32-bits wide. All statistics rollover from 0xFFFF FFFF to 0x0000 0000.

NOTE: Statistics start at physical address 0423 A000h for CPPI port 0.

Statistics start at physical address 0423 A200h for Ethernet port 1.

Table 11-2031 and Table 11-2032 summarize network statistics.

11.13.4.4.6.11.1 Rx-only Statistics Descriptions

11.13.4.4.6.11.1.1 Good Rx Frames (Offset = 0h)

All ports

The total number of good frames received on the port. A good frame is defined to be:

- Any data or MAC control frame which matched a unicast, broadcast or multicast address, or matched due to promiscuous mode
- Had a length of 64 to RX_MAXLEN bytes inclusive
- Had no CRC error, alignment error or code error.

See the Section 11.13.4.4.6.11.1.6, *Rx Align/Code Errors* and Section 11.13.4.4.6.11.1.5, *Rx CRC errors* statistic descriptions for definitions of alignment, code and CRC errors.

Overruns have no effect upon this statistic.

11.13.4.4.6.11.1.2 Broadcast Rx Frames (Offset = 4h)

All ports

The total number of good broadcast frames received on the port. A good broadcast frame is defined to be:

- Any data or MAC control frame which was destined for address FF.FF.FF.FF.FF.FF
- Had a length of 64 to RX_MAXLEN bytes inclusive
- Had no CRC error, alignment error or code error.

See the Section 11.13.4.4.6.11.1.6, *Rx Align/Code Errors* and Section 11.13.4.4.6.11.1.5, *Rx CRC errors* statistic descriptions for definitions of alignment, code and CRC errors.

Overruns have no effect upon this statistic.

11.13.4.4.6.11.1.3 Multicast Rx Frames (Offset = 8h)

All ports

The total number of good multicast frames received on the port. A good multicast frame is defined to be:

- Any data or MAC control frame which was destined for any multicast address other than FF.FF.FF.FF.FF.FF
- Had a length of 64 to RX_MAXLEN bytes inclusive

- Had no CRC error, alignment error or code error

See the Section 11.13.4.4.6.11.1.6, *Rx Align/Code Errors* and Section 11.13.4.4.6.11.1.5, *Rx CRC errors* statistic descriptions for definitions of alignment, code and CRC errors.

Overruns have no effect upon this statistic.

11.13.4.4.6.11.1.4 Pause Rx Frames (Offset = Ch)

Ethernet port 1

The total number of IEEE 802.3X pause frames received by the port (whether acted upon or not). Such a frame:

- Contained any unicast, broadcast, or multicast address
- Contained the length/type field value 88.08 (hex) and the opcode 0x0001
- Was of length 64 to RX_MAXLEN bytes inclusive
- Had no CRC error, alignment error or code error
- Pause-frames had been enabled on the port (TX_FLOW_EN = 1).

The port could have been in either half or full-duplex mode.

See the Section 11.13.4.4.6.11.1.6, *Rx Align/Code Errors* and Section 11.13.4.4.6.11.1.5, *Rx CRC errors* statistic descriptions for definitions of alignment, code and CRC errors.

Overruns have no effect upon this statistic.

11.13.4.4.6.11.1.5 Rx CRC Errors (Offset = 10h)

All ports

The total number of frames received on the port that experienced a CRC error. Such a frame:

- Was any data or MAC control frame which matched a unicast, broadcast or multicast address, or matched due to promiscuous mode
- Was of length 64 to RX_MAXLEN bytes inclusive
- Had no code/align error
- Had a CRC error

Overruns have no effect upon this statistic.

A CRC error is defined to be:

- A frame containing an even number of nibbles, and
- Failing the Frame Check Sequence test.

11.13.4.4.6.11.1.6 Rx Align/Code Errors (Offset = 14h)

Ethernet port 1

The total number of frames received on the port that experienced an alignment error or code error. Such a frame:

- Was any data or MAC control frame which matched a unicast, broadcast or multicast address, or matched due to promiscuous mode
- Was of length 64 to RX_MAXLEN bytes inclusive
- Had either an alignment error, or a code error.

Overruns have no effect upon this statistic.

An alignment error is defined to be:

- A frame containing an odd number of nibbles
- Failing the Frame Check Sequence test if the final nibble is ignored

A code error is defined to be a frame which has been discarded because the port's MRXER pin driven with a one for at least one bit-time's duration at any point during the frame's reception.

NOTE: RFC 1757 etherStatsCRCAlignErrors Ref. 1.5 can be calculated by summing Rx Align/Code Errors and Rx CRC errors.

11.13.4.4.6.11.1.7 Oversize Rx Frames (Offset = 18h)

All ports

The total number of oversized frames received on the port. An oversized frame is defined to be:

- Was any data or MAC control frame which matched a unicast, broadcast or multicast address, or matched due to promiscuous mode
- Was greater than RX_MAXLEN in bytes
- Had no CRC error, alignment error, or code error.

See the Section 11.13.4.4.6.11.1.6, *Rx Align/Code Errors* and Section 11.13.4.4.6.11.1.5, *Rx CRC errors* statistic descriptions for definitions of alignment, code and CRC errors.

Overruns have no effect upon this statistic.

11.13.4.4.6.11.1.8 Rx Jabbers (Offset = 1Ch)

All ports

The total number of jabber frames received on the port. A jabber frame:

- Was any data or MAC control frame which matched a unicast, broadcast or multicast address, or matched due to promiscuous mode
- Was greater than RX_MAXLEN in bytes
- Had no CRC error, alignment error or code error

See the Section 11.13.4.4.6.11.1.6, *Rx Align/Code Errors* and Section 11.13.4.4.6.11.1.5, *Rx CRC errors* statistic descriptions for definitions of alignment, code and CRC errors.

Overruns have no effect upon this statistic.

11.13.4.4.6.11.1.9 Undersize (Short) Rx Frames (Offset = 20h)

All ports

The total number of undersized frames received on the port. An undersized frame is defined to be:

- Was any data frame which matched a unicast, broadcast or multicast address, or matched due to promiscuous mode
- Was less than 64 bytes
- Had no CRC error, alignment error, or code error

See the Section 11.13.4.4.6.11.1.6, *Rx Align/Code Errors* and Section 11.13.4.4.6.11.1.5, *Rx CRC errors* statistic descriptions for definitions of alignment, code and CRC errors.

Overruns have no effect upon this statistic.

11.13.4.4.6.11.1.10 Rx Fragments (Offset = 24h)

Ethernet port 1

The total number of frame fragments received on the port. A frame fragment is defined to be:

- Any data frame (address matching does not matter)
- Less than 64 bytes long
- Having a CRC error, an alignment error, or a code error
- Not the result of a collision caused by half-duplex, collision-based flow control

See the Section 11.13.4.4.6.11.1.6, *Rx Align/Code Errors* and Section 11.13.4.4.6.11.1.5, *Rx CRC errors* statistic descriptions for definitions of alignment, code and CRC errors.

Overruns have no effect upon this statistic.

11.13.4.4.6.11.1.11 ALE Drop (Offset = 28h)

All ports

The total number of frames received on a port such that the destination address was not equal to the source address and the packet was not destined to the port it was received on, but the frame was not forwarded to any port (the PORT_MASK was zero).

- Any data or MAC control frame which matched a unicast, broadcast or multicast address, or matched due to promiscuous mode
- Any length (including less than 64 bytes and greater than RX_MAXLEN bytes)
- had no CRC error, alignment error, or code error
- the destination address was not equal to the source address
- the packet was not destined for the port it was receive on
- had a zero PORT_MASK

11.13.4.4.6.11.1.12 ALE Overrun Drop (Offset = 2Ch)

All ports

The total number of frames received on a port that were dropped (zero PORT_MASK) due to exceeding the maximum ALE lookup rate (Port 0 should not have ALE Overrun Drops because the ingress rate is controlled to prevent it). This statistic should be zero and when non-zero indicates a system clock issue or indicates that short packets were sent with RX_CSF_EN at a rate that exceeded the maximum lookup rate.

- Any data or MAC control frame which matched a unicast, broadcast or multicast address, or matched due to promiscuous mode
- Any length (including less than 64 bytes and greater than RX_MAXLEN bytes)
- had no CRC error, alignment error, or code error
- the maximum ALE lookup rate was exceeded so the lookup was aborted and the packet was dropped.

11.13.4.4.6.11.1.13 Rx Octets (Offset = 30h)

All ports

The total number of bytes in all good frames received on the port. A good frame is defined to be:

- Any data or MAC control frame which matched a unicast, broadcast or multicast address, or matched due to promiscuous mode
- Of length 64 to RX_MAXLEN bytes inclusive
- Had no CRC error, alignment error or code error

See the Section 11.13.4.4.6.11.1.6, *Rx Align/Code Errors* and Section 11.13.4.4.6.11.1.5, *Rx CRC errors* statistic descriptions for definitions of alignment, code and CRC errors.

Overruns have no effect upon this statistic.

11.13.4.4.6.11.1.14 Rx Bottom of FIFO Drop (Offset = 84h)

Ethernet port 1

The total number of frames received on a port that overran the port's receive FIFO and were dropped (bottom of receive FIFO). Port 0 (CPPI receive port) should not drop packets on receive because port 0 receive flow control should be enabled. The Ethernet ports will only drop packets in the receive FIFO when receive flow control is enabled and the sending port ignores sent pause frame and then overruns the receive FIFO. An overrun frame is defined to be:

- Any data or MAC control frame which matched a unicast, broadcast or multicast address, or matched due to promiscuous mode
- Any length (including less than 64 bytes and greater than RX_MAXLEN bytes)
- Was dropped on port 0 due to a lack of memory space in the receive FIFO.

NOTE: This statistic should be zero if proper flow control is being followed.

Host port 0

This statistic also counts frames dropped on port 0 that were 17 to 63 bytes (only for port 0). For Ethernet ports, the drop count for frames shorter than 63 bytes is included in the undersized or fragment count. Port 0 simply gives an indication that a packet was dropped. No other statistics are counted for frames shorter than 64 bytes.

11.13.4.4.6.11.1.15 Portmask Drop (Offset = 88h)

All ports

The total number of frames received on a port that were dropped by the ALE (the ALE did not forward the packet to any port). Port mask drop frame is defined to be:

- Any data or MAC control frame
- Any length greater than 32 bytes
- Was dropped by the ALE due to PORT_MASK=0 (was not sent to any destination port)
- The frame could have been dropped due to error or other counted reason, so it could be counted elsewhere also.

NOTE: This statistic does not count in the overall total as it includes every packet received greater than 32 bytes that had a zero PORT_MASK.

11.13.4.4.6.11.1.16 Rx Top of FIFO Drop (Offset = 8Ch)

All ports

The total number of frames received on a port that had a start-of-frame (SOF) overrun on any destination port egress (when attempting to load the packet from the top of the ingress port receive FIFO into any other port's transmit FIFO). If a multicast/broadcast packet is dropped by multiple destination ports then this statistic will increment by the number of ports that dropped the packet. Rx Top Of FIFO Drop is defined to be:

- Any data or MAC control frame which matched a unicast, broadcast or multicast address, or matched due to promiscuous mode
- Any length (including less than 64 bytes and greater than RX_MAXLEN bytes)
- had no CRC error, alignment error or code error
- had a SOF of frame overrun on another port egress.

11.13.4.4.6.11.1.17 ALE Rate Limit Drop (Offset = 90h)

All ports

The total number of frames received on a port that were dropped (zero PORT_MASK) due to receive rate limiting on this port or due to transmit rate limiting on any destination port (not sent to all expected destination ports if transmit rate limiting).

- Any data or MAC control frame which matched a unicast, broadcast or multicast address, or matched due to promiscuous mode
- Any length (including less than 64 bytes and greater than RX_MAXLEN bytes)
- had no CRC error, alignment error, or code error
- the receive rate was exceeded and the packet was dropped, or the transmit rate was exceeded to any

destination port and the packet was dropped to one or more expected destination ports (indicates that the destinations were reduced due to rate limiting).

11.13.4.4.6.11.1.18 ALE VLAN Ingress Check Drop (Offset = 94h)

All ports

The total number of frames received on a port that were dropped (zero PORT_MASK) due to VLAN ingress check failure.

- Any data or MAC control frame which matched a unicast, broadcast or multicast address, or matched due to promiscuous mode
- Any length (including less than 64 bytes and greater than RX_MAXLEN bytes)
- had no CRC error, alignment error, or code error
- the VLAN ID ingress check failed (the receive port was not in the group)
- The address lookup did not return a match with the SUPER bit set.

11.13.4.4.6.11.1.19 ALE DA=SA Drop (Offset = 98h)

All ports

The total number of frames received on a port that were dropped (zero PORT_MASK) due to destination address equal to source address.

- Any data or MAC control frame which matched a unicast, broadcast or multicast address, or matched due to promiscuous mode
- Any length (including less than 64 bytes and greater than RX_MAXLEN bytes)
- had no CRC error, alignment error, or code error
- the destination address was equal to the source address
- the source address was not an entry in the table.

11.13.4.4.6.11.1.20 ALE Unknown Unicast (Offset = A8h)

All ports

The total number of frames received on a port that had a unicast destination address with an unknown source address.

- was any data frame with a unicast destination address
- the source address was not a table entry
- was of length 64 to RX_MAXLEN bytes inclusive
- had no CRC error, alignment error, or code error

11.13.4.4.6.11.1.21 ALE Unknown Unicast Bytecount (Offset = ACh)

The total number of bytes received on a port that had a unicast destination address with an unknown source address.

11.13.4.4.6.11.1.22 ALE Unknown Multicast (Offset = B0h)

The total number of frames received on a port that had a multicast destination address with an unknown source address.

11.13.4.4.6.11.1.23 ALE Unknown Multicast Bytecount(Offset = B4h)

The total number of bytes received on a port that had a multicast destination address with an unknown source address.

11.13.4.4.6.11.1.24 ALE Unknown Broadast (Offset = B8h)

The total number of frames received on a port that had a broadcast destination address with an unknown source address.

11.13.4.4.6.11.1.25 ALE Unknown Broadast Bytecount(Offset = BCh)

The total number of bytes received on a port that had a broadcast destination address with an unknown source address.

11.13.4.4.6.11.1.26 ALE Policer/Classifier Match (Offset = C0h)

All ports

The total number of frames received on a port that had a matched a policer.

- was any data frame
- matched a condition on a policer,
- was of length 64 to RX_MAXLEN bytes inclusive
- had no CRC error, alignment error, or code error

11.13.4.4.6.11.2 Tx-only Statistics Descriptions

The maximum and minimum transmit frame size is software controllable.

11.13.4.4.6.11.2.1 Good Tx Frames (Offset = 34h)

All ports

The total number of good frames received on the port. A good frame is defined to be:

- Any data or MAC control frame which matched a unicast, broadcast or multicast address, or matched due to promiscuous mode
- Any length
- Had no late or excessive collisions, no carrier loss and no underrun

11.13.4.4.6.11.2.2 Broadcast Tx Frames (Offset = 38h)

All ports

The total number of good broadcast frames received on the port. A good broadcast frame is defined to be:

- Any data or MAC control frame which was destined for only address FF.FF.FF.FF.FF.FF
- Any length
- Had no late or excessive collisions, no carrier loss and no underrun

11.13.4.4.6.11.2.3 Multicast Tx Frames (Offset = 3Ch)

All ports

The total number of good multicast frames received on the port. A good multicast frame is defined to be:

- Any data or MAC control frame which was destined for any multicast address other than FF.FF.FF.FF.FF.FF
- Any length
- Had no late or excessive collisions, no carrier loss and no underrun

11.13.4.4.6.11.2.4 Pause Tx Frames (Offset = 40h)

Ethernet port 1

This statistic indicates the number of IEEE 802.3X pause frames transmitted by the port.

Pause frames cannot contain a CRC error because they are created in the transmitting MAC, so these error conditions have no effect upon the statistic. Pause frames sent by software will not be included in this count.

Since pause frames are only transmitted in full duplex, carrier loss and collisions have no effect upon this statistic.

Transmitted pause frames are always 64-byte multicast frames so will appear in the *Tx Multicast Frames* and *64octet Frames* statistics.

11.13.4.4.6.11.2.5 Deferred Tx Frames (Offset = 44h)

Ethernet port 1

The total number of frames transmitted on the port that first experienced deferment. Such a frame:

- Was any data or MAC control frame destined for any unicast, broadcast or multicast address
- Was any size
- Had no carrier loss and no underrun
- Experienced no collisions before being successfully transmitted
- Found the medium busy when transmission was first attempted, so had to wait.

CRC errors have no effect upon this statistic.

11.13.4.4.6.11.2.6 Collisions (Offset = 48h)

Ethernet port 1

This statistic records the total number of times that the port experienced a collision. Collisions occur under two circumstances.

1. When a transmit data or MAC control frame:
   - Was destined for any unicast, broadcast or multicast address
   - Was any size
   - Had no carrier loss and no underrun
   - Experienced a collision. A jam sequence is sent for every non-late collision, so this statistic will increment on each occasion if a frame experiences multiple collisions (and increments on late collisions)

     CRC errors have no effect upon this statistic.
2. When the port is in half-duplex mode, flow control is active, and a frame reception begins.

11.13.4.4.6.11.2.7 Single Collision Tx Frames (Offset = 4Ch)

Ethernet port 1

The total number of frames transmitted on the port that experienced exactly one collision. Such a frame:

- Was any data or MAC control frame destined for any unicast, broadcast or multicast address
- Was any size
- Had no carrier loss and no underrun
- Experienced one collision before successful transmission. The collision was not late.

CRC errors have no effect upon this statistic.

11.13.4.4.6.11.2.8 Multiple Collision Tx Frames (Offset = 50h)

Ethernet port 1

The total number of frames transmitted on the port that experienced multiple collisions. Such a frame:

- Was any data or MAC control frame destined for any unicast, broadcast or multicast address
- Was any size

- Had no carrier loss and no underrun
- Experienced 2 to 15 collisions before being successfully transmitted. None of the collisions were late.

CRC errors have no effect upon this statistic.

11.13.4.4.6.11.2.9 Excessive Collisions (Offset = 54h)

Ethernet port 1

The total number of frames for which transmission was abandoned due to excessive collisions. Such a frame:

- Was any data or MAC control frame destined for any unicast, broadcast or multicast address
- Was any size
- Had no carrier loss and no underrun
- Experienced 16 collisions before abandoning all attempts at transmitting the frame. None of the collisions were late.

CRC errors have no effect upon this statistic.

11.13.4.4.6.11.2.10 Late Collisions (Offset = 58h)

Ethernet port 1

The total number of frames on the port for which transmission was abandoned because they experienced a late collision. Such a frame:

- Was any data or MAC control frame destined for any unicast, broadcast or multicast address
- Was any size
- Experienced a collision later than 512 bit-times into the transmission. There may have been up to 15 previous (non-late) collisions which had previously required the transmission to be re-attempted. The *Late Collisions* statistic dominates over the single-, multiple- , and excessive- collision statistics. If a late collision occurs, the frame will not be counted in any of these other three statistics.

CRC errors have no effect upon this statistic.

11.13.4.4.6.11.2.11 Carrier Sense Errors (Offset = 60h)

Ethernet port 1

The total number of frames received on the port that had a CPDMA middle of frame (MOF) overrun. MOF overrun frame is defined to be:

- Was any data or MAC control frame destined for any unicast, broadcast or multicast address
- Was any size
- The carrier sense condition was lost or never asserted when transmitting the frame (the frame is not retransmitted). This is a transmit only statistic. Carrier Sense is a don’t care for received frames. Transmit frames with carrier sense errors are sent until completion and are not aborted.

CRC errors have no effect upon this statistic.

11.13.4.4.6.11.2.12 Tx Octets (Offset = 64h)

All ports

The total number of bytes in all good frames transmitted on the port. A good frame is defined to be:

- Any data or MAC control frame which was destined for any unicast, broadcast or multicast address
- Was any size
- Had no late or excessive collisions, no carrier loss and no underrun.

11.13.4.4.6.11.2.13 Tx Memory Protect Errors (Offset = 17Ch)

All ports

The total number of transmit frames on the port that had a memory protect CRC error on egress:

- Any frame destined to be transmitted,
- Was any size
- Had a memory protect CRC error on egress.

NOTE:

1. Frames to the host with memory protect errors are dropped via TXST_DROP. Ethernet frames will have at least one byte of the generated port type CRC inverted on egress.
2. This statistic is 8-bits wide only and will not rollover but will saturate at 0xFF.
3. A non-zero value in this statistic will issue a STAT_PEND interrupt for the associated port.

11.13.4.4.6.11.3 Rx- and Tx-Shared Statistics Descriptions

11.13.4.4.6.11.3.1 Rx + Tx 64 Octet Frames (Offset = 68h)

All ports

The total number of 64-byte frames received and transmitted on the port. Such a frame is defined to be:

- Any data or MAC control frame which was destined for any unicast, broadcast or multicast address
- Did not experience late collisions, excessive collisions, or carrier sense error
- Was exactly 64 bytes long. (If the frame was being transmitted and experienced carrier loss that resulted in a frame of this size being transmitted, then the frame will be recorded in this statistic).

CRC errors, code/align errors and overruns do not affect the recording of frames in this statistic.

11.13.4.4.6.11.3.2 Rx + Tx 65–127 Octet Frames (Offset = 6Ch)

All ports

The total number of frames of size 65 to 127 bytes received and transmitted on the port. Such a frame is defined to be:

- Any data or MAC control frame which was destined for any unicast, broadcast or multicast address
- Did not experience late collisions, excessive collisions, or carrier sense error
- Was 65 to 127 bytes long

CRC errors, code/align errors, underruns and overruns do not affect the recording of frames in this statistic.

11.13.4.4.6.11.3.3 Rx + Tx 128–255 Octet Frames (Offset = 70h)

All ports

The total number of frames of size 128 to 255 bytes received and transmitted on the port. Such a frame is defined to be:

- Any data or MAC control frame which was destined for any unicast, broadcast or multicast address
- Did not experience late collisions, excessive collisions, or carrier sense error
- Was 128 to 255 bytes long

CRC errors, code/align errors, underruns and overruns do not affect the recording of frames in this statistic.

11.13.4.4.6.11.3.4 Rx + Tx 256–511 Octet Frames (Offset = 74h)

All ports

The total number of frames of size 256 to 511 bytes received and transmitted on the port. Such a frame is defined to be:

- Any data or MAC control frame which was destined for any unicast, broadcast or multicast address
- Did not experience late collisions, excessive collisions, or carrier sense error
- Was 256 to 511 bytes long

CRC errors, code/align errors, underruns and overruns do not affect the recording of frames in this statistic.

11.13.4.4.6.11.3.5 Rx + Tx 512–1023 Octet Frames (Offset = 78h)

All ports

The total number of frames of size 512 to 1023 bytes received and transmitted on the port. Such a frame is defined to be:

- Any data or MAC control frame which was destined for any unicast, broadcast or multicast address
- Did not experience late collisions, excessive collisions, or carrier sense error
- Was 512 to 1023 bytes long

CRC errors, code/align errors, underruns and overruns do not affect the recording of frames in this statistic.

11.13.4.4.6.11.3.6 Rx + Tx 1024_Up Octet Frames (Offset = 7Ch)

All ports

The total number of frames of size 1024 to RX_MAXLEN bytes for receive or 1024 up for transmit on the port. Such a frame is defined to be:

- Any data or MAC control frame which was destined for any unicast, broadcast or multicast address
- Did not experience late collisions, excessive collisions, or carrier sense error
- Was 1024 to RX_MAXLEN bytes long on receive, or any size on transmit

CRC errors, code/align errors, underruns and overruns do not affect the recording of frames in this statistic.

11.13.4.4.6.11.3.7 Net Octets (Offset = 80h)

All ports

The total number of bytes of frame data received and transmitted on the port. Each frame counted:

- was any data or MAC control frame destined for any unicast, broadcast or multicast address (address match does not matter)
- Any length (including less than 64 bytes and greater than RX_MAXLEN bytes)

Also counted in this statistic is:

- Every byte transmitted before a carrier-loss was experienced
- Every byte transmitted before each collision was experienced, (that is, multiple retries are counted each time)
- Every byte received if the port is in half-duplex mode until a jam sequence was transmitted to initiate flow control. (The jam sequence was not counted to prevent double-counting)

Error conditions such as alignment errors, CRC errors, code errors, overruns and underruns do not affect the recording of bytes by this statistic.

The objective of this statistic is to give a reasonable indication of ethernet utilization.

Table 11-2031. Rx Statistics Summary

Rx Statistic	Frame /Oct	Rx/Rx +Tx	Frame Type					Frame Size (bytes)								Event				
			MAC control		Data[1]															
			Pause frame	Non-pause [3]	Multi cast	Broad cast	Uni cast	<64	64	65-127	128-255	256-511	512-1023	1024-rx_maxlen	>rx_maxlen	Flow Coll.[2]	CRC Error	Align/ Code	Overrun	Addr. Disc.
Good Rx Frames	F	R	(y|[4]	y|	y|	y|	y)	n	(y|	y|	y|	y|	y|	y)	n	-[5]	n	n	-	n
Broadcast Rx Frames	F	R	(%|[6]	%|	n	y)	n	n	(y|	y|	y|	y|	y|	y)	n	-	n	n	-	n
Multicast Rx Frames	F	R	(%|	%|	y)	n	n	n	(y|	y|	y|	y|	y|	y)	n	-	n	n	-	n
Pause Rx Frames	F	R	y	n	n	n	n	n	(y|	y|	y|	y|	y|	y)	n	-	n	n	-	-
Rx CRC Errors	F	R	(y|	y|	y|	y|	y)	n	(y|	y|	y|	y|	y|	y)	n	-	y	n	-	n
Rx Align/Code Errors	F	R	(y|	y|	y|	y|	y)	n	(y|	y|	y|	y|	y|	y)	n	-	-	y	-	n
Oversized Rx Frames	F	R	(y|	y|	y|	y|	y)	n	n	n	n	n	n	n	y	-	n	n	-	n
Rx Jabbers	F	R	(y|	y|	y|	y|	y)	n	n	n	n	n	n	n	y	-	(y|	y)	-	n
Undersized Rx Frames	F	R	n	n	(y|	y|	y)	y	n	n	n	n	n	n	n	-	n	n	-	n
Rx Fragments	F	R	n	n	(y|	y|	y)	y^[7]	n	n	n	n	n	n	n	-	(y|	y)	-	-
Rx Overruns[8]	F	R	(y|	y|	y|	y|	y)	(y|	y|	y|	y|	y|	y|	y|	y)	-	-	-	y	n
64octet Frames	F	R+T[9]	(y|	y|	y|	y|	y)	n	y	n	n	n	n	n	n	-	-	-	-	n
65-127octet Frames	F	R+T	(y|	y|	y|	y|	y)	n	n	y	n	n	n	n	n	-	-	-	-	n
128-255octet Frames	F	R+T	(y|	y|	y|	y|	y)	n	n	n	y	n	n	n	n	-	-	-	-	n
256-511octet Frames	F	R+T	(y|	y|	y|	y|	y)	n	n	n	n	y	n	n	n	-	-	-	-	n
512-1023octet Frames	F	R+T	(y|	y|	y|	y|	y)	n	n	n	n	n	y	n	n	-	-	-	-	n
1024-UPoctet Frames	F	R+T	(y|	y|	y|	y|	y)	n	n	n	n	n	n	y	n	-	-	-	-	n
Rx Octets	O	R	(y|	y|	y|	y|	y)	n	(y|	y|	y|	y|	y|	y)	n	-	n	n	-	n
Net Octets	O	R+T	(y|	y|	y|	y|	y)	(y|	y|	y|	y|	y|	y|	y|	y|	y)	-	-	-	-

(1) The multicast, broadcast and unicast columns in the table refer to non-MAC Control/non-pause frames (i.e. data frames).

(2) Flow coll. are half-duplex collisions forced by the MAC to achieve flow-control. A collision will be forced during the first 8 bytes so should not show in frame fragments. Some of the '-'s in this column might in reality be 'n's.

(3) The non-pause column refers to all MAC control frames (for example, frames with length/type=88.08) with opcodes other than 0x0001. The pauseframe column refers to MAC frames with the opcode=0x0001.

(4) "AND" is assumed horizontally across the table between all conditions which form the statistic (marked y or n) except where (y|y), meaning "OR" is indicated. Parentheses are significant.

(5) "-" indicates conditions which are ignored in the formations of the statistic.

(6) "%" If either a MAC control frame or pause frame has a multicast or broadcast destination address then the appropriate statistics will be updated.

(7) "y^" Frame fragments are not counted if less than 8 bytes.

(8) The rx_overruns stat is for RX_MOF_OVERRUNS and RX_SOF_OVERRUNS added together.

(9) Statistics marked "R+T" are formed by summing the Rx and Tx statistics, each of which is formed independently.

Table 11-2032. Tx Statistics Summary

Tx Statistic[1]	Frame/Oct	Tx/Rx+Tx	Frame Type					Frame Size (bytes)							Event										
			MAC control[2]		Data											Collision Type									
			Pause-MAC	Any-CPU	Multicast	Broadcast	Unicast	64	65-127	128-255	256-511	512-1023	1024-1535	>1535	CRC Error	Flow[3]	1	2-15	16	Late	No Carrier	Queued	Deferred	Underrun
Good Tx Frames	F	T	(y|[4]	y|	y|	y|	y)	(y|	y|	y|	y|	y|	y|	y)	-[5]	-	-	-	n	n	n	-	-	n
Broadcast Tx Frames	F	T	n	(%|[6]	n	y)	n	(y|	y|	y|	y|	y|	y|	y)	-	-	-	-	n	n	n	-	-	n
Multicast Tx Frames	F	T	(y|	%|	y)	n	n	(y|	y|	y|	y|	y|	y|	y)	-	-	-	-	n	n	n	-	-	n
Pause Tx Frames	F	T	y	n	n	n	n	y	n	n	n	n	n	n	-	-	-	-	-	-	-	-	-	-	-
Collisions	F	T	n	(y|	y|	y|	y)	(y|	y|	y|	y|	y|	y|	y)	-	(+[7]	+	+	+	+)	n	-	-	-
Single Collision Tx Frames	F	T	n	(y|	y|	y|	y)	(y|	y|	y|	y|	y|	y|	y)	-	-	y	n	n	n	n	-	-	-
Multiple Collision Tx Frames	F	T	n	(y|	y|	y|	y)	(y|	y|	y|	y|	y|	y|	y)	-	-	n	y	n	n	n	-	-	-
Excessive Collisions	F	T	n	(y|	y|	y|	y)	(y|	y|	y|	y|	y|	y|	y)	-	-	n	n	y	n	n	-	-	-
Late Collisions	F	T	n	(y|	y|	y|	y)	n	(y|	y|	y|	y|	y|	y)	-	-	-	-	-	y	-	-	-	-
Deferred Tx Frames	F	T	n	(y|	y|	y|	y)	(y|	y|	y|	y|	y|	y|	y)	-	-	n	n	n	n	n	-	y	n
Carrier Sense Errors	F	T	(y|	y|	y|	y|	y)	(y|	y|	y|	y|	y|	y|	y)	-	-	-	-	-	-	y	-	-	-
64octet Frames	F	R+T[8]	(y|	y|	y|	y|	y)	y	n	n	n	n	n	n	-	-	-	-	n	n	n	-	-	-
65-127octet Frames	F	R+T	(y|	y|	y|	y|	y)	n	y	n	n	n	n	n	-	-	-	-	n	n	n	-	-	-
128-255octet Frames	F	R+T	(y|	y|	y|	y|	y)	n	n	y	n	n	n	n	-	-	-	-	n	n	n	-	-	-

(1) When the transmit Tx FIFO is drained due to the MAC being disabled or link being lost, then the frames being purged will not appear in the Tx statistics.
(2) Pause (MAC) frames are issued in the MAC as perfect (no CRC error) 64 byte frames in full duplex only, so they cannot collide.
(3) The flow collision type is for half-duplex collisions forced by the MAC to achieve flow control. Some of the '-'s in this column might in reality be 'n's. To prevent double-counting, Net Octets are unaffected by the jam sequence – the 'received' bytes, however, are counted. (See Table 11-2031.)
(4) "AND" is assumed horizontally across the table between all conditions which form the statistic (marked y or n) except where (y|y), meaning "OR" is indicated. Parentheses are significant.
(5) "-" indicates conditions which are ignored in the formations of the statistic.
(6) "%" If a CPU sourced MAC control frame has a multicast or broadcast destination address then the appropriate statistics will be updated.
(7) "+" indicates collisions which are "summed" (i.e. every collision is counted in the Collisions statistic). Jam sequences used for halfduplex flow control are also counted.
(8) Statistics marked "R+T" are formed by summing the Rx and Tx statistics, each of which is formed independently.

Table 11-2032. Tx Statistics Summary (continued)

Tx Statistic[1]	Frame/Oct	Tx/Rx+Tx	Frame Type					Frame Size (bytes)							Event										
			MAC control[2]		Data			64	65-127	128-255	256-511	512-1023	1024-1535	>1535	CRC Error	Collision Type					No Carrier	Queued	Deferred	Underrun
			Pause-MAC	Any-CPU	Multicast	Broadcast	Unicast									Flow[3]	1	2-15	16	Late				
256-511octet Frames	F	R+T	(y|	y|	y|	y|	y)	n	n	n	y	n	n	n	-	-	-	-	n	n	n	-	-	-
512-1023octet Frames	F	R+T	(y|	y|	y|	y|	y)	n	n	n	n	y	n	n	-	-	-	-	n	n	n	-	-	-
1024-UPoctet Frames	F	R+T	(y|	y|	y|	y|	y)	n	n	n	n	n	y	y	-	-	-	-	n	n	n	-	-	-
Tx Octets	O	T	(y|	y|	y|	y|	y)	(y|	y|	y|	y|	y|	y|	y)	-	-	-	-	n	n	n	-	-	n
Net Octets	O	R+T	(y|	y|	y|	y|	y)	(y|	y|	y|	y|	y|	y|	y)	-	-	$[9]	$	$	$	$	-	-	-

[9] "$" Every byte written on the wire during each retry attempt is also counted in addition to frames which experience no collisions or carrier loss.

11.13.4.4.7 Common Platform Time Sync (CPTS)

The Common Platform Time Sync (CPTS) module is used to facilitate host control of time sync operations. It enables compliance with the IEEE 1588-2008 standard for a precision clock synchronization protocol.

Main features of CPTS module are:

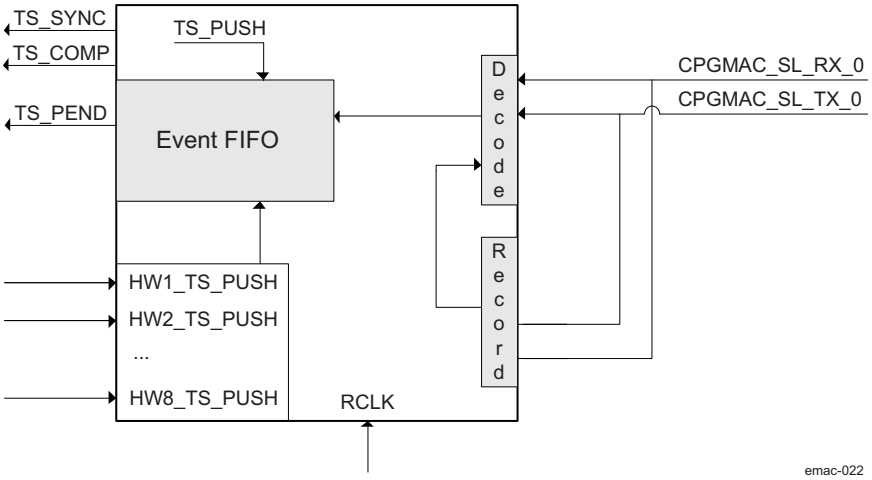
- Supports the selection of multiple external clock sources
- Software control of time sync events via interrupt or polling
- Supports up to 8 hardware timestamp push inputs
- Supports timestamp counter compare output (TS_COMP)
- Supports timestamp counter bit output (TS_SYNC)
- 32-bit and 64-bit timestamp modes

11.13.4.4.7.1 CPTS Architecture

Figure 11-922 shows the architecture of the CPTS module inside the EMAC Ethernet Subsystem. Time stamp values for every packet transmitted or received on either port of the EMAC are recorded. At the same time, each packet is decoded to determine if it is a valid time sync event. If so, an event is loaded into the Event FIFO for processing containing the recorded time stamp value when the packet was transmitted or received.

In addition, both hardware (HWx_TS_PUSH) and software (TS_PUSH) can be used to read the current time stamp value though the Event FIFO. The reference clock used for the time stamp (CPTS_RFTCLK) can be derived from several sources.

Figure 11-922. CPTS Block Diagram



NOTE: See Figure 11-914, *CPTS Integration* for CPTS integration in the device.

11.13.4.4.7.2 CPTS Initialization

The CPTS module should be configured as follows:

1. Reset the CPTS module.
2. Clear the CPTS_EN bit in the CPTS_CONTROL register.
3. Write the RFTCLK_SEL value in the CPTS_RFTCLK_SEL register with the desired reference clock selection.
4. Set the CPTS_EN bit in the CPTS_CONTROL register.
5. If using interrupts and not polling, enable the interrupt by setting the TS_PEND_EN bit in the CPTS_INT_ENABLE register.

11.13.4.4.7.3 32-bit Time Stamp Value

The time stamp value is a 32-bit value that increments on each CPTS_RFTCLK rising edge when CPTS_EN is set to 1. When CPTS_EN is cleared to 0, the time stamp value is reset to 0.

If more than 32-bits of time stamp are required by the application, the host software must maintain the necessary number of upper bits. The upper time stamp value should be incremented by the host when the rollover event is detected.

For test purposes, the time stamp can be written via the time stamp load function (CPTS_TS_LOAD_LOW_VAL and CPTS_TS_LOAD_EN registers).

11.13.4.4.7.4 64-bit Time Stamp Value

The time stamp value is a 64-bit value that increments on each CPTS_RFTCLK rising edge when CPTS_EN is set to 1. When CPTS_EN is cleared to 0, the time stamp value is reset to 0.

64-bit mode is selected via CPTS_CONTROL[5] MODE_64BIT bit set to 1.

For test purposes, the time stamp can be written via the time stamp load function (CPTS_TS_LOAD_LOW_VAL, CPTS_TS_LOAD_HIGH_VAL, and CPTS_TS_LOAD_EN registers).

11.13.4.4.7.5 Event FIFO

All time sync events are push onto the Event FIFO. There are 10 locations in the event FIFO with no overrun indication supported. Software must service the event FIFO in a timely manner to prevent FIFO overrun.

11.13.4.4.7.6 Timestamp Compare Output

CPTS features one Time Stamp Compare (TS_COMP) output.

11.13.4.4.7.6.1 Non-Toggle Mode

The TS_COMP output is asserted for CPTS_TS_COMP_LEN[23-0] RCLK periods when the TIME_STAMP[31-0] value compares with the CPTS_TS_COMP_LOW_VAL[31-0] and the length value is non-zero. The TS_COMP rising edge occurs three RCLK periods after the values compare. A timestamp compare event is pushed into the event FIFO when TS_COMP is asserted. The polarity of the TS_COMP output is determined by the CPTS_CONTROL[2] TS_POLARITY bit. The output is asserted low when the polarity bit is 0.

64-bit mode operation is identical to 32-bit mode except that all 64-bits of the TIME_STAMP[63-0] are used instead of only the lower 32-bits.

11.13.4.4.7.6.2 Toggle Mode

The TS_COMP output is asserted for CPTS_TS_COMP_LEN RCLK periods when the TIME_STAMP[31-0] value compares with the CPTS_TS_COMP_LOW_VAL[31-0] and the length value is non-zero. The TS_COMP toggles thereafter on CPTS_TS_COMP_LEN[23-0] RCLK periods. The length high or low can be adjusted by writing the CPTS_TS_COMP_NUDGE[7-0] register value which is a two's complement value. A value of 0xFF will subtract one RCLK from the CPTS_TS_COMP_LEN value. A value of 0x01 will add one RCLK to the CPTS_TS_COMP_LEN value. Only a single high or low time is adjusted (nudged) and the CPTS_TS_COMP_NUDGE value is cleared to zero when the nudge has occurred. The TS_COMP output is asserted low when the CPTS_CONTROL[2] TS_COMP_POLARITY bit is 0.

No compare events and no CPTS_EVNT interrupts are generated in toggle mode.

The CPTS_CONTROL[6] TS_COMP_TOG bit must be set for toggle mode, and must be set before writing a non-zero value to TS_COMP_LEN[23-0].

64-bit mode operation is identical to 32-bit mode except that all 64-bits of the TIMESTAMP[63-0] are used instead of only the lower 32-bits.

11.13.4.4.7.7 Timestamp Sync Output

The TS_SYNC output is a selected bit of the TIME_STAMP counter value. One of bits 17-31 can be selected in CPTS_CONTROL[31-28] TS_SYNC_SEL. The TS_SYNC output is disabled when CPTS_CONTROL[31-28] TS_SYNC_SEL is zero.

If the selected counter bit is 1 at the time when TS_SYNC_SEL value is written then a rising edge will not occur on the TS_SYNC output. A rising edge will occur on the TS_SYNC output upon the next transition-to-1 of the selected counter bit. The TS_SYNC_SEL value must be written to zero before changing to a different non-zero value. No events are generated due to the TS_SYNC operation. The TS_SYNC output is two RCLK periods after the actual count value.

11.13.4.4.7.8 Time Sync Events

Time Sync events are 96-bit or 128-bit (for 32-bit and 64-bit time stamp respectively) values that are pushed onto the event FIFO and read by software in 32-bit reads. Four 32-bit registers, CPTS_EVENT_0 throught CPTS_EVENT_3 hold the data of a time sync event. There are eight types of sync events:

- Time stamp push event
- Time stamp counter rollover event (32-bit mode only)
- Time stamp counter half-rollover event (32-bit mode only)
- Hardware Time Stamp Push Event
- Ethernet receive event
- Ethernet transmit event
- Time Stamp Compare Event
- Host Event

11.13.4.4.7.8.1 Time Stamp Push Event

Software can obtain the current time stamp value (at the time of the write) by initiating a time stamp push event. The push event is initiated by setting the TS_PUSH bit of the CPTS_TS_PUSH register. The time stamp value is returned in the event, along with a time stamp push event code.

11.13.4.4.7.8.2 Time Stamp Counter Rollover Event (32-bit mode only)

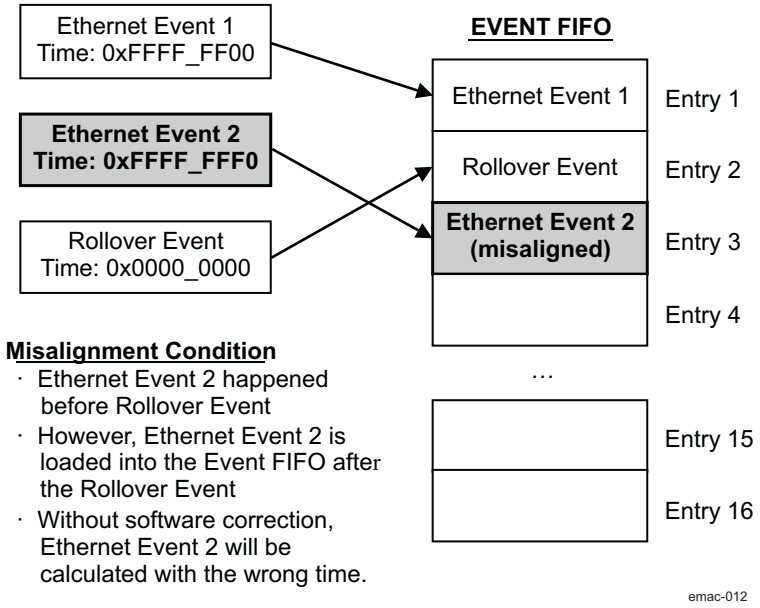
The CPTS module contains a 32-bit time stamp value. The counter upper bits are maintained by host software. The rollover event indicates to software that the time stamp counter has rolled over from 0xFFFF FFFF to 0x0000 0000 and the software-maintained upper count value should be incremented.

11.13.4.4.7.8.3 Time Stamp Counter Half-rollover Event (32-bit mode only)

The CPTS includes a time stamp counter half-rollover event. The half-rollover event indicates to software that the time stamp value has incremented from 0x7FFF FFFF to 0x8000 0000. The half-rollover event is included to enable software to correct a misaligned event condition. The half-rollover event is included to enable software to determine the correct time for each event that contains a valid time stamp value, such as an Ethernet event. If an Ethernet event occurs around a counter rollover (full rollover), the rollover event could possibly be loaded into the event FIFO before the Ethernet event, even though the Ethernet event time was actually taken before the rollover. Figure 11-923 shows a misalignment condition.

Host software must detect and correct for misaligned event conditions. For every event after a rollover and before a half-rollover, software must examine the time stamp most significant bit. If bit 31 of the time stamp value is low (0x0000 0000 through 0x7FFF FFFF), then the event time stamp was taken after the rollover and no correction is required. If the value is high (0x8000 0000 through 0xFFFF FFFF), the time stamp value was taken before the rollover and a misalignment is detected. The misaligned case indicates to software that it must subtract one from the upper count value stored in software to calculate the correct time for the misaligned event. The misaligned event occurs only on the rollover boundary and not on the half-rollover boundary. Software only needs to check for misalignment from a rollover event to a half-rollover event.

Figure 11-923. Event FIFO Misalignment Condition



11.13.4.4.7.8.4 Hardware Time Stamp Push Event

There are eight hardware time stamp inputs (HW1/8_TS_PUSH) that can cause hardware time stamp push events to be loaded into the Event FIFO. Each time stamp input is mapped in the device as shown in Figure 11-914.

The event is loaded into the event FIFO on the rising edge of the timer, and the PORT_NUMBER field in the CPTS_EVENT_1 register indicates the hardware time stamp input that caused the event.

Each hardware time stamp input must be asserted for at least 10 periods of the selected RCLK clock. Each input can be enabled or disabled by setting the respective bits in the CPTS_CONTROL register.

Hardware time stamps are intended to be an extremely low frequency signals, such that the event FIFO does not overrun. Software must keep up with the event FIFO and ensure that there is no overrun, or events will be lost.

11.13.4.4.7.8.5 Ethernet Port Events

Packets transmitted or received on each Ethernet port can generate Ethernet Transmit Events or Ethernet Receive Events, respectively. The CPTS hardware will decode each packet to determine if it is a valid CPTS time sync event.

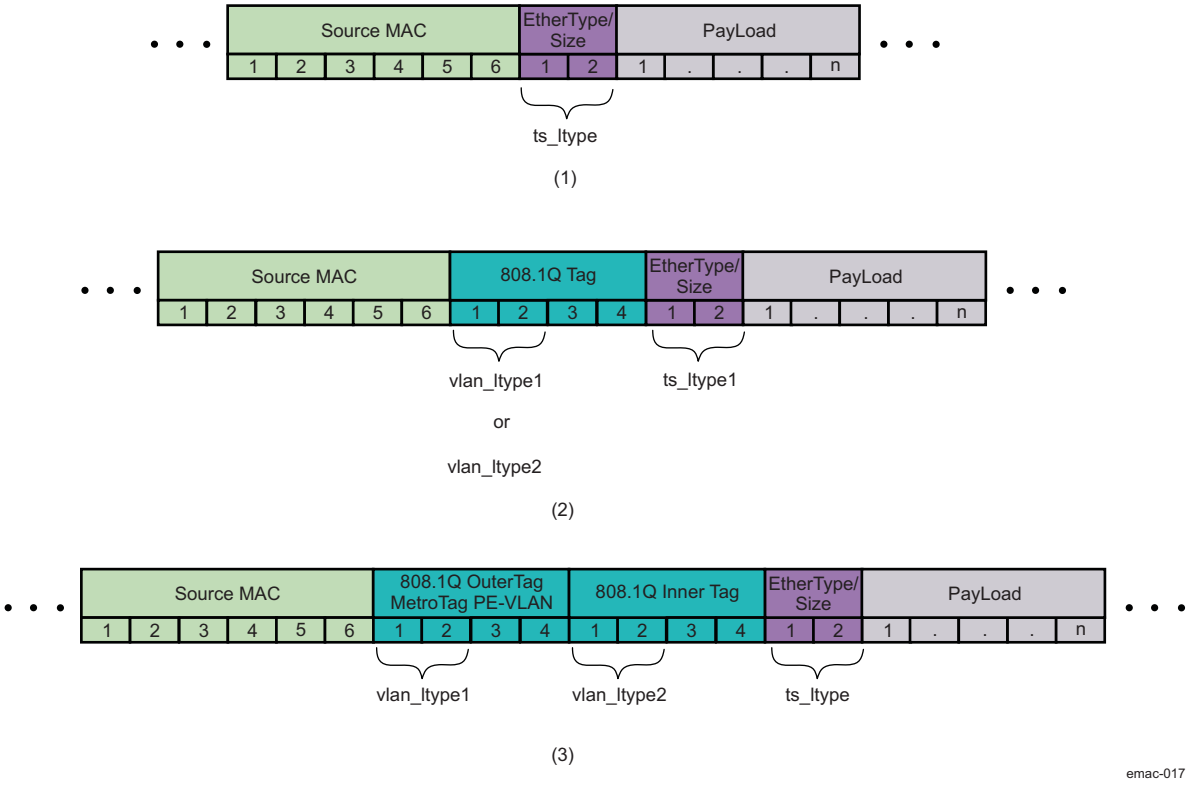
According to the IEEE 802.3 Ethernet standard, each Ethernet frame contains a 2-octet EtherType field to indicate which protocol is encapsulated in the PayLoad field, as shown in Figure 11-924. For standard time sync packets, this will contain the EtherType for the Precision Time Protocol (IEEE 1588), which is defined as 0x88F7. The CPTS hardware will compare this field to the TS_LTYPE1 field or the TS_LTYPE2 field (depending on which enable bit was set) in CPSW_P1_TS_SEQ_LTYPE or CPSW_P1_TS_CTL_LTYPE2 register, which should also be programmed to 88F7h.

When a virtual LAN is used, an additional 4-octet 802.1Q tag is inserted in the Ethernet frame before the EtherType field, as shown in Figure 11-924. To indicate to the CPTS hardware that a virtual LAN is in use, the VLAN_LTYPE1_EN (or VLAN_LTYPE2_EN) enable bit must be set in the CPSW_P1_TS_CTL register. The EtherType for the 802.1Q tag is defined as 0x8100, and the CPTS hardware will compare this value to the VLAN_LTYPE1 (or VLAN_LTYPE2 depending on which enable bit was set) field in the

CPSW_P1_TS_VLAN_LTYPE register, which should also be programmed to 0x8100.

When two stacked VLANs are used, two additional 4-octet 801.Q tags are inserted in the Ethernet frame before the EtherType field, as shown in Figure 11-924. In this case, both VLAN_LTYPE1 and VLAN_LTYPE2 must be enabled. The outer tag must match the value of the VLAN_LTYPE1 field, and the inner tag must match the value of the VLAN_LTYPE2 field.

Figure 11-924. Partial Ethernet-II Frames Showing Register Mapping of EtherTypes for a Simple Frame (1), a Single 1Q Tag Added (2), and Two 1Q Tags Added (3)



11.13.4.4.7.8.5.1 Ethernet Receive Event

This section describes Ethernet port receive events. Ethernet port generates time synchronization events for valid received time sync packets. For every packet received on the Ethernet port, a timestamp will be captured by the receive module inside the CPTS for the corresponding port. The time stamp will be captured by the receive module regardless of whether or not the packet is a time synchronization packet to make sure that the time stamp is captured as soon as possible. The packet is sampled on both the rising and falling edges of the CPTS_RCLK, and the time stamp will be captured once the start of frame delimiter for the receive packet is detected.

After the time stamp has been captured, the receive interface will begin parsing the packet to determine if it is a valid Ethernet time synchronization packet. The CPSW decoder determines if the packet is a valid ethernet receive time synchronization event. The receive interface for the port will use the following criteria to determine if the packet is a valid Annex D, Annex E, or Annex F time synchronization Ethernet receive event:

Annex D (Ethernet Receive)

1. Receive annex D time sync is enabled (TS_RX_ANNEX_D_EN is set in the CPSW_P1_TS_CTL register).
2. One of the sequences below is true.
   a. The first packet LTYPE matches 0x0800
   b. The first packet LTYPE matches TS_VLAN_LTYPE1 in the CPSW_P1_TS_VLAN_LTYPE register and TS_RX_VLAN_LTYPE1_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches 0x0800

c. The first packet LTYPE matches TS_VLAN_LTYPE2 in the CPSW_P1_TS_VLAN_LTYPE register and TS_RX_VLAN_LTYPE2_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches 0x0800

d. The first packet LTYPE matches TS_VLAN_LTYPE1 in the CPSW_P1_TS_VLAN_LTYPE register and TS_RX_VLAN_LTYPE1_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches TS_VLAN_LTYPE2 in the CPSW_P1_TS_VLAN_LTYPE register and TS_RX_VLAN_LTYPE2_EN is set in the CPSW_P1_TS_CTL register and the third packet LTYPE matches 0x0800

3. Byte 14 (the byte after the LTYPE) contains 0x45 (IPv4).

NOTE: The byte numbering assumes that there are no VLANs. The byte number is intended to show the relative order of the bytes.

4. Byte 20 contains 0bXXX00000 (5 lower bits zero) and Byte 21 contains 0x00 (fragment offset zero)
5. Byte 22 contains 0x00 if the TS_TTL_NONZERO_EN bit in the switch CPSW_P1_TS_CTL_LTYPE2 register is cleared to 0 -OR- byte 22 contains any value if TS_TTL_NONZERO_EN is set to 1. Byte 22 is the time to live field.
6. Byte 23 contains 0x11 (Next Header UDP Fixed).
7. The TS_UNI_EN bit in the CPSW_P1_TS_CTL_LTYPE2 register is cleared to 0 and Bytes 30 through 33 contain:
   a. Decimal 224.0.1.129 and the TS_129 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
   b. Decimal 224.0.1.130 and the TS_130 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
   c. Decimal 224.0.1.131 and the TS_131 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
   d. Decimal 224.0.1.132 and the TS_132 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
   e. Decimal 224.0.0.107 and the TS_107 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set

   -OR-

   The TS_UNI_EN bit in the CPSW_P1_TS_CTL_LTYPE2 register is set and Bytes 30 through 33 contain any values.
8. Bytes 36 and 37 contain:
   a. Decimal 0x01 and 0x3f respectively and the TS_319 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set -OR-
   b. Decimal 0x01 and 0x40 respectively and the TS_320 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set.
9. The PTP message begins in byte 42.
10. The packet message type is enabled in the MSG_TYPE_EN field in the CPSW_P1_TS_CTL register.
11. The packet was received without error (not long/short/mac_ctl/CRC/code/align).

Annex E (Ethernet Receive)

1. Receive annex E time sync is enabled (RX_ANNEXE_EN bit is set in the switch CPSW_P1_TS_CTL register).
2. One of the sequences below is true.
   a. The first packet LTYPE matches 0x86dd.
   b. The first packet LTYPE matches TS_VLAN_LTYPE1 in the CPSW_P1_TS_VLAN_LTYPE register and TS_RX_VLAN_LTYPE1_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches 0x86dd
   c. The first packet LTYPE matches TS_VLAN_LTYPE2 in the CPSW_P1_TS_VLAN_LTYPE register and TS_RX_VLAN_LTYPE2_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches 0x86dd
   d. The first packet LTYPE matches TS_VLAN_LTYPE1 in the CPSW_P1_TS_VLAN_LTYPE register and TS_RX_VLAN_LTYPE1_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches TS_VLAN_LTYPE2 in the CPSW_P1_TS_VLAN_LTYPE register and TS_RX_VLAN_LTYPE2_EN is set in the CPSW_P1_TS_CTL register and the third packet LTYPE

matches 0x86dd

3. Byte 14 (the byte after the LTYPE) contains 0x6X (IPv6).
4. Byte 20 contains 0x11 (UDP Fixed Next Header).
5. Byte 21 contains 0x01 (Hop Limit = 1).
6. The TS_UNI_EN bit in the CPSW_P1_TS_CTL_LTYPE2 register is cleared to 0 and Bytes 38 through 53 contain:
   a. FF0M:0:0:0:0:0:0:0181 and the TS_129 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
   b. FF0M:0:0:0:0:0:0:0182 and the TS_130 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
   c. FF0M:0:0:0:0:0:0:0183 and the TS_131 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
   d. FF0M:0:0:0:0:0:0:0184 and the TS_132 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
   e. FF0M:0:0:0:0:0:0:006B and the TS_107 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set

   NOTE: All values above are 16-bit hex numbers where M is enabled in the TS_MCAST_TYPE_EN field in the CPSW_P1_TS_CTL2 register.

   -OR-

   The TS_UNI_EN bit in the CPSW_P1_TS_CTL_LTYPE2 register is set to 1 and Bytes 38 through 53 contain any value.
7. Bytes 56 and 57 contain (UDP Header in bytes 54 through 61):
   a. Decimal 0x01 and 0x3f respectively and the TS_319 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
   b. Decimal 0x01 and 0x40 respectively and the TS_320 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set.
8. The PTP message begins in byte 62.
9. The packet message type is enabled in the MSG_TYPE_EN field in the CPSW_P1_TS_CTL register.
10. The packet was received without error (not long/short/mac_ctl/CRC/code/align).

Annex F (Ethernet Receive)

1. Receive Annex F time sync is enabled (TS_RX_ANNEX_F_EN is set in the switch CPSW_P1_TS_CTL register).
2. One of the sequences below is true:
   a. The first packet LTYPE matches TS_LTYPE1 in the CPSW_P1_TS_SEQ_LTYPE register. LTYPE 1 should be used when only one time sync LTYPE is to be enabled.
   b. The first packet LTYPE matches TS_LTYPE2 in the CPSW_P1_TS_CTL_LTYPE2 register and LTYPE2_EN is set in the CPSW_P1_TS_CTL register.
   c. The first packet LTYPE matches TS_VLAN_LTYPE1 in the CPSW_P1_TS_VLAN_LTYPE register and TS_RX_VLAN_LTYPE1_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches TS_LTYPE1 in the CPSW_P1_TS_SEQ_LTYPE register
   d. The first packet LTYPE matches TS_VLAN_LTYPE1 in the CPSW_P1_TS_VLAN_LTYPE register and TS_RX_VLAN_LTYPE1_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches TS_LTYPE2 in the CPSW_P1_TS_CTL_LTYPE2 register and LTYPE2_EN is set in the CPSW_P1_TS_CTL register.
   e. The first packet LTYPE matches TS_VLAN_LTYPE2 in the CPSW_P1_TS_VLAN_LTYPE register and TS_RX_VLAN_LTYPE2_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches TS_LTYPE1 in the CPSW_P1_TS_SEQ_LTYPE register.
   f. The first packet LTYPE matches TS_VLAN_LTYPE2 in the CPSW_P1_TS_VLAN_LTYPE register and TS_RX_VLAN_LTYPE2_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches TS_LTYPE2 in the CPSW_P1_TS_CTL_LTYPE2 register and LTYPE2_EN is set in the CPSW_P1_TS_CTL register.
   g. The first packet LTYPE matches TS_VLAN_LTYPE1 in the CPSW_P1_TS_VLAN_LTYPE register and TS_RX_VLAN_LTYPE1_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches TS_VLAN_LTYPE2 in the CPSW_P1_TS_VLAN_LTYPE register and

TS_RX_VLAN_LTYPE2_EN is set in the CPSW_P1_TS_CTL register and the third packet LTYPE matches TS_LTYPE1 in the CPSW_P1_TS_SEQ_LTYPE register.

h. The first packet LTYPE matches TS_VLAN_LTYPE1 in the CPSW_P1_TS_VLAN_LTYPE register and TS_RX_VLAN_LTYPE1_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches TS_VLAN_LTYPE2 in the CPSW_P1_TS_VLAN_LTYPE register and TS_RX_VLAN_LTYPE2_EN is set in the CPSW_P1_TS_CTL register and the third packet LTYPE matches TS_LTYPE2 in the CPSW_P1_TS_CTL_LTYPE2 register and LTYPE2_EN is set in the CPSW_P1_TS_CTL register

3. The PTP message begins in the byte after the LTYPE.
4. The packet message type is enabled in the TS_MSG_TYPE_EN field in the CPSW_P1_TS_CTL register.
5. The packet was received without error (not long/short/mac_ctl/CRC/code/align).

If all of the criteria described above are met for either Annex D, Annex E, or Annex F, and the packet is determined to be a valid time synchronization packet, then the RX interface will push an Ethernet receive event into the event FIFO.

11.13.4.4.7.8.5.2 Ethernet Transmit Event

This section describes Ethernet port transmit events. For every packet transmitted on the Ethernet ports, the port transmit interface will begin parsing the packet to determine if it is a valid Ethernet time synchronization packet. The CPTS transmit interface for the port will use to the following criteria to determine if the packet is a valid time synchronization Ethernet transmit event. The CPSW decoder determines if the packet is a valid ethernet receive time synchronization event. To be a valid Ethernet transmit time synchronization event, the conditions listed below must be true for either Annex D, Annex E, or Annex F:

Annex D (Ethernet Transmit)

1. Transmit annex D time sync is enabled (TS_TX_ANNEX_D_EN is set in the CPSW_P1_TS_CTL register).
2. One of the sequences below is true.
   a. The first packet LTYPE matches 0x0800
   b. The first packet LTYPE matches TS_VLAN_LTYPE1 in the CPSW_P1_TS_VLAN_LTYPE register and TX_VLAN_LTYPE1_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches 0x0800
   c. The first packet LTYPE matches TS_VLAN_LTYPE2 in the CPSW_P1_TS_VLAN_LTYPE register and TX_VLAN_LTYPE2_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches 0x0800
   d. The first packet LTYPE matches TS_VLAN_LTYPE1 in the CPSW_P1_TS_VLAN_LTYPE register and TX_VLAN_LTYPE1_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches TS_VLAN_LTYPE2 in the CPSW_P1_TS_VLAN_LTYPE register and TX_VLAN_LTYPE2_EN is set in the CPSW_P1_TS_CTL register and the third packet LTYPE matches 0x0800
3. Byte 14 (the byte after the LTYPE) contains 0x45 (IPv4).

NOTE: The byte numbering assumes that there are no VLANs. The byte number is intended to show the relative order of the bytes.

4. Byte 20 contains 0bXXX00000 (5 lower bits zero) and Byte 21 contains 0x00 (fragment offset zero)
5. Byte 22 contains 0x00 if the TS_TTL_NONZERO_EN bit in the switch CPSW_P1_TS_CTL_LTYPE2 register is cleared to 0 -OR- byte 22 contains any value if TS_TTL_NONZERO_EN is set to 1. Byte 22 is the time to live field.
6. Byte 23 contains 0x11 (Next Header UDP Fixed).
7. The TS_UNI_EN bit in the CPSW_P1_TS_CTL_LTYPE2 register is cleared to 0 and Bytes 30 through 33 contain:

a. Decimal 224.0.1.129 and the TS_129 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
b. Decimal 224.0.1.130 and the TS_130 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
c. Decimal 224.0.1.131 and the TS_131 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
d. Decimal 224.0.1.132 and the TS_132 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
e. Decimal 224.0.0.107 and the TS_107 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
f. The TS_UNI_EN bit in the CPSW_P1_TS_CTL_LTYPE2 register is set and Bytes 30 through 33 contain any values.

8. Bytes 36 and 37 contain:
   a. Decimal 0x01 and 0x3f respectively and the TS_319 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set -OR-
   b. Decimal 0x01 and 0x40 respectively and the TS_320 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set.
9. The PTP message begins in byte 42.
10. The packet message type is enabled in the MSG_TYPE_EN field in the CPSW_P1_TS_CTL register.
11. The packet was sent by host port 0.

Annex E (Ethernet Transmit)

1. Transmit annex E time sync is enabled (TS_TX_ANNEX_E_EN bit is set in the switch CPSW_P1_TS_CTL register).
2. One of the sequences below is true.
   a. The first packet LTYPE matches 0x86dd.
   b. The first packet LTYPE matches TS_VLAN_LTYPE1 in the CPSW_P1_TS_VLAN_LTYPE register and RX_VLAN_LTYPE1_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches 0x86dd
   c. The first packet LTYPE matches TS_VLAN_LTYPE2 in the CPSW_P1_TS_VLAN_LTYPE register and RX_VLAN_LTYPE2_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches 0x86dd
   d. The first packet LTYPE matches TS_VLAN_LTYPE1 in the CPSW_P1_TS_VLAN_LTYPE register and RX_VLAN_LTYPE1_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches TS_VLAN_LTYPE2 in the CPSW_P1_TS_VLAN register and RX_VLAN_LTYPE2_EN is set in the CPSW_P1_TS_CTL register and the third packet LTYPE matches 0x86dd
3. Byte 14 (the byte after the LTYPE) contains 0x6X (IPv6).
4. Byte 20 contains 0x11 (UDP Fixed Next Header).
5. Byte 21 contains 0x01 (Hop Limit = 1).
6. The TS_UNI_EN bit in the CPSW_P1_TS_CTL_LTYPE2 register is cleared to 0 and Bytes 38 through 53 contain:
   a. FF0M:0:0:0:0:0:0:0181 and the TS_129 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
   b. FF0M:0:0:0:0:0:0:0182 and the TS_130 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
   c. FF0M:0:0:0:0:0:0:0183 and the TS_131 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
   d. FF0M:0:0:0:0:0:0:0184 and the TS_132 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or
   e. FF0M:0:0:0:0:0:0:006B and the TS_107 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set

   NOTE: All values above are 16-bit hex numbers where M is enabled in the TS_MCAST_TYPE_EN field in the CPSW_P1_TS_CTL2 register.

   -OR-

   The TS_UNI_EN bit in the CPSW_P1_TS_CTL_LTYPE2 register is set to 1 and Bytes 38 through 53 contain any value.
7. Bytes 56 and 57 contain (UDP Header in bytes 54 through 61):

a. Decimal 0x01 and 0x3f respectively and the TS_319 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set, or

b. Decimal 0x01 and 0x40 respectively and the TS_320 bit in the CPSW_P1_TS_CTL_LTYPE2 register is set.

8. The PTP message begins in byte 62.
9. The packet message type is enabled in the MSG_TYPE_EN field in the CPSW_P1_TS_CTL register.
10. The packet was sent by host port 0.

Annex F (Ethernet Transmit)

1. Transmit Annex F time sync is enabled (TS_TX_ANNEX_F_EN is set in the switch CPSW_P1_TS_CTL register).
2. One of the sequences below is true:
   a. The first packet LTYPE matches TS_LTYPE1 in the CPSW_P1_TS_SEQ_LTYPE register. LTYPE 1 should be used when only one time sync LTYPE is to be enabled.
   b. The first packet LTYPE matches TS_LTYPE2 in the CPSW_P1_TS_CTL_LTYPE2 register and LTYPE2_EN is set in the CPSW_P1_TS_CTL register.
   c. The first packet LTYPE matches TS_VLAN_LTYPE1 in the CPSW_P1_TS_VLAN_LTYPE register and RX_VLAN_LTYPE1_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches TS_LTYPE1 in the CPSW_P1_TS_SEQ_LTYPE register
   d. The first packet LTYPE matches TS_VLAN_LTYPE1 in the CPSW_P1_TS_VLAN_LTYPE register and TX_VLAN_LTYPE1_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches TS_LTYPE2 in the CPSW_P1_TS_CTL_LTYPE2 register and LTYPE2_EN is set in the CPSW_P1_TS_CTL register.
   e. The first packet LTYPE matches TS_VLAN_LTYPE2 in the CPSW_P1_TS_VLAN register and TX_VLAN_LTYPE2_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches TS_LTYPE1 in the CPSW_P1_TS_SEQ_LTYPE register.
   f. The first packet LTYPE matches TS_VLAN_LTYPE2 in the CPSW_P1_TS_VLAN register and TX_VLAN_LTYPE2_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches TS_LTYPE2 in the CPSW_P1_TS_CTL_LTYPE2 register and LTYPE2_EN is set in the CPSW_P1_TS_CTL register.
   g. The first packet LTYPE matches TS_VLAN_LTYPE1 in the CPSW_P1_TS_VLAN register and TX_VLAN_LTYPE1_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches TS_VLAN_LTYPE2 in the CPSW_P1_TS_VLAN_LTYPE register and TX_VLAN_LTYPE2_EN is set in the CPSW_P1_TS_CTL register and the third packet LTYPE matches TS_LTYPE1 in the CPSW_P1_TS_SEQ_LTYPE register.
   h. The first packet LTYPE matches TS_VLAN_LTYPE1 in the CPSW_P1_TS_VLAN register and TX_VLAN_LTYPE1_EN is set in the CPSW_P1_TS_CTL register and the second packet LTYPE matches TS_VLAN_LTYPE2 in the CPSW_P1_TS_VLAN_LTYPE register and TX_VLAN_LTYPE2_EN is set in the CPSW_P1_TS_CTL register and the third packet LTYPE matches TS_LTYPE2 in the CPSW_P1_TS_CTL_LTYPE2 register and LTYPE2_EN is set in the CPSW_P1_TS_CTL register
3. The packet message type is enabled in the MSG_TYPE_EN field in the CPSW_P1_TS_CTL register.
4. The packet was sent by host port 0.

If all of the criteria described above are met, and the packet is determined to be a valid time synchronization packet, then the time stamp for the transmit event will not be generated until the start of frame delimiter of the packet is actually transmitted. The start of frame delimiter will be sampled on every rising and falling edge of the CPTS_RCLK. Once the packet is transmitted, then the TX interface will push an Ethernet transmit event into the event FIFO.

Table 11-2033. Values of Message Type Field

Message Type	Value (hex)
Sync	0
Delay_Req	1

Table 11-2033. Values of Message Type Field (continued)

Message Type	Value (hex)
Pdelay_Req	2
Pdelay_Resp	3
Reserved	4:7
Follow_Up	8
Delay_Resp	9
Pdelay_Resp_Follow_Up	A
Announce	B
Signaling	C
Management	D
Reserved	E:F

Once a transmitted or received packet is determined to be a valid time sync packet, the Ethernet Transmit Event or Ethernet Receive Event is loaded onto the Event FIFO.

The CPTS_EVENT_1 register contains the Message Type and Sequence ID values from the original time sync packet. The CPTS_EVENT_0 (and CPTS_EVENT_3) register contains the time stamp value when the packet arrived at the corresponding port.

11.13.4.4.7.9 Timestamp Compare Event

NOTE: Timestamp compare events are generated for non-toggle mode only.

The CPTS can generate an event for a time stamp comparison in 32-bit or 64-bit mode. The TS_COMP output is also asserted when the event is generated. The event is generated when the TIME_STAMP value compares with the CPTS_TS_COMP_LOW_VAL/CPTS_TS_COMP_HIGH_VAL register and the CPTS_TS_COMP_LEN value is non-zero. The CPTS_TS_COMP_LEN value should be written by software after the CPTS_TS_COMP_LOW_VAL/CPTS_TS_COMP_HIGH_VAL register(s) is written and should be zero when the comparison value is written.

11.13.4.4.7.10 Host Event

The host can send a packet to be transmitted on an Ethernet port that will generate a time synchronization event. The host sets the TIMESTAMP_EN bit and sends the DOMAIN, MSG_TYPE, and SEQUENCE_ID in the additional control information that resides in the protocol specific section of the descriptor that is transmitted to the CPSW_2U (see *Receive Streaming Packet Interface (Host Port 0 Switch Ingress)*). An event is then generated and placed on the event FIFO once the packet is transmitted. Host events allow the user to timestamp exactly when a software generated packet exits the device.

11.13.4.4.7.11 CPTS Interrupt Handling

When an event is push onto the Event FIFO, an interrupt can be generated to indicate to software that a time sync event occurred. The following steps should be taken to process time sync events using interrupts:

1. Enable the TS_PEND interrupt by setting the TS_PEND_EN bit of the CPTS_INT_ENABLE register.
2. Upon interrupt, read the CPTS_EVENT_0 through CPTS_EVENT_3 registers values.
3. Set the CPTS_EVENT_POP[0] bit to 1 to pop the previously read value off of the event FIFO.
4. Process the interrupt as required by the application software.

Software has the option of processing more than a single event from the event FIFO in the interrupt service routine in the following way:

1. Enable the TS_PEND interrupt by setting the TS_PEND_EN bit of the CPTS_INT_ENABLE register.
2. Upon interrupt, read the CPTS_EVENT_0 through CPTS_EVENT_3 registers values.

3. Set the CPTS_EVENT_POP[0] bit to 1 to pop the previously read value off of the event FIFO.
4. Wait for an amount of time greater than four CPTS_RFTCLK periods plus four VBUSP_GCLK periods.
5. Read the TS_PEND_RAW bit in the CPTS_INTSTAT_RAW register to determine if another valid event is in the event FIFO. If it is asserted, go to step 2; otherwise, proceed to step 6.
6. Process the interrupt(s) as required by the application software.

Software also has the option of disabling the interrupt and polling the TS_PEND_RAW bit of the CPTS_INTSTAT_RAW register to determine if a valid event is on the event FIFO.

11.13.4.4.8 CPPI Packet Streaming Interface

The receive streaming interface on port 0 of the CPSW_2U is responsible for receiving packets from the packet streaming switch in the NAVSS. The CPSW_2U has one receive streaming packet interface for port 0. The CPPI receive port is equivalent to a MAC port with the difference being that the data is provided to the CPSW_2U in the 128-bit streaming interface data format instead of 8-bit GMII data format.

In addition to the packet data, the receive streaming interface also can provide additional control information that resides in the information words of the descriptor that was transmitted to the CPSW_2U.

The tables below show the information that may be passed along with which descriptor information word to put it in. See Section 11.13.2, *Navigator Subsystem* as well as the descriptor structure definitions in the CPPI LLD (Low Level Driver) for more information on these bits.

11.13.4.4.8.1 Port 0 CPPI Transmit Packet Streaming Interface (CPSW_2U Egress)

The CPSW_2U makes use of CPPI descriptor fields to communicate additional information about the packet being transmitted by the switch (and being received by the DMA). Table 11-2034 details the CPSW_2U information field, and corresponding CPPI descriptor field, and the information conveyed.

Table 11-2034. Port 0 CPPI Transmit Packet Streaming Interface

CPPI Descriptor Field	Contents	Description
PS Flags Bit 3	PASSED_CRC	This bit is cleared to zero (no CRC passed) when the P0_TX_CRC_REMOVE bit in the CPSW_CONTROL register is set (and the egress packet has no errors). When the remove bit is cleared to zero then this bit is cleared and no CRC is passed with the output packet. The packet length includes the CRC if it is present.
PS Flags Bit 2	CRC_TYPE	CRC Type – The packet CRC type. The type of CRC passed is determined by the P0_TX_CRC_TYPE bit in the CPSW_CONTROL register (not by the type of CRC the packet had on Ethernet port ingress). 0 – Ethernet CRC 1 – Castagnoli CRC
Configurable	SRC_ID	The packet SRC_ID value comes from the PN_SRC_ID field in the CPSW_P0_SRC_ID_A register. This field can be optionally written to the packet descriptor in the source or destination tag, based on DMA channel configuration.
Configurable	FLOW_ID	The default flow ID is the remapped received packet priority plus the FLOW_ID_OFFSET. The default flow ID can be overridden by ALE classification match. This is also called “flow index” by the DMA. This field can be optionally written to the packet descriptor in the source or destination tag, based on DMA channel configuration.
Timestamp Info	TIMESTAMP[31:0]	64-bit Timestamp Low
Software Info 0	TIMESTAMP[63:32]	64-bit Timestamp High

11.13.4.4.8.2 CPPI Receive Packet Streaming Interface (CPSW_2U Ingress)

The CPSW_2U makes use of CPPI descriptor fields to communicate additional information about the packet being sent by the DMA (and being received by the CPSW_2U). Table 11-2035 details the CPSW_2U information field, and corresponding CPPI descriptor field, and the information conveyed.

Table 11-2035. Port 0 CPPI Transmit Packet Streaming Interface

CPPI Descriptor Field	Contents	Description
PS Flags Bit 3	PASSED_CRC	The PASSED_CRC bit indicates that the CRC is passed with the packet data. 0 - CRC is not passed with packet (CRC_TYPE is don't care) 1 - CRC of type CRC_TYPE is passed with the packet.
PS Flags Bit 2	CRC_TYPE	Must be set to zero when PASSED_CRC is set.
Dest_Tag[4:0]	TO_PORT	Port number to send the directed packet to. This field is set by the host. Directed packets go to the directed port, but an ALE lookup is performed to determine untagged egress in VLAN_AWARE mode. 0 - Not directed 1 - Send the packet to port 1 2 - Send the packet to port 2
Software Info 0 [31]	TIMESTAMP_EN	Software Info 0 [31] TIMESTAMP_EN When set, this bit indicates that the packet will generate a timesync event on Ethernet egress (if the CPTS is configured properly) with the associated DOMAIN, MSG_TYPE, and SEQUENCE_ID.
Software Info 0 [27:20]	DOMAIN	Timesync domain
Software Info 0 [19:16]	MSG_TYPE	Timesync message type
Software Info 0 [15:0]	SEQUENCE_ID	Timesync sequence ID

11.13.4.4.9 MII Management Interface (MDIO)

The MII Management interface module implements the 802.3 serial management interface to interrogate and control external Ethernet PHY using a two-wire bus.

11.13.4.4.9.1 MDIO Frame Formats

Table 11-2036 shows the read format and Table 11-2037 shows the write format of the 32-bit MII Management interface frames.

Table 11-2036. MDIO Read Frame Format

Pre-amble	Start Delimiter	Operation Code	PHY Address	Register Address	Turnaround	Data
FFFF FFFFh	01	10	AAAAA	RRRRR	Z0	DDDD.DDDD.DDDD.DDDD

Table 11-2037. MDIO Write Frame Format

Pre-amble	Start Delimiter	Operation Code	PHY Address	Register Address	Turnaround	Data
FFFF FFFFh	01	01	AAAAA	RRRRR	10	DDDD.DDDD.DDDD.DDDD

The default or idle state of the two wire serial interface is a logic one. All tri-state drivers should be disabled and the PHY's pull-up resistor will pull the MDIO line to a logic 1. Prior to initiating any other transaction, the station management entity shall send a preamble sequence of 32 contiguous logic 1 bits on the MDIO line with 32 corresponding cycles on MDCLK to provide the PHY with a pattern that it can use to establish synchronization. A PHY shall observe a sequence of 32 contiguous logic one bits on MDIO with 32 corresponding MDCLK cycles before it responds to any other transaction.

Preamble

The start of a frame is indicated by a preamble, which consists of a sequence of 32 contiguous bits all of which are a 1. This sequence provides the PHY a pattern to use to establish synchronization.

Start Delimiter

The preamble is followed by the start delimiter which is indicated by a 01 pattern. The pattern assures transitions from the default logic 1 state to logic 0, and back to logic 1.

Operation Code

The operation code for a read is 10, while the operation code for a write is a 01.

PHY Address

The PHY address is 5 bits allowing 32 unique values. The first bit transmitted is the MSB of the PHY address.

Register Address

The Register address is 5 bits allowing 32 registers to be addressed within each PHY. Refer to the 10/100 PHY address map for addresses of individual registers.

Turnaround

An idle bit time during which no device actively drives the MDIO signal shall be inserted between the register address field and the data field of a read frame in order to avoid contention. During a read frame, the PHY shall drive a zero bit onto MDIO for the first bit time following the idle bit and preceding the Data field. During a write frame, this field shall consist of a one bit followed by a zero bit.

Data

The Data field is 16 bits. The first bit transmitted and received is the MSB of the data word.

11.13.4.4.9.2 MDIO Functional Description

The MII Management I/F will remain idle until enabled by setting the ENABLE bit in the MDIO_CONTROL register. The MII Management I/F will then continuously poll the link status from within the Generic Status Register of all possible 32 PHY addresses in turn recording the results in the MDIO_LINK register. The LINKSEL bit in the MDIO_USER_PHY_SEL_0 register determines the status input that is used. A change in the link status of the two PHYs being monitored will set the appropriate bit in the MDIO_LINK_INT_RAW register and the MDIO_LINK_INT_MASKED register, if enabled by the LINKINT_ENABLE bit in the MDIO_USER_PHY_SEL_0 register.

The MDIO_ALIVE register is updated by the MII Management I/F module if the PHY acknowledged the read of the generic status register. In addition, any PHY register read transactions initiated by the host also cause the MDIO_ALIVE register to be updated.

At any time, the host can define a transaction for the MII Management interface module to undertake using the DATA, PHYADR, REGADR, and WRITE fields in a MDIO_USER_ACCESS_0 register. When the host sets the GO bit in this register, the MII Management interface module will begin the transaction without any further intervention from the host. Upon completion, the MII Management interface will clear the GO bit and set the USERINTRAW bit in the MDIO_USER_INT_RAW register corresponding to the MDIO_USER_ACCESS_0 register being used. The corresponding bit in the MDIO_USER_INT_MASKED register may also be set depending on the mask setting in the MDIO_USER_INT_MASK_SET and MDIO_USERINTMASKCLR registers. A round-robin arbitration scheme is used to schedule transactions that may be queued by the host in different MDIO_USER_ACCESS_0 registers. The host should check the status of the GO bit in the MDIO_USER_ACCESS_0 register before initiating a new transaction to ensure that the previous transaction has completed. The host can use the ACK bit in the MDIO_USER_ACCESS_0 register to determine the status of a read transaction.

It is necessary for software to use the MII Management interface module to setup the auto-negotiation parameters of each PHY attached to a MAC port, retrieve the negotiation results, and setup the CPSW_P1_MAC_CONTROL register in the corresponding MAC.

11.13.4.5 EMAC Programming Guide

11.13.4.5.1 Initialization and Configuration of EMAC Subsystem

To configure the EMAC Ethernet Subsystem for operation, the host must perform the following:

1. Select the Interface (G/MII, RGMII, or RMII) Mode. See the ETHERNET_CFG register.
2. Configure pads (pin muxing), as per the interface selected. Refer to Section 5.1.3.1.1, *Pad Configuration Registers* and the device data manual.
3. Lock the NSS PLL if in bypass. Enable the EMAC Ethernet Subsystem clocks. Refer to Section 5.4, *Clock Management* and Section 5.2, *Power Management*
4. Apply Soft Reset to CPGMAC_SL. See CPSW_P1_MAC_SOFT_RESET register.
5. Ensure that at least 2000 VBUSP_CLK periods are run after reset is de-asserted.
6. Configure the CPSW_CONTROL register
7. Configure the Ethernet Port Source Address registers (CPSW_P1_SA_L and CPSW_P1_SA_H)
8. Configure the CPSW_STAT_PORT_EN register
9. Configure the ALE (Section 11.13.4.4.6.1, *Address Lookup Engine*)
10. Configure the MDIO (Section 11.13.4.5.3.1, *Initializing the MDIO Module*)
11. Configure Navigator Subsystem
12. Configure CPGMAC_SL, as per the desired mode of operations

11.13.4.5.2 Ethernet MAC Reset

To reset the CPMAC_SL, the host must perform the following:

1. Set CMD_IDLE in the Ethernet port CPSW_P1_MAC_CONTROL register
2. Wait for Idle to be indicated in the Ethernet port CPSW_P1_MAC_STATUS register
3. Set SOFT_RESET in the Ethernet port CPSW_P1_MAC_SOFT_RESET register
4. Wait for SOFT_RESET in the CPSW_P1_MAC_SOFT_RESET to be cleared to confirm reset completion
5. Configure the Ethernet ports.

11.13.4.5.3 MDIO Software Interface

11.13.4.5.3.1 Initializing the MDIO Module

The following steps are performed by the application software or device driver to initialize the MDIO device:

1. Configure the PREAMBLE and CLKDIV bits in the MDIO Control register (MDIO_CONTROL).
2. Enable the MDIO module by setting the ENABLE bit in MDIO_CONTROL.
3. The MDIO PHY alive status register (MDIO_ALIVE) can be read in polling fashion until a PHY connected to the system responded, and the MDIO PHY link status register (MDIO_LINK) can determine whether this PHY already has a link.
4. Setup the appropriate PHY addresses in the MDIO user PHY select register (MDIO_USER_PHY_SEL_0/1), and set the LINKINTENB bit to enable a link change event interrupt if desirable.
5. If an interrupt on general MDIO register access is desired, set the corresponding bit in the MDIO user command complete interrupt mask set register (MDIO_USER_INT_MASK_SET) to use the MDIO user access register (MDIO_USER_ACCESS_0/1). Since only one PHY is used in this device, the application software can use one MDIO_USER_ACCESS_0/1 to trigger a completion interrupt; the other MDIO_USER_ACCESS_0/1 is not setup.

11.13.4.5.3.2 Writing Data To a PHY Register

The MDIO module includes a user access register (MDIO_USER_ACCESS_0/1) to directly access a specified PHY device.To write a PHY register, perform the following:

1. Check to ensure that the GO bit in the MDIO user access register (MDIO_USER_ACCESS_0/1) is cleared.
2. Write to the GO, WRITE, REGADR, PHYADR, and DATA bits in MDIO_USER_ACCESS_0/1 corresponding to the PHY and PHY register software wants to write.
3. The write operation to the PHY is scheduled and completed by the MDIO module. Completion of the write operation can be determined by polling the GO bit in MDIO_USER_ACCESS_0/1 for a 0.
4. Completion of the operation sets the corresponding USERINTRAW bit (0 or 1) in the MDIO user command complete interrupt register (MDIO_USER_INT_RAW) corresponding to MDIO_USER_ACCESS_0/1 used. If interrupts have been enabled on this bit using the MDIO user command complete interrupt mask set register (MDIO_USER_INT_MASK_SET), then the bit is also set in the MDIO user command complete interrupt register (MDIO_USER_INT_MASKED) and an interrupt is triggered on the host processor.

11.13.4.5.3.3 Reading Data From a PHY Register

The MDIO module includes a user access register (MDIO_USER_ACCESS_0/1) to directly access a specified PHY device. To read a PHY register, perform the following:

1. Check to ensure that the GO bit in the MDIO user access register (MDIO_USER_ACCESS_0*n*) is cleared.
2. Write to the GO, REGADR, and PHYADR bits in MDIO_USER_ACCESS_0/1 corresponding to the PHY and PHY register software wants to read.
3. The read data value is available in the DATA bits in MDIO_USER_ACCESS_0/1 after the module completes the read operation on the serial bus. Completion of the read operation can be determined by polling the GO and ACK bits in MDIO_USER_ACCESS_0/1. After the GO bit has cleared, the ACK bit is set on a successful read.
4. Completion of the operation sets the corresponding USERINTRAW bit (0 or 1) in the MDIO user command complete interrupt register (MDIO_USER_INT_RAW) corresponding to MDIO_USER_ACCESS_0/1 used. If interrupts have been enabled on this bit using the MDIO user command complete interrupt mask set register (MDIO_USER_INT_MASK_SET), then the bit is also set in the MDIO user command complete interrupt register (MDIO_USER_INT_MASKED) and an interrupt is triggered on the host processor.

11.13.5 NSS Registers

For NSS top level memory map, please refer to Section 11.13.1.3, *NSS Memory Map*.

11.13.5.1 EMAC Registers

11.13.5.1.1 NSS_0_CFG_EMAC Registers

Table 11-2039 lists the memory-mapped registers for the NSS_0_CFG_EMAC. All register offset addresses not listed in Table 11-2039 should be considered as reserved locations and the register contents should not be modified.

Table 11-2038. NSS_0_CFG_EMAC Instances

Instance	Base Address
NSS_0_CFG_EMAC	0420 0000h

Table 11-2039. NSS_0_CFG_EMAC Registers

Offset	Acronym	Register Name	NSS_0_CFG_EMAC Physical Address	Section
0h	SS_IDVER	Subsystem ID Version Register	0420 0000h	Section 11.13.5.1.1.1
4h	RESERVED		0420 0004h	
8h	RESERVED		0420 0008h	
Ch	SS_CONTROL	Subsystem Control Register	0420 000Ch	Section 11.13.5.1.1.2
18h	SS_RGMII_STATUS	RGMII Status Register	0420 0018h	Section 11.13.5.1.1.3
1Ch	SS_SUBSSYSTEM_STATUS	Subsystem Status Register	0420 001Ch	Section 11.13.5.1.1.4

11.13.5.1.1.1 SS_IDVER Register (Offset = 0h) [reset = 4EE8 3100h]

SS_IDVER is shown in Figure 11-925 and described in Table 11-2041.

Table 11-2040. SS_IDVER Instances

Instance	Physical Address
NSS_0_CFG_EMAC	0420 0000h

Figure 11-925. SS_IDVER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 4EE8 3100h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2041. SS_IDVER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4EE71100h	TI internal data. Identifies revision of peripheral.

Table 11-2042. Register Call Summary for SS_IDVER

EMAC Registers
- NSS_0_CFG_EMAC Registers: [0]
- SS_IDVER Register (Offset = 0h) [reset = 4EE8 3100h]: [0]

11.13.5.1.1.2 SS_CONTROL Register (Offset = Ch) [reset = 0h]

SS_CONTROL is shown in Figure 11-926 and described in Table 11-2044.

Table 11-2043. SS_CONTROL Instances

Instance	Physical Address
NSS_0_CFG_EMAC	0420 000Ch

Figure 11-926. SS_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						EEE_PHY_ONLY	EEE_EN
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2044. SS_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	EEE_PHY_ONLY	R/W	0h	Energy Efficient Enable Phy Only Mode: • 0=The low power indicate state includes gating off the CPPI_GCLK to the CPSW • 1=The low power indicate state does not gate the clock to the CPSW
0	EEE_EN	R/W	0h	Energy Efficient Ethernet Enable: • 0=EEE is disabled • 1=EEE is enabled

Table 11-2045. Register Call Summary for SS_CONTROL

Gigabit Ethernet MAC (EMAC) Subsystem • Energy Efficient Ethernet Support (802.3az): [0]
EMAC Registers • NSS_0_CFG_EMAC Registers: [0] • SS_CONTROL Register (Offset = Ch) [reset = 0h]: [0]

11.13.5.1.1.3 SS_RGMII_STATUS Register (Offset = 18h) [reset = 0h]

SS_RGMII_STATUS is shown in Figure 11-927 and described in Table 11-2047.

Table 11-2046. SS_RGMII_STATUS Instances

Instance	Physical Address
NSS_0_CFG_EMAC	0420 0018h

Figure 11-927. SS_RGMII_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				FULLDUPLEX	SPEED		LINK
R-0h				R-0h	R-0h		R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2047. SS_RGMII_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	FULLDUPLEX	R	0h	Rgmii full dulex: • 0 = Half-duplex • 1 = Full-duplex
2-1	SPEED	R	0h	Rgmii speed: • 00 = 10Mbps • 01 = 100Mbps • 10 = 1000Mbps • 11 = RESERVED
0	LINK	R	0h	Rgmii link indicator: • 0 = Link is down • 1 = Link is up

Table 11-2048. Register Call Summary for SS_RGMII_STATUS

EMAC Registers
- NSS_0_CFG_EMAC Registers: [0]
- SS_RGMII_STATUS Register (Offset = 18h) [reset = 0h]: [0]

11.13.5.1.1.4 SS_SUBSSYSTEM_STATUS Register (Offset = 1Ch) [reset = 0h]

SS_SUBSSYSTEM_STATUS is shown in Figure 11-928 and described in Table 11-2050.

Table 11-2049. SS_SUBSSYSTEM_STATUS Instances

Instance	Physical Address
NSS_0_CFG_EMAC	0420 001Ch

Figure 11-928. SS_SUBSSYSTEM_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EEE_CLKSTOP_ACK
R-0h							R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2050. SS_SUBSSYSTEM_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EEE_CLKSTOP_ACK	R	0h	Energy Efficient Ethernet clockstop acknowledge from CPSW_2U

Table 11-2051. Register Call Summary for SS_SUBSSYSTEM_STATUS

EMAC Registers
- NSS_0_CFG_EMAC Registers: [0]
- SS_SUBSSYSTEM_STATUS Register (Offset = 1Ch) [reset = 0h]: [0]

11.13.5.1.2 NSS_0_CFG_CPSW Registers

lists the memory-mapped registers for the NSS_0_CFG_CPSW. All register offset addresses not listed in should be considered as reserved locations and the register contents should not be modified.

Table 11-2052. NSS_0_CFG_CPSW Instances

Instance	Base Address
NSS_0_CFG_CPSW	0422 0000h

Table 11-2053. NSS_0_CFG_CPSW Registers

Offset	Acronym	Register Name	NSS_0_CFG_CPSW Physical Address	Section
0h	CPSW_IDVER	CPSW ID Version	0422 0000h	Section 11.13.5.1.2.1
4h	CPSW_CONTROL	CPSW Switch Control	0422 0004h	Section 11.13.5.1.2.2
10h	CPSW_EM_CONTROL	CPSW Emulation Control	0422 0010h	Section 11.13.5.1.2.3
14h	CPSW_STAT_PORT_EN	CPSW Statistics Port Enable	0422 0014h	Section 11.13.5.1.2.4
1Ch	CPSW_SOFT_IDLE	CPSW Software Idle	0422 001Ch	Section 11.13.5.1.2.5
2Ch	CPSW_EEE_PRESCALE	CPSW Energy Efficient Ethernet Prescale Value	0422 002Ch	Section 11.13.5.1.2.6
1004h	CPSW_P0_CONTROL	CPPI Port 0 Control	0422 1004h	Section 11.13.5.1.2.7
1008h	CPSW_P0_FLOW_ID_OFFSET	CPPI Port 0 Flow ID Offset	0422 1008h	Section 11.13.5.1.2.8
1010h	CPSW_P0_BLK_CNT	CPPI Port 0 FIFO Block Usage Count	0422 1010h	Section 11.13.5.1.2.9
1014h	CPSW_P0_PORT_VLAN	CPPI Port 0 VLAN	0422 1014h	Section 11.13.5.1.2.10
101Ch	CPSW_P0_PRI_CTL	CPPI Port 0 Priority Control	0422 101Ch	Section 11.13.5.1.2.11
1020h	CPSW_P0_RX_PRI_MAP	CPPI Port 0 RX Pkt Pri to Header Pri Map	0422 1020h	Section 11.13.5.1.2.12
1024h	CPSW_P0_RX_MAXLEN	CPPI Port 0 Receive Frame Max Length	0422 1024h	Section 11.13.5.1.2.13
1030h	CPSW_P0_IDLE2LPI	Port 0 EEE Idle to LPI counter	042 1030h	Section 11.13.5.1.2.14
1034h	CPSW_P0_LPI2WAKE	Port 0 EEE LPI to wake counter	0422 1034h	Section 11.13.5.1.2.15
1038h	CPSW_P0_EEE_STATUS	Port 0 EEE status	0422 1038h	Section 11.13.5.1.2.16
1120h to 113Ch	CPSW_P0_RX_DSCP_MAP_0 to CPSW_P0_RX_DSCP_MAP_7	CPPI Port 0 Receive IPV4/IPV6 DSCP Map N	0422 1120h to 0422 113Ch	Section 11.13.5.1.2.17
1140h to 115Ch	CPSW_P0_PRI_SEND_0 to CPSW_P0_PRI_SEND_7	CPPI Port 0 Rx Priority P Send Count Value	0422 1140h to 0422 115Ch	Section 11.13.5.1.2.18
1160h to 117Ch	CPSW_P0_PRI_IDLE_0 to CPSW_P0_PRI_IDLE_7	CPPI Port 0 Rx Priority P Idle Count Value	0422 1160h to 0422 117Ch	Section 11.13.5.1.2.19
1300h	CPSW_P0_SRC_ID_A	CPPI Port 0 CPPI Source ID A	0422 1300h	Section 11.13.5.1.2.20
2000h	CPSW_P1_RESERVED	Reserved	0422 2000h	Section 11.13.5.1.2.21

Table 11-2053. NSS_0_CFG_CPSW Registers (continued)

Offset	Acronym	Register Name	NSS_0_CFG_CPSW Physical Address	Section
2004h	CPSW_P1_CONTROL	Enet Port 1 Control	0422 2004h	Section 11.13.5.1.2.22
2008h	CPSW_P1_MAX_BLKS	Enet Port 1 FIFO Max Blocks	0422 2008h	Section 11.13.5.1.2.23
2010h	CPSW_P1_BLK_CNT	Enet Port 1 FIFO Block Usage Count	0422 2010h	Section 11.13.5.1.2.24
2014h	CPSW_P1_PORT_VLAN	Enet Port 1 VLAN	0422 2014h	Section 11.13.5.1.2.25
201Ch	CPSW_P1_PRI_CTL	Enet Port 1 Priority Control	0422 201Ch	Section 11.13.5.1.2.26
2020h	CPSW_P1_RX_PRI_MAP	Enet Port 1 RX Pkt Pri to Header Pri Map	0422 2020h	Section 11.13.5.1.2.27
2024h	CPSW_P1_RX_MAXLEN	Enet Port 1 Receive Frame Max Length	0422 2024h	Section 11.13.5.1.2.28
2030h	CPSW_P1_IDLE2LPI	Enet Port 1 EEE Idle to LPI counter	0422 2030h	Section 11.13.5.1.2.29
2034h	CPSW_P1_LPI2WAKE	Enet Port 1 EEE LPI to wake counter	0422 2034h	Section 11.13.5.1.2.30
2038h	CPSW_P1_EEE_STATUS	Enet Port 1 EEE status	0422 2038h	Section 11.13.5.1.2.31
2120h to 214Ch	CPSW_P1_RX_DSCP_MAP_0 to CPSW_P1_RX_DSCP_MAP_7	Enet Port 1 Receive IPV4/IPV6 DSCP Map M	0422 2120h to 0422 214Ch	Section 11.13.5.1.2.32
2140h to 215Ch	CPSW_P1_PRI_SEND_0 to CPSW_P1_PRI_SEND_7	Enet Port 1 Rx Priority P Send Count Value	0422 2140h to 0422 215Ch	Section 11.13.5.1.2.33
2160h to 217Ch	CPSW_P1_PRI_IDLE_0 to CPSW_P1_PRI_IDLE_7	Enet Port 1 Rx Priority P Idle Count Value	0422 2160h to 0422 217Ch	Section 11.13.5.1.2.34
2308h	CPSW_P1_SA_L	Enet Port 1 Tx Pause Frame Source Address Low	0422 2308h	Section 11.13.5.1.2.35
230Ch	CPSW_P1_SA_H	Enet Port 1 Tx Pause Frame Source Address High	0422 230Ch	Section 11.13.5.1.2.36
2310h	CPSW_P1_TS_CTL	Enet Port 1 Time Sync Control	0422 2310h	Section 11.13.5.1.2.37
2314h	CPSW_P1_TS_SEQ_LTYPE	Enet Port 1 Time Sync LTYPE (and SEQ_ID_OFFSET)	0422 2314h	Section 11.13.5.1.2.38
2318h	CPSW_P1_TS_VLAN_LTYPE	Enet Port 1 Time Sync VLAN2 and VLAN2	0422 2318h	Section 11.13.5.1.2.39
231Ch	CPSW_P1_TS_CTL_LTYPE2	Enet Port 1 Time Sync Control and LTYPE 2	0422 231Ch	Section 11.13.5.1.2.40
2320h	CPSW_P1_TS_CTL2	Enet Port 1 Time Sync Control 2	0422 2320h	Section 11.13.5.1.2.41
2330h	CPSW_P1_MAC_CONTROL	Enet Port 1 Mac Control	0422 2330h	Section 11.13.5.1.2.42
2334h	CPSW_P1_MAC_STATUS	Enet Port 1 Mac Status	0422 2334h	Section 11.13.5.1.2.43
2338h	CPSW_P1_MAC_SOFT_RESET	Enet Port 1 Mac Soft Reset	0422 2338h	Section 11.13.5.1.2.44
233Ch	CPSW_P1_MAC_BOFFTEST	Enet Port 1 Mac Backoff Test	0422 233Ch	Section 11.13.5.1.2.45
2340h	CPSW_P1_MAC_RX_PAUSETIMER	Enet Port 1 802.3 Receive Pause Timer	0422 2340h	Section 11.13.5.1.2.46

Table 11-2053. NSS_0_CFG_CPSW Registers (continued)

Offset	Acronym	Register Name	NSS_0_CFG_CPSW Physical Address	Section
2370h	CPSW_P1_MAC_TX_PAUSETIMER	Enet Port 1 802.3 Tx Pause Timer	0422 2370h	Section 11.13.5.1.2.47
23A0h	CPSW_P1_MAC_EMCONTROL	Enet Port 1 Emulation Control	0422 23A0h	Section 11.13.5.1.2.48
23A4h	CPSW_P1_MAC_TX_GAP	Enet Port 1 Tx Inter Packet Gap	0422 23A4h	Section 11.13.5.1.2.49

11.13.5.1.2.1 CPSW_IDVER Register (Offset = 0h) [reset = 4EE71100h]

CPSW_IDVER is shown in and described in Table 11-2055.

CPSW ID Version

Table 11-2054. CPSW_IDVER Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 0000h

Figure 11-929. CPSW_IDVER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 4EE71100h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2055. CPSW_IDVER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4EE71100h	TI internal data. Identifies revision of peripheral.

Table 11-2056. Register Call Summary for CPSW_IDVER

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_IDVER Register (Offset = 0h) [reset = 4EE71100h]: [0]

11.13.5.1.2.2 CPSW_CONTROL Register (Offset = 4h) [reset = 0h]

CPSW_CONTROL is shown in Figure 11-930 and described in Table 11-2058.

CPSW Switch Control

Table 11-2057. CPSW_CONTROL Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 0004h

Figure 11-930. CPSW_CONTROL Register

31	30	29	28	27	26	25	24
ECC_CRC_MODE	RESERVED						
R/W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							EEE_ENABLE
R-0h							R/W-0h
15	14	13	12	11	10	9	8
P0_RX_PASS_CRC_ERR	P0_RX_PAD	P0_TX_CRC_REMOVE	P0_TX_CRC_TYPE	RESERVED			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h			
7	6	5	4	3	2	1	0
RESERVED			P1_PASS_PRI_TAGGED	P0_PASS_PRI_TAGGED	P0_ENABLE	VLAN_AWARE	RESERVED
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2058. CPSW_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ECC_CRC_MODE	R/W	0h	ECC CRC Mode • 0 – ECC errors induced through the ECC aggregator flip bits in the packet headers (not in packet data). • 1 – ECC errors induced through the ECC aggregator flip bits in the packet data (not in the packet headers).
30-17	RESERVED	R	0h	Reserved
16	EEE_ENABLE	R/W	0h	Energy Efficient Ethernet enable • 0 – Energy Efficient Ethernet is disabled • 1 – Energy Efficient Ethernet is enabled
15	P0_RX_PASS_CRC_ERR	R/W	0h	Port 0 Pass Received CRC errors • 0 – Packets received with CRC errors on port 0 are dropped. • 1 – Packets received with CRC errors on port 0 are transferred to the destination ports.
14	P0_RX_PAD	R/W	0h	Port 0 Receive Short Packet Pad • 0 – short packets are dropped. • 1 – short packets are padded to 64-bytes (with pad and added CRC) if the CRC is not passed in. Short packets are dropped if the CRC is passed (in the Info0 word).
13	P0_TX_CRC_REMOVE	R/W	0h	Port 0 Transmit CRC remove • 0 – Do not remove the CRC on Port 0 transmit (egress) packets. • 1 – Remove the CRC on all Port 0 transmit (egress) packets.

Table 11-2058. CPSW_CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	P0_TX_CRC_TYPE	R/W	0h	Port 0 Transmit CRC Type – The type of CRC on all Port 0 transmit packet (egress), regardless of the CRC type of in ingress Ethernet port. • 0 – Ethernet CRC on Port 0 Transmit. • 1 –reserved (Ethernet CRC only on host egress)
11-5	RESERVED	R	0h	RESERVED
4	P1_PASS_PRI_TAGGED	R/W	0h	Port 1 Pass Priority Tagged • 0 – Priority tagged packets have the zero VID replaced with the input port Enet_P1_PORT_VLAN[11:0] on ingress. • 1 – Priority tagged packets are processed unchanged.
3	P0_PASS_PRI_TAGGED	R/W	0h	Port 0 Pass Priority Tagged • 0 – Priority tagged packets have the zero VID replaced with the input port P0_PORT_VLAN[11:0] on ingress. • 1 – Priority tagged packets are processed unchanged.
2	P0_ENABLE	R/W	0h	Port 0 Enable • 0 – CPPI port (port 0) packet operations are disabled • 1 – CPPI port (port 0) packet operations are enabled
1	VLAN_AWARE	R/W	0h	VLAN Aware Mode • 0 – CPSW_2U is in the VLAN unaware mode. • 1 – CPSW_2U is in the VLAN aware mode.
0	RESERVED	R	0h	Reserved

Table 11-2059. Register Call Summary for CPSW_CONTROL

Gigabit Ethernet MAC (EMAC) Subsystem • Packet CRC Handling: [0] • Address Lookup Engine (ALE): [0][1] • Port 0 CPPI Transmit Packet Streaming Interface (CPSW_2U Egress): [0][1] • Energy Efficient Ethernet Support (802.3az): [0] • Initialization and Configuration of EMAC Subsystem: [0]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_CONTROL Register (Offset = 4h) [reset = 0h]: [0]
Memory Error Detection and Correction • Packet Header ECC: [0] • Packet Protect CRC: [0]

11.13.5.1.2.3 CPSW_EM_CONTROL Register (Offset = 10h) [reset = 0h]

CPSW_EM_CONTROL is shown in Figure 11-931 and described in Table 11-2061.

CPSW Emulation Control

Table 11-2060. CPSW_EM_CONTROL Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 0010h

Figure 11-931. CPSW_EM_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						SOFT	FREE
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2061. CPSW_EM_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	SOFT	R/W	0h	Emulation Soft Bit
0	FREE	R/W	0h	Emulation Free Bit

Table 11-2062. Register Call Summary for CPSW_EM_CONTROL

EMAC Registers

- NSS_0_CFG_CPSW Registers: [0]
- CPSW_EM_CONTROL Register (Offset = 10h) [reset = 0h]: [0]

11.13.5.1.2.4 CPSW_STAT_PORT_EN Register (Offset = 14h) [reset = 0h]

CPSW_STAT_PORT_EN is shown in Figure 11-932 and described in Table 11-2064.

CPSW Statistics Port Enable

Table 11-2063. CPSW_STAT_PORT_EN Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 0014h

Figure 11-932. CPSW_STAT_PORT_EN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						P1_STAT_EN	P0_STAT_EN
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2064. CPSW_STAT_PORT_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reads return 0 and writes have no effect.
1	P1_STAT_EN	R/W	0h	Port 1 Statistics Enable • 0 – Port 1 statistics are not enabled • 1 – Port 1 statistics are enabled.
0	P0_STAT_EN	R/W	0h	Port 0 Statistics Enable • 0 – Port 0 statistics are not enabled • 1 – Port 0 statistics are enabled.

Table 11-2065. Register Call Summary for CPSW_STAT_PORT_EN

Gigabit Ethernet MAC (EMAC) Subsystem • CPSW_2U Network Statistics: [0] • Initialization and Configuration of EMAC Subsystem: [0]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_STAT_PORT_EN Register (Offset = 14h) [reset = 0h]: [0]

11.13.5.1.2.5 CPSW_SOFT_IDLE Register (Offset = 1Ch) [reset = 0h]

CPSW_SOFT_IDLE is shown in Figure 11-933 and described in Table 11-2067.

CPSW Software Idle

Table 11-2066. CPSW_SOFT_IDLE Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 001Ch

Figure 11-933. CPSW_SOFT_IDLE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							SOFT_IDLE
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2067. CPSW_SOFT_IDLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	SOFT_IDLE	R/W	0h	Software Idle • 0 – not in idle • 1 – Command a CPSW_2U software idle. When set, any packet currently being transmtted or received will be completed but no new packets will be started. Any packet received on an Ethernet port while IDLE will be dropped with no statistics taken.

Table 11-2068. Register Call Summary for CPSW_SOFT_IDLE

Gigabit Ethernet MAC (EMAC) Subsystem • Software IDLE: [0]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_SOFT_IDLE Register (Offset = 1Ch) [reset = 0h]: [0]

11.13.5.1.2.6 CPSW_EEE_PRESCALE Register (Offset = 2Ch) [reset = 0h]

CPSW_EEE_PRESCALE is shown in Figure 11-934 and described in Table 11-2070.

CPSW Energy Efficient Ethernet Prescale Value

Table 11-2069. CPSW_EEE_PRESCALE Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 002Ch

Figure 11-934. CPSW_EEE_PRESCALE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				EEE_PRESCALE											
R-0h																				R/W-0h											

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2070. CPSW_EEE_PRESCALE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11-0	EEE_PRESCALE	R/W	0h	Energy Efficient Ethernet Pre-scale count load value - This value is loaded into the EEE pre-scale counter each time the pre-scale count decrements to zero. The EEE counters are enabled to decrement each time the pre-scale counter reaches zero (and the EEE counters are enabled to count time). If this value is zero then the EEE counters decrement on every clock. If this value is 0x001 then the counters decrement on every other clock (and so on).

Table 11-2071. Register Call Summary for CPSW_EEE_PRESCALE

Gigabit Ethernet MAC (EMAC) Subsystem • Energy Efficient Ethernet Support (802.3az): [0]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_EEE_PRESCALE Register (Offset = 2Ch) [reset = 0h]: [0]

11.13.5.1.2.7 CPSW_P0_CONTROL Register (Offset = 1004h) [reset = 0h]

CPSW_P0_CONTROL is shown in Figure 11-935 and described in Table 11-2073.

CPPI Port 0 Control

Table 11-2072. CPSW_P0_CONTROL Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 1004h

Figure 11-935. CPSW_P0_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					DSCP_IPV6_EN	DSCP_IPV4_EN	RESERVED
R-0h					R/W-0h	R/W-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2073. CPSW_P0_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2	DSCP_IPV6_EN	R/W	0h	Port 0 IPv6 DSCP enable • 0 – Ipv6 DSCP priority mapping is disabled • 1 – Ipv6 DSCP priority mapping is enabled
1	DSCP_IPV4_EN	R/W	0h	Port 0 IPv4 DSCP enable • 0 – Ipv4 DSCP priority mapping is disabled • 1 – Ipv4 DSCP priority mapping is enabled
0	RESERVED	R	0h	Reserved

Table 11-2074. Register Call Summary for CPSW_P0_CONTROL

Gigabit Ethernet MAC (EMAC) Subsystem • Packet Priority Handling: [0][1]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P0_CONTROL Register (Offset = 1004h) [reset = 0h]: [0]

11.13.5.1.2.8 CPSW_P0_FLOW_ID_OFFSET Register (Offset = 1008h) [reset = 0h]

CPSW_P0_FLOW_ID_OFFSET is shown in Figure 11-936 and described in Table 11-2076.

CPPI Port 0 Flow ID Offset

Table 11-2075. CPSW_P0_FLOW_ID_OFFSET Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 1008h

Figure 11-936. CPSW_P0_FLOW_ID_OFFSET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								FLOW_ID_OFFSET							
R-0h																								R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2076. CPSW_P0_FLOW_ID_OFFSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	FLOW_ID_OFFSET	R/W	0h	This value is added to the thread/Flow_ID in CPPI transmit PSI Info Word 0

Table 11-2077. Register Call Summary for CPSW_P0_FLOW_ID_OFFSET

Gigabit Ethernet MAC (EMAC) Subsystem • Address Lookup Engine (ALE): [0]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P0_FLOW_ID_OFFSET Register (Offset = 1008h) [reset = 0h]: [0]
NSS Overview • CDMA Receive Flow Mapping: [0]

11.13.5.1.2.9 CPSW_P0_BLK_CNT Register (Offset = 1010h) [reset = 11h]

CPSW_P0_BLK_CNT is shown in Figure 11-937 and described in Table 11-2079.

CPPI Port 0 FIFO Block Usage Count

Table 11-2078. CPSW_P0_BLK_CNT Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 1010h

Figure 11-937. p0_blk_cnt Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			TX_BLK_CNT					RESERVED		RX_BLK_CNT					
R-0h			R-1h					R-0		R-1h					

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2079. CPSW_P0_BLK_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-8	TX_BLK_CNT	R	1h	Port 0 Transmit Block Count Usage – This value is the number of blocks allocated to the FIFO logical transmit queues.
7-6	RESERVED	R	0h	Reserved
5-0	RX_BLK_CNT	R	1h	Port 0 Receive Block Count Usage – This value is the number of blocks allocated in the receive FIFO.

Table 11-2080. Register Call Summary for CPSW_P0_BLK_CNT

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P0_BLK_CNT Register (Offset = 1010h) [reset = 11h]: [0]

11.13.5.1.2.10 CPSW_P0_PORT_VLAN Register (Offset = 1014h) [reset = 0h]

CPSW_P0_PORT_VLAN is shown in Figure 11-938 and described in Table 11-2082.

CPPI Port 0 VLAN

Table 11-2081. CPSW_P0_PORT_VLAN Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 1014h

Figure 11-938. p0_port_vlan Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
PORT_PRI			PORT_CFI	PORT_VID			
R/W-0h			R/W-0h	R/W-0h			
7	6	5	4	3	2	1	0
PORT_VID							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2082. CPSW_P0_PORT_VLAN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-13	PORT_PRI	R/W	0h	Port VLAN Priority
12	PORT_CFI	R/W	0h	Port CFI bit
11-0	PORT_VID	R/W	0h	Port VLAN ID

Table 11-2083. Register Call Summary for CPSW_P0_PORT_VLAN

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P0_PORT_VLAN Register (Offset = 1014h) [reset = 0h]: [0]

11.13.5.1.2.11 CPSW_P0_PRI_CTL Register (Offset = 101Ch) [reset = 0h]

CPSW_P0_PRI_CTL is shown in Figure 11-939 and described in Table 11-2085.

CPPI Port 0 Priority Control

Table 11-2084. CPSW_P0_PRI_CTL Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 101Ch

Figure 11-939. CPSW_P0_PRI_CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							RX_PTYPE
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RX_RLIM							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2085. CPSW_P0_PRI_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	RESERVED	R/W	0h	Reserved
15-9	RESERVED	R	0h	Reserved
8	RX_PTYPE	R/W	0h	Receive Priority Type • 0 – Fixed priority • 1 - Reserved
7-0	RX_RLIM	R/W	0h	Receive Rate Limit Enable • 00000000 – no rate-limited channels • 10000000 – channel 7 is rate-limited • 11000000 – channels 7 down to 6 are rate-limited • 11100000 – channels 7 down to 5 are rate-limited • 11110000 – channels 7 down to 4 are rate-limited • 11111000 – channels 7 down to 3 are rate-limited • 11111100 – channels 7 down to 2 are rate-limited • 11111110 – channels 7 down to 1 are rate-limited • 11111111 – channels 7 down to 0 are rate-limited • others - invalid – this bus must be set msb towards lsb. This is for Port 0 receive rate limiting into the switch (ingress). Port 0 transmit (egress) is not rate limited

Table 11-2086. Register Call Summary for CPSW_P0_PRI_CTL

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P0_PRI_CTL Register (Offset = 101Ch) [reset = 0h]: [0]

11.13.5.1.2.12 CPSW_P0_RX_PRI_MAP Register (Offset = 1020h) [reset = 76543210h]

p0_rx_pri_map is shown in Figure 11-940 and described in Table 11-2088.

CPPI Port 0 RX Pkt Pri to Header Pri Map

Table 11-2087. CPSW_P0_RX_PRI_MAP Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 1020h

Figure 11-940. p0_rx_pri_map Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED		PRI7		RESERVED		PRI6		RESERVED		PRI5		RESERVED		PRI4	
R-0h		R/W-7h		R-0h		R/W-6h		R-0h		R/W-5h		R-0h		R/W-4h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		PRI3		RESERVED		PRI2		RESERVED		PRI1		RESERVED		PRI0	
R-0h		R/W-3h		R-0h		R/W-2h		R-0h		R/W-1h		R-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2088. CPSW_P0_RX_PRI_MAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30-28	PRI7	R/W	7h	Priority 7 – A packet pri of 0x7 is mapped (changed) to this header packet pri.
27	RESERVED	R	0h	Reserved
26-24	PRI6	R/W	6h	Priority 6 – A packet pri of 0x6 is mapped (changed) to this header packet pri.
23	RESERVED	R	0h	Reserved
22-20	PRI5	R/W	5h	Priority 5 – A packet pri of 0x5 is mapped (changed) to this header packet pri.
19	RESERVED	R	0h	Reserved
18-16	PRI4	R/W	4h	Priority 4 – A packet pri of 0x4 is mapped (changed) to this header packet pri.
15	RESERVED	R	0h	Reserved
14-12	PRI3	R/W	3h	Priority 3 – A packet pri of 0x3 is mapped (changed) to this header packet pri.
11	RESERVED	R	0h	Reserved
10-8	PRI2	R/W	2h	Priority 2 – A packet pri of 0x2 is mapped (changed) to this header packet pri.
7	RESERVED	R	0h	Reserved
6-4	PRI1	R/W	1h	Priority 1 – A packet pri of 0x1 is mapped (changed) to this header packet pri.
3	RESERVED	R	0h	Reserved
2-0	PRI0	R/W	0h	Priority 0 – A packet pri of 0x0 is mapped (changed) to this header packet pri.

Table 11-2089. Register Call Summary for CPSW_P0_RX_PRI_MAP

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]

11.13.5.1.2.13 CPSW_P0_RX_MAXLEN Register (Offset = 1024h) [reset = 5EEh]

CPSW_P0_RX_MAXLEN is shown in Figure 11-941 and described in Table 11-2091.

CPPI Port 0 Receive Frame Max Length

Table 11-2090. CPSW_P0_RX_MAXLEN Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 1024h

Figure 11-941. CPSW_P0_RX_MAXLEN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																		RX_MAXLEN													
R-0h																		R/W-5EEh													

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2091. CPSW_P0_RX_MAXLEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reads return 0 and writes have no effect.
13-0	RX_MAXLEN	R/W	5EEh	Rx Maximum Frame Length – This field determines the maximum length of a received frame. The reset value is 1518 (dec). Frames with byte counts greater than RX_MAXLEN are long frames. Long frames with no errors are oversized frames. Long frames with CRC, code, or alignment error are jabber frames. The maximum value is 2020 (including VLAN).

Table 11-2092. Register Call Summary for CPSW_P0_RX_MAXLEN

Gigabit Ethernet MAC (EMAC) Subsystem • Ethernet MAC Sliver (CPGMAC_SL): [0]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P0_RX_MAXLEN Register (Offset = 1024h) [reset = 5EEh]: [0]

11.13.5.1.2.14 CPSW_P0_IDLE2LPI Register (Offset = 1030h) [reset = 0h]

CPSW_P0_IDLE2LPI is shown in Figure 11-942 and described in Table 11-2094.

Port 0 EEE Idle to LPI counter

Table 11-2093. CPSW_P0_IDLE2LPI Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 1030h

Figure 11-942. CPSW_P0_IDLE2LPI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								IDLE2LPI																							
R-0h								R/W-0h																							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2094. CPSW_P0_IDLE2LPI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return 0 and writes have no effect.
23-0	IDLE2LPI	R/W	0h	Port 0 EEE Idle to LPI counter load value - After EEE_CLKSTOP_REQ is asserted, this value is loaded into the port 0 idle to LPI counter on each clock that the port 0 transmit or receive is not idle. Port 0 enters the LPI state when this count decrements to zero (on the prescale count). This count value should be large relative to switch operations.

Table 11-2095. Register Call Summary for CPSW_P0_IDLE2LPI

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P0_IDLE2LPI Register (Offset = 1030h) [reset = 0h]: [0]

11.13.5.1.2.15 CPSW_P0_LPI2WAKE Register (Offset = 1034h) [reset = 0h]

p0_lpi2wake is shown in Figure 11-943 and described in Table 11-2097.

Port 0 EEE LPI to wake counter

Table 11-2096. CPSW_P0_LPI2WAKE Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 1034h

Figure 11-943. p0_lpi2wake Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								LPI2WAKE																							
R-0h								R/W-0h																							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2097. CPSW_P0_LPI2WAKE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return 0 and writes have no effect.
23-0	LPI2WAKE	R/W	0h	Port 0 EEE LPI to wake counter load value – When the port is in the transmit LPI state and the EEE_CLKSTOP_REQ signal is deasserted, this value is loaded into the port 0 LPI to wake counter. Transmit and receive packet operations may begin (resume) when the LPI to wake count decrements to zero (on the prescale count). This is the time waited before CPPI packet operations begin (resume after wakeup). This count value should be large relative to switch operations.

Table 11-2098. Register Call Summary for CPSW_P0_LPI2WAKE

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]

11.13.5.1.2.16 CPSW_P0_EEE_STATUS Register (Offset = 1038h) [reset = 0h]

CPSW_P0_EEE_STATUS is shown in Figure 11-944 and described in Table 11-2100.

Port 0 EEE status

Table 11-2099. CPSW_P0_EEE_STATUS Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 1038h

Figure 11-944. CPSW_P0_EEE_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	TX_FIFO_EMPTY	RX_FIFO_EMPTY	TX_FIFO_HOLD	TX_WAKE	TX_LPI	RX_LPI	WAIT_IDLE2LPI
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2100. CPSW_P0_EEE_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reads return 0 and writes have no effect.
6	TX_FIFO_EMPTY	R	0h	CPPI port 0 transmit FIFO (switch egress) is empty - contains no packets
5	RX_FIFO_EMPTY	R	0h	CPPI port 0 receive FIFO (switch ingress) is empty - contains no packets
4	TX_FIFO_HOLD	R	0h	CPPI port 0 transmit FIFO hold - asserted in the LPI state and during the LPI2WAKE count time
3	TX_WAKE	R	0h	CPPI port 0 transmit wakeup - asserted in the transmit LPI2WAKE count time
2	TX_LPI	R	0h	CPPI port 0 transmit LPI state - asserted when the port 0 transmit is in the LPI state
1	RX_LPI	R	0h	CPPI port 0 receive LPI state - asserted when the port 0 receive is in the LPI state
0	WAIT_IDLE2LPI	R	0h	CPPI port 0 wait idle to LPI - asserted when port 0 is counting the IDLE2LPI time

Table 11-2101. Register Call Summary for CPSW_P0_EEE_STATUS

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P0_EEE_STATUS Register (Offset = 1038h) [reset = 0h]: [0]

11.13.5.1.2.17 CPSW_P0_RX_DSCP_MAP_0 to CPSW_P0_RX_DSCP_MAP_7 Register (Offset = 1120h to 113Ch) [reset = 0h]

CPSW_P0_RX_DSCP_MAP_0 to CPSW_P0_RX_DSCP_MAP_7 is shown in Figure 11-945 and described in Table 11-2103.

CPPI Port 0 Receive IPV4/IPV6 DSCP Map N

Table 11-2102. CPSW_P0_RX_DSCP_MAP_0 to CPSW_P0_RX_DSCP_MAP_7 Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 1120h to 0422 113Ch

Figure 11-945. CPSW_P0_RX_DSCP_MAP_0 to CPSW_P0_RX_DSCP_MAP_7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED		PRI7		RESERVED		PRI6		RESERVED		PRI5		RESERVED		PRI4	
R-0h		R/W-0h		R-0h		R/W-0h		R-0h		R/W-0h		R-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		PRI3		RESERVED		PRI2		RESERVED		PRI1		RESERVED		PRI0	
R-0h		R/W-0h		R-0h		R/W-0h		R-0h		R/W-0h		R-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2103. CPSW_P0_RX_DSCP_MAP_0 to CPSW_P0_RX_DSCP_MAP_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30-28	PRI7	R/W	0h	A DSCP IPV4/V6 packet TOS of N*8+7 is mapped to this received priority, where N = 0 to 7
27	RESERVED	R	0h	Reserved
26-24	PRI6	R/W	0h	A DSCP IPV4/V6 packet TOS of N*8+6 is mapped to this received priority, where N = 0 to 7
23	RESERVED	R	0h	Reserved
22-20	PRI5	R/W	0h	A DSCP IPV4/V6 packet TOS of N*8+5 is mapped to this received priority, where N = 0 to 7
19	RESERVED	R	0h	Reserved
18-16	PRI4	R/W	0h	A DSCP IPV4/V6 packet TOS of N*8+4 is mapped to this received priority, where N = 0 to 7
15	RESERVED	R	0h	Reserved
14-12	PRI3	R/W	0h	A DSCP IPV4/V6 packet TOS of N*8+3 is mapped to this received priority, where N = 0 to 7
11	RESERVED	R	0h	Reserved
10-8	PRI2	R/W	0h	A DSCP IPV4/V6 packet TOS of N*8+2 is mapped to this received priority, where N = 0 to 7
7	RESERVED	R	0h	Reserved
6-4	PRI1	R/W	0h	A DSCP IPV4/V6 packet TOS of N*8+1 is mapped to this received priority, where N = 0 to 7
3	RESERVED	R	0h	Reserved
2-0	PRI0	R/W	0h	A DSCP IPV4/V6 packet TOS of N*8+0 is mapped to this received priority, where N = 0 to 7

Table 11-2104. Register Call Summary for CPSW_P0_RX_DSCP_MAP_0

Gigabit Ethernet MAC (EMAC) Subsystem • Address Lookup Engine (ALE): [0]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P0_RX_DSCP_MAP_0 to CPSW_P0_RX_DSCP_MAP_7 Register (Offset = 1120h to 113Ch) [reset = 0h]: [0]

11.13.5.1.2.18 CPSW_P0_PRI_SEND_0 to CPSW_P0_PRI_SEND_7 Register (Offset = 1140h to 115Ch) [reset = 0h]

CPSW_P0_PRI_SEND_0 to CPSW_P0_PRI_SEND_7 is shown in Figure 11-946 and described in Table 11-2106.

CPPI Port 0 Rx Priority P Send Count Value

Table 11-2105. CPSW_P0_PRI_SEND_0 to CPSW_P0_PRI_SEND_7 Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 1140h to 0422 115Ch

Figure 11-946. CPSW_P0_PRI_SEND_0 to CPSW_P0_PRI_SEND_7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														COUNT																	
R-0h														R/W-0h																	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2106. CPSW_P0_PRI_SEND_0 to CPSW_P0_PRI_SEND_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reads return 0 and writes have no effect.
17-0	COUNT	R/W	0h	Priority N send count, where N = 0 to 7

Table 11-2107. Register Call Summary for CPSW_P0_PRI_SEND_0

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P0_PRI_SEND_0 to CPSW_P0_PRI_SEND_7 Register (Offset = 1140h to 115Ch) [reset = 0h]: [0]

11.13.5.1.2.19 CPSW_P0_PRI_IDLE_0 to CPSW_P0_PRI_IDLE_7 Register (Offset = 1160h to 117Ch) [reset = 0h]

CPSW_P0_PRI_IDLE_0 to CPSW_P0_PRI_IDLE_7 is shown in Figure 11-947 and described in Table 11-2109.

CPPI Port 0 Rx Priority P Idle Count Value

Table 11-2108. CPSW_P0_PRI_IDLE_0 to CPSW_P0_PRI_IDLE_7 Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 1160h to 0422 117Ch

Figure 11-947. CPSW_P0_PRI_IDLE_0 to CPSW_P0_PRI_IDLE_7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														COUNT																	
R-0h														R/W-0h																	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2109. CPSW_P0_PRI_IDLE_0 to CPSW_P0_PRI_IDLE_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reads return 0 and writes have no effect.
17-0	COUNT	R/W	0h	Priority N idle count, where N = 0 to 7

Table 11-2110. Register Call Summary for CPSW_P0_PRI_IDLE_0

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P0_PRI_IDLE_0 to CPSW_P0_PRI_IDLE_7 Register (Offset = 1160h to 117Ch) [reset = 0h]: [0]

11.13.5.1.2.20 CPSW_P0_SRC_ID_A Register (Offset = 1300h) [reset = 1h]

CPSW_P0_SRC_ID_A is shown in Figure 11-948 and described in Table 11-2112.

CPPI Port 0 CPPI Source ID A

Table 11-2111. CPSW_P0_SRC_ID_A Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 1300h

Figure 11-948. CPSW_P0_SRC_ID_A Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PORT1							
R-0h																								R/W-1h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2112. CPSW_P0_SRC_ID_A Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reads return 0 and writes have no effect.
7-0	PORT1	R/W	1h	Port 1 CPPI Info Word0 Source ID Value. This value is contained in the CPPI Info Word 0 src_id field for packets received on port 1.

Table 11-2113. Register Call Summary for CPSW_P0_SRC_ID_A

Gigabit Ethernet MAC (EMAC) Subsystem • Port 0 CPPI Transmit Packet Streaming Interface (CPSW_2U Egress): [0]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P0_SRC_ID_A Register (Offset = 1300h) [reset = 1h]: [0]

11.13.5.1.2.21 CPSW_P1_RESERVED Register (Offset = 2000h) [reset = 0h]

CPSW_P1_RESERVED is shown in Figure 11-949 and described in Table 11-2115.

Reserved

Table 11-2114. CPSW_P1_RESERVED Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2000h

Figure 11-949. CPSW_P1_RESERVED Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2115. CPSW_P1_RESERVED Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	0h	Reserved register for memory map alignment

Table 11-2116. Register Call Summary for CPSW_P1_RESERVED

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P1_RESERVED Register (Offset = 2000h) [reset = 0h]: [0]

11.13.5.1.2.22 CPSW_P1_CONTROL Register (Offset = 2004h) [reset = 0h]

CPSW_P1_CONTROL is shown in Figure 11-950 and described in Table 11-2118.

Enet Port 1 Control

Table 11-2117. CPSW_P1_CONTROL Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2004h

Figure 11-950. CPSW_P1_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			TX_LPI_CLKSTOP_EN	RESERVED			
R-0h			R/W-0h	R-0h			
7	6	5	4	3	2	1	0
RESERVED					DSCP_IPV6_EN	DSCP_IPV4_EN	RESERVED
R-0h					R/W-0h	R/W-0h	R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2118. CPSW_P1_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12	TX_LPI_CLKSTOP_EN	R/W	0h	Transmit LPI clockstop enable • 0 – The GMII or RGMII transmit clock is not stopped in the EEE LPI state. • 1 – The GMII or RGMII transmit clock is stopped in the EEE LPI state
11-3	RESERVED	R	0h	Reserved
2	DSCP_IPV6_EN	R/W	0h	IPv6 DSCP enable • 0 – Ipv6 DSCP priority mapping is disabled • 1 – Ipv6 DSCP priority mapping is enabled
1	DSCP_IPV4_EN	R/W	0h	IPv4 DSCP enable • 0 – Ipv4 DSCP priority mapping is disabled • 1 – Ipv4 DSCP priority mapping is enabled
0	RESERVED	R	0h	Reserved

Table 11-2119. Register Call Summary for CPSW_P1_CONTROL

Gigabit Ethernet MAC (EMAC) Subsystem • Packet Priority Handling: [0][1]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P1_CONTROL Register (Offset = 2004h) [reset = 0h]: [0]

11.13.5.1.2.23 CPSW_P1_MAX_BLKS Register (Offset = 2008h) [reset = 408h]

CPSW_P1_MAX_BLKS is shown in Figure 11-951 and described in Table 11-2121.

Enet Port 1 FIFO Max Blocks

Table 11-2120. CPSW_P1_MAX_BLKS Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2008h

Figure 11-951. CPSW_P1_MAX_BLKS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				TX_MAX_BLKS				RESERVED				RX_MAX_BLKS			
R-0h				R-4h				R-0				R-8h			

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2121. CPSW_P1_MAX_BLKS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11-8	TX_MAX_BLKS	R	4h	Transmit FIFO maximum blocks - This value is the maximum number of 1k memory blocks that may be allocated to the FIFO's logical transmit priority queues. The recommended value of TX_MAX_BLKS is 0x4.
7-4	RESERVED	R	0h	Reserved
3-0	RX_MAX_BLKS	R	8h	Receive FIFO maximum blocks – This value is the maximum number of 1k memory blocks that may be allocated to the FIFO's logical receive queue. This value must be greater than or equal to 0x3. The recommended value of RX_MAX_BLKS is 0x8.

Table 11-2122. Register Call Summary for CPSW_P1_MAX_BLKS

EMAC Registers

- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P1_MAX_BLKS Register (Offset = 2008h) [reset = 408h]: [0]

11.13.5.1.2.24 CPSW_P1_BLK_CNT Register (Offset = 2010h) [reset = 101h]

CPSW_P1_BLK_CNT is shown in Figure 11-952 and described in Table 11-2124.

Enet Port 1 FIFO Block Usage Count

Table 11-2123. CPSW_PN_BLK_CNT Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2010h

Figure 11-952. CPSW_P1_BLK_CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				TX_BLK_CNT				RESERVED				RX_BLK_CNT			
R-0h				R-1h				R-0h				R-1h			

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2124. CPSW_P1_BLK_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11-8	TX_BLK_CNT	R	1h	Transmit Block Count Usage – This value is the number of blocks allocated to the port's FIFO logical transmit queues.
7-4	RESERVED	R	0h	Reserved
3-0	RX_BLK_CNT	R	1h	Receive Block Count Usage – This value is the number of blocks allocated to the port's FIFO receive queue.

Table 11-2125. Register Call Summary for CPSW_P1_BLK_CNT

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P1_BLK_CNT Register (Offset = 2010h) [reset = 101h]: [0]

11.13.5.1.2.25 CPSW_P1_PORT_VLAN Register (Offset = 2014h) [reset = 0h]

CPSW_P1_PORT_VLAN is shown in Figure 11-953 and described in Table 11-2127.

Enet Port 1 VLAN

Table 11-2126. CPSW_P1_PORT_VLAN Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2014h

Figure 11-953. CPSW_P1_PORT_VLAN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
PORT_PRI			PORT_CFI	PORT_VID			
R/W-0h			R/W-0h	R/W-0h			
7	6	5	4	3	2	1	0
PORT_VID							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2127. CPSW_P1_PORT_VLAN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-13	PORT_PRI	R/W	0h	Port VLAN Priority (7 is highest priority)
12	PORT_CFI	R/W	0h	Port CFI bit
11-0	PORT_VID	R/W	0h	Port VLAN ID

Table 11-2128. Register Call Summary for CPSW_P1_PORT_VLAN

EMAC Registers

- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P1_PORT_VLAN Register (Offset = 2014h) [reset = 0h]: [0]

11.13.5.1.2.26 CPSW_P1_PRI_CTL Register (Offset = 201Ch) [reset = 90h]

CPSW_P1_PRI_CTL is shown in Figure 11-954 and described in Table 11-2130.

Enet Port 1 Priority Control

Table 11-2129. CPSW_P1_PRI_CTL Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 201Ch

Figure 11-954. CPSW_P1_PRI_CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
TX_HOST_BLKS_REM				RESERVED			
R/W-9h				R-0h			
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2130. CPSW_P1_PRI_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-12	TX_HOST_BLKS_REM	R/W	9h	Transmit FIFO Blocks that must be free before a non rate-limited CPPI Port 0 receive thread can begin sending a packet
11-0	RESERVED	R	0h	Reserved

Table 11-2131. Register Call Summary for CPSW_P1_PRI_CTL

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P1_PRI_CTL Register (Offset = 201Ch) [reset = 90h]: [0]

11.13.5.1.2.27 CPSW_P1_RX_PRI_MAP Register (Offset = 2020h) [reset = 76543210h]

CPSW_P1_RX_PRI_MAP is shown in Figure 11-955 and described in Table 11-2133.

Enet Port 1 RX Pkt Pri to Header Pri Map

Table 11-2132. CPSW_P1_RX_PRI_MAP Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2020h

Figure 11-955. CPSW_P1_RX_PRI_MAP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED		PRI7		RESERVED		PRI6		RESERVED		PRI5		RESERVED		PRI4	
R-0h		R/W-7h		R-0h		R/W-6h		R-0h		R/W-5h		R-0h		R/W-4h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		PRI3		RESERVED		PRI2		RESERVED		PRI1		RESERVED		PRI0	
R-0h		R/W-3h		R-0h		R/W-2h		R-0h		R/W-1h		R-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2133. CPSW_P1_RX_PRI_MAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30-28	PRI7	R/W	7h	Priority 7 – A packet pri of 0x7 is mapped (changed) to this header packet pri.
27	RESERVED	R	0h	Reserved
26-24	PRI6	R/W	6h	Priority 6 – A packet pri of 0x6 is mapped (changed) to this header packet pri.
23	RESERVED	R	0h	Reserved
22-20	PRI5	R/W	5h	Priority 5 – A packet pri of 0x5 is mapped (changed) to this header packet pri.
19	RESERVED	R	0h	Reserved
18-16	PRI4	R/W	4h	Priority 4 – A packet pri of 0x4 is mapped (changed) to this header packet pri.
15	RESERVED	R	0h	Reserved
14-12	PRI3	R/W	3h	Priority 3 – A packet pri of 0x3 is mapped (changed) to this header packet pri.
11	RESERVED	R	0h	Reserved
10-8	PRI2	R/W	2h	Priority 2 – A packet pri of 0x2 is mapped (changed) to this header packet pri.
7	RESERVED	R	0h	Reserved
6-4	PRI1	R/W	1h	Priority 1 – A packet pri of 0x1 is mapped (changed) to this header packet pri.
3	RESERVED	R	0h	Reserved
2-0	PRI0	R/W	0h	Priority 0 – A packet pri of 0x0 is mapped (changed) to this header packet pri.

Table 11-2134. Register Call Summary for CPSW_P1_RX_PRI_MAP

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P1_RX_PRI_MAP Register (Offset = 2020h) [reset = 76543210h]: [0]

11.13.5.1.2.28 CPSW_P1_RX_MAXLEN Register (Offset = 2024h) [reset = 5EEh]

CPSW_P1_RX_MAXLEN is shown in Figure 11-956 and described in Table 11-2136.

Enet Port 1 Receive Frame Max Length

Table 11-2135. CPSW_P1_RX_MAXLEN Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2024h

Figure 11-956. CPSW_P1_RX_MAXLEN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																		RX_MAXLEN													
R-0h																		R/W-5EEh													

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2136. CPSW_PN_RX_MAXLEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reads return 0 and writes have no effect.
13-0	RX_MAXLEN	R/W	5EEh	Rx Maximum Frame Length – This field determines the maximum length of a received frame. The reset value is 1518 (dec). Frames with byte counts greater than RX_MAXLEN are long frames. Long frames with no errors are oversized frames. Long frames with CRC, code, or alignment error are jabber frames. The maximum value is 9604 (including VLAN).

Table 11-2137. Register Call Summary for CPSW_P1_RX_MAXLEN

Gigabit Ethernet MAC (EMAC) Subsystem • Ethernet MAC Sliver (CPGMAC_SL): [0]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P1_RX_MAXLEN Register (Offset = 2024h) [reset = 5EEh]: [0]

11.13.5.1.2.29 CPSW_P1_IDLE2LPI Register (Offset = 2030h) [reset = 0h]

CPSW_P1_IDLE2LPI is shown in Figure 11-957 and described in Table 11-2139.

Enet Port 1 EEE Idle to LPI counter

Table 11-2138. CPSW_P1_IDLE2LPI Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2030h

Figure 11-957. CPSW_P1_IDLE2LPI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								IDLE2LPI																							
R-0h								R/W-0h																							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2139. CPSW_P1_IDLE2LPI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Read as zero.
23-0	IDLE2LPI	R/W	0h	EEE Idle to LPI counter load value- After EEE_CLKSTOP_REQ is asserted, this value is loaded into the port idle to LPI counter on each clock that the port transmit is not idle. The port enters the transmit LPI state when this count decrements to zero. This count decrements each time the EEE prescale count decrements to zero.

Table 11-2140. Register Call Summary for CPSW_P1_IDLE2LPI

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P1_IDLE2LPI Register (Offset = 2030h) [reset = 0h]: [0]

11.13.5.1.2.30 CPSW_P1_LPI2WAKE Register (Offset = 2034h) [reset = 0h]

CPSW_P1_LPI2WAKE is shown in Figure 11-958 and described in Table 11-2142.

Enet Port 1 EEE LPI to wake counter

Table 11-2141. CPSW_P1_LPI2WAKE Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2034h

Figure 11-958. CPSW_P1_LPI2WAKE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								LPI2WAKE																							
R-0h								R/W-0h																							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2142. CPSW_P1_LPI2WAKE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Read as zero.
23-0	LPI2WAKE	R/W	0h	EEE LPI to wake counter load value – When the port is in the transmit LPI state and the EEE_CLKSTOP_REQ signal is deasserted, this value is loaded into the LPI to wake counter. Transmit packet operations may begin (resume) when the LPI to wake count decrements to zero (on the pre-scale count). This is the time waited before Ethernet transmit operations resume after wakeup.

Table 11-2143. Register Call Summary for CPSW_P1_LPI2WAKE

EMAC Registers

- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P1_LPI2WAKE Register (Offset = 2034h) [reset = 0h]: [0]

11.13.5.1.2.31 CPSW_P1_EEE_STATUS Register (Offset = 2038h) [reset = 0h]

CPSW_P1_EEE_STATUS is shown in Figure 11-959 and described in Table 11-2145.

Enet Port 1 EEE status

Table 11-2144. CPSW_P1_EEE_STATUS Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2038h

Figure 11-959. CPSW_P1_EEE_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	TX_FIFO_EMPTY	RX_FIFO_EMPTY	TX_FIFO_HOLD	TX_WAKE	TX_LPI	RX_LPI	WAIT_IDLE2LPI
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2145. CPSW_P1_EEE_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	TX_FIFO_EMPTY	R	0h	Transmit FIFO (switch egress) is empty - contains no packets
5	RX_FIFO_EMPTY	R	0h	Receive FIFO (switch ingress) is empty - contains no packets
4	TX_FIFO_HOLD	R	0h	Transmit FIFO hold - asserted in the LPI state and during the LPI2WAKE count time
3	TX_WAKE	R	0h	Transmit wakeup - asserted in the transmit LPI2WAKE count time
2	TX_LPI	R	0h	Transmit LPI state - asserted when the port 0 transmit is in the LPI state
1	RX_LPI	R	0h	Receive LPI state - asserted when the port 0 receive is in the LPI state
0	WAIT_IDLE2LPI	R	0h	CPPI port 0 wait idle to LPI - asserted when port 0 is counting the IDLE2LPI time

Table 11-2146. Register Call Summary for CPSW_P1_EEE_STATUS

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P1_EEE_STATUS Register (Offset = 2038h) [reset = 0h]: [0]

11.13.5.1.2.32 CPSW_P1_RX_DSCP_MAP_0 to CPSW_P1_RX_DSCP_MAP_7 Register (Offset =2120h - 214Ch) [reset = 0h]

CPSW_P1_RX_DSCP_MAP_0 to CPSW_P1_RX_DSCP_MAP_7 is shown in Figure 11-960 and described in Table 11-2148.

Enet Port 1 Receive IPV4/IPV6 DSCP Map M

Table 11-2147. CPSW_P1_RX_DSCP_MAP_0 to CPSW_P1_RX_DSCP_MAP_7 Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2120h - 0422 214Ch

Figure 11-960. CPSW_P1_RX_DSCP_MAP_0 to CPSW_P1_RX_DSCP_MAP_7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED		PRI7		RESERVED		PRI6		RESERVED		PRI5		RESERVED		PRI4	
R-0h		R/W-7h		R-0h		R/W-6h		R-0h		R/W-5h		R-0h		R/W-4h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		PRI3		RESERVED		PRI2		RESERVED		PRI1		RESERVED		PRI0	
R-0h		R/W-3h		R-0h		R/W-2h		R-0h		R/W-1h		R-0h		R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2148. CPSW_P1_RX_DSCP_MAP_0 to CPSW_P1_RX_DSCP_MAP_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30-28	PRI7	R/W	7h	A DSCP IPV4/V6 packet TOS of N*8+7 is mapped to this received priority, where N = 0 to 7.
27	RESERVED	R	0h	Reserved
26-24	PRI6	R/W	6h	A DSCP IPV4/V6 packet TOS of N*8+6 is mapped to this received priority, where N = 0 to 7.
23	RESERVED	R	0h	Reserved
22-20	PRI5	R/W	5h	A DSCP IPV4/V6 packet TOS of N*8+5 is mapped to this received priority, where N = 0 to 7.
19	RESERVED	R	0h	Reserved
18-16	PRI4	R/W	4h	A DSCP IPV4/V6 packet TOS of N*8+4 is mapped to this received priority, where N = 0 to 7.
15	RESERVED	R	0h	Reserved
14-12	PRI3	R/W	3h	A DSCP IPV4/V6 packet TOS of N*8+3 is mapped to this received priority, where N = 0 to 7.
11	RESERVED	R	0h	Reserved
10-8	PRI2	R/W	2h	A DSCP IPV4/V6 packet TOS of N*8+2 is mapped to this received priority, where N = 0 to 7.
7	RESERVED	R	0h	Reserved
6-4	PRI1	R/W	1h	A DSCP IPV4/V6 packet TOS of N*8+1 is mapped to this received priority, where N = 0 to 7.
3	RESERVED	R	0h	Reserved
2-0	PRI0	R/W	0h	A DSCP IPV4/V6 packet TOS of N*8+0 is mapped to this received priority, where N = 0 to 7.

Table 11-2149. Register Call Summary for CPSW_P1_RX_DSCP_MAP_0

Gigabit Ethernet MAC (EMAC) Subsystem • Address Lookup Engine (ALE): [0]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P1_RX_DSCP_MAP_0 to CPSW_P1_RX_DSCP_MAP_7 Register (Offset =2120h - 214Ch) [reset = 0h]: [0]

11.13.5.1.2.33 CPSW_P1_PRI_SEND_0 to CPSW_P1_PRI_SEND_7 Register (Offset = 2140h - 215Ch) [reset = 0h]

CPSW_P1_PRI_SEND_0 to CPSW_P1_PRI_SEND_7 is shown in Figure 11-961 and described in Table 11-2151.

Enet Port 1 Rx Priority P Send Count Value

Table 11-2150. CPSW_P1_PRI_SEND_0 to CPSW_P1_PRI_SEND_7 Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2140h - 0422 215Ch

Figure 11-961. CPSW_P1_PRI_SEND_0 to CPSW_P1_PRI_SEND_7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														COUNT																	
R-0h														R/W-0h																	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2151. CPSW_P1_PRI_SEND_0 to CPSW_P1_PRI_SEND_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17-0	COUNT	R/W	0h	Priority N send count, where N = 0 to 7.

Table 11-2152. Register Call Summary for CPSW_P1_PRI_SEND_0

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P1_PRI_SEND_0 to CPSW_P1_PRI_SEND_7 Register (Offset = 2140h - 215Ch) [reset = 0h]: [0]

11.13.5.1.2.34 CPSW_P1_PRI_IDLE_0 to CPSW_P1_PRI_IDLE_7 Register (Offset = 2160h - 217Ch) [reset = 0h]

CPSW_P1_PRI_IDLE_0 to CPSW_P1_PRI_IDLE_7 is shown in Figure 11-962 and described in Table 11-2154.

Enet Port 1 Rx Priority P Idle Count Value

Table 11-2153. CPSW_P1_PRI_IDLE_0 to CPSW_P1_PRI_IDLE_7 Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2160h - 0422 217Ch

Figure 11-962. CPSW_P1_PRI_IDLE_0 to CPSW_P1_PRI_IDLE_7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														COUNT																	
R-0h														R/W-0h																	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2154. CPSW_P1_PRI_IDLE_0 to CPSW_P1_PRI_IDLE_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17-0	COUNT	R/W	0h	Priority N idle count, where N = 0 to 7

Table 11-2155. Register Call Summary for CPSW_P1_PRI_IDLE_0

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P1_PRI_IDLE_0 to CPSW_P1_PRI_IDLE_7 Register (Offset = 2160h - 217Ch) [reset = 0h]: [0]

11.13.5.1.2.35 CPSW_P1_SA_L Register (Offset = 2308h) [reset = 0h]

CPSW_P1_SA_L is shown in Figure 11-963 and described in Table 11-2157.

Enet Port 1 Tx Pause Frame Source Address Low

Table 11-2156. CPSW_P1_SA_L Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2308h

Figure 11-963. CPSW_P1_SA_L Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MACSRCADDR_7_0								MACSRCADDR_15_8							
R/W-0h								R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2157. CPSW_P1_SA_L Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	MACSRCADDR_7_0	R/W	0h	Source Address Lower 8 bits (byte 0)
7-0	MACSRCADDR_15_8	R/W	0h	Source Address bits 15:8 (byte 1)

Table 11-2158. Register Call Summary for CPSW_P1_SA_L

Gigabit Ethernet MAC (EMAC) Subsystem • Initialization and Configuration of EMAC Subsystem: [0]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P1_SA_L Register (Offset = 2308h) [reset = 0h]: [0]

11.13.5.1.2.36 CPSW_P1_SA_H Register (Offset = 230Ch) [reset = 0h]

CPSW_P1_SA_H is shown in Figure 11-964 and described in Table 11-2160.

Enet Port 1 Tx Pause Frame Source Address High

Table 11-2159. CPSW_P1_SA_H Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 230Ch

Figure 11-964. CPSW_P1_SA_H Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MACSRCADDR_23_16								MACSRCADDR_31_24							
R/W-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MACSRCADDR_39_32								MACSRCADDR_47_40							
R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2160. CPSW_P1_SA_H Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	MACSRCADDR_23_16	R/W	0h	Source Address bits 23:16 (byte 2)
23-16	MACSRCADDR_31_24	R/W	0h	Source Address bits 31:24 (byte 3)
15-8	MACSRCADDR_39_32	R/W	0h	Source Address bits 39:32 (byte 4)
7-0	MACSRCADDR_47_40	R/W	0h	Source Address bits 47:40 (byte 5)

Table 11-2161. Register Call Summary for CPSW_P1_SA_H

Gigabit Ethernet MAC (EMAC) Subsystem • Initialization and Configuration of EMAC Subsystem: [0]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P1_SA_H Register (Offset = 230Ch) [reset = 0h]: [0]

11.13.5.1.2.37 CPSW_P1_TS_CTL Register (Offset = 2310h) [reset = 0h]

CPSW_P1_TS_CTL is shown in Figure 11-965 and described in Table 11-2163.

Enet Port 1 Time Sync Control

Table 11-2162. CPSW_P1_TS_CTL Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2310h

Figure 11-965. CPSW_P1_TS_CTL Register

31	30	29	28	27	26	25	24
TS_MSG_TYPE_EN							
R/W-0h							
23	22	21	20	19	18	17	16
TS_MSG_TYPE_EN							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED				TS_TX_HOST_TS_EN	TS_TX_ANNEX_E_EN	TS_RX_ANNEX_E_EN	TS_LTYPE2_EN
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TS_TX_ANNEX_D_EN	TS_TX_VLAN_LTYPE2_EN	TS_TX_VLAN_LTYPE1_EN	TS_TX_ANNEX_F_EN	TS_RX_ANNEX_D_EN	TS_RX_VLAN_LTYPE2_EN	TS_RX_VLAN_LTYPE1_EN	TS_RX_ANNEX_F_EN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2163. CPSW_P1_TS_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	TS_MSG_TYPE_EN	R/W	0h	Time Sync Message Type Enable Each bit in this field enables the corresponding message type in receive and transmit time sync messages (Bit 0 enables message type 0, etc.).
15-12	RESERVED	R	0h	Reserved
11	TS_TX_HOST_TS_EN	R/W	0h	Time Sync Transmit Host Time Stamp Enable
10	TS_TX_ANNEX_E_EN	R/W	0h	Time Sync Transmit Annex E Enable
9	TS_RX_ANNEX_E_EN	R/W	0h	Time Sync Receive Annex E Enable
8	TS_LTYPE2_EN	R/W	0h	Time Sync LTYPE 2 enable (Transmit and Receive)
7	TS_TX_ANNEX_D_EN	R/W	0h	Time Sync Transmit Annex D Enable
6	TS_TX_VLAN_LTYPE2_EN	R/W	0h	Time Sync Transmit VLAN LTYPE 2 enable
5	TS_TX_VLAN_LTYPE1_EN	R/W	0h	Time Sync Transmit VLAN LTYPE 1 enable
4	TS_TX_ANNEX_F_EN	R/W	0h	Time Sync Transmit Annex F Enable
3	TS_RX_ANNEX_D_EN	R/W	0h	Time Sync Receive Annex D Enable
2	TS_RX_VLAN_LTYPE2_EN	R/W	0h	Time Sync Receive VLAN LTYPE 2 enable
1	TS_RX_VLAN_LTYPE1_EN	R/W	0h	Time Sync Receive VLAN LTYPE 1 enable
0	TS_RX_ANNEX_F_EN	R/W	0h	Time Sync Receive Annex F Enable

Table 11-2164. Register Call Summary for CPSW_P1_TS_CTL

Gigabit Ethernet MAC (EMAC) Subsystem • Time Sync Events: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25][26][27][28][29][30][31][32][33][34][35][36][37][38][39][40][41][42][43][44][45][46][47][48][49][50][51][52]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P1_TS_CTL Register (Offset = 2310h) [reset = 0h]: [0]

11.13.5.1.2.38 CPSW_P1_TS_SEQ_LTYPE Register (Offset = 2314h) [reset = 1E0h]

CPSW_P1_TS_SEQ_LTYPE is shown in Figure 11-966 and described in Table 11-2166.

Enet Port 1 Time Sync LTYPE (and SEQ_ID_OFFSET)

Table 11-2165. CPSW_P1_TS_SEQ_LTYPE Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2314h

Figure 11-966. CPSW_P1_TS_SEQ_LTYPE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED										TS_SEQ_ID_OFFSET					
R-0h										R/W-1Eh					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS_LTYPE1															
R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2166. CPSW_PN_TS_SEQ_LTYPE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved
21-16	TS_SEQ_ID_OFFSET	R/W	1Eh	Time Sync Sequence ID Offset. This is the number of octets that the sequence ID is offset in the tx and rx time sync message header. The minimum value is 6.
15-0	TS_LTYPE1	R/W	0h	Time Sync LTYPE1. This is the port's time sync LTYPE1 value.

Table 11-2167. Register Call Summary for CPSW_P1_TS_SEQ_LTYPE

Gigabit Ethernet MAC (EMAC) Subsystem • Time Sync Events: [0][1][2][3][4][5][6][7][8]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P1_TS_SEQ_LTYPE Register (Offset = 2314h) [reset = 1E0h]: [0]

11.13.5.1.2.39 CPSW_P1_TS_VLAN_LTYPE Register (Offset = 2318h) [reset = 81008100h]

CPSW_P1_TS_VLAN_LTYPE is shown in Figure 11-967 and described in Table 11-2169.

Enet Port 1 Time Sync VLAN2 and VLAN2

Table 11-2168. CPSW_P1_TS_VLAN_LTYPE Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2318h

Figure 11-967. CPSW_P1_TS_VLAN_LTYPE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS_VLAN_LTYPE2																TS_VLAN_LTYPE1															
R/W-8100h																R/W-8100h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2169. CPSW_P1_TS_VLAN_LTYPE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	TS_VLAN_LTYPE2	R/W	8100h	Time Sync VLAN LTYPE2. This VLAN LTYPE value is used for the port tx and rx time sync decode.
15-0	TS_VLAN_LTYPE1	R/W	8100h	Time Sync VLAN LTYPE1. This VLAN LTYPE value is used for the port tx and rx time sync decode.

Table 11-2170. Register Call Summary for CPSW_P1_TS_VLAN_LTYPE

Gigabit Ethernet MAC (EMAC) Subsystem • Time Sync Events: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25][26][27]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P1_TS_VLAN_LTYPE Register (Offset = 2318h) [reset = 81008100h]: [0]

11.13.5.1.2.40 CPSW_P1_TS_CTL_LTYPE2 Register (Offset = 231Ch) [reset = 0h]

CPSW_P1_TS_CTL_LTYPE2 is shown in Figure 11-968 and described in Table 11-2172.

Enet Port 1 Time Sync Control and LTYPE 2

Table 11-2171. CPSW_P1_TS_CTL_LTYPE2 Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 231Ch

Figure 11-968. CPSW_P1_TS_CTL_LTYPE2 Register

31	30	29	28	27	26	25	24
RESERVED							TS_UNI_EN
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TS_TTL_NONZERO	TS_320	TS_319	TS_132	TS_131	TS_130	TS_129	TS_107
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
TS_LTYPE2							
R/W-0h							
7	6	5	4	3	2	1	0
TS_LTYPE2							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2172. CPSW_P1_TS_CTL_LTYPE2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24	TS_UNI_EN	R/W	0h	Time Sync Unicast Enable • 0 – Unicast disabled • 1 – Unicast enabled
23	TS_TTL_NONZERO	R/W	0h	Time Sync Time to Live Non-zero Enable • 0 – TTL must be zero • 1 – TTL may be non-zero
22	TS_320	R/W	0h	Time Sync Destination IP Address 320 Enable • 0 – disabled • 1 – destination port number (dec) 320 is enabled.
21	TS_319	R/W	0h	Time Sync Destination IP Address 319 Enable • 0 – disabled • 1 – destination port number (dec) 319 is enabled.
20	TS_132	R/W	0h	Time Sync Destination IP Address 132 Enable • 0 – disabled • 1 – destination IP address (dec) 224.0.1.132 is enabled.
19	TS_131	R/W	0h	Time Sync Destination IP Address 131 Enable • 0 – disabled • 1 – destination IP address (dec) 224.0.1.131 is enabled
18	TS_130	R/W	0h	Time Sync Destination IP Address 130 Enable • 0 – disabled • 1 – destination IP address (dec) 224.0.1.130 is enabled.

Table 11-2172. CPSW_P1_TS_CTL_LTYPE2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	TS_129	R/W	0h	Time Sync Destination IP Address 129 Enable • 0 – disabled • 1 – destination IP address (dec) 224.0.1.129 is enabled.
16	TS_107	R/W	0h	Time Sync Destination IP Address 107 Enable • 0 – disabled • 1 – destination IP address (dec) 224.0.0.107 is enabled.
15-0	TS_LTYPE2	R/W	0h	Time Sync LTYPE2. This is the time sync LTYPE1 value for port 1.

Table 11-2173. Register Call Summary for CPSW_P1_TS_CTL_LTYPE2

Gigabit Ethernet MAC (EMAC) Subsystem • Time Sync Events: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25][26][27][28][29][30][31][32][33][34][35][36][37][38][39][40][41][42][43][44][45][46]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P1_TS_CTL_LTYPE2 Register (Offset = 231Ch) [reset = 0h]: [0]

11.13.5.1.2.41 CPSW_P1_TS_CTL2 Register (Offset = 2320h) [reset = 400h]

CPSW_P1_TS_CTL2 is shown in Figure 11-969 and described in Table 11-2175.

Enet Port 1 Time Sync Control 2

Table 11-2174. CPSW_P1_TS_CTL2 Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2320h

Figure 11-969. CPSW_P1_TS_CTL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED										TS_DOMAIN_OFFSET					
R-0h										R/W-4h					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS_MCAST_TYPE_EN															
R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2175. CPSW_P1_TS_CTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved
21-16	TS_DOMAIN_OFFSET	R/W	4h	Time Sync Domain Offset
15-0	TS_MCAST_TYPE_EN	R/W	0h	Time Sync Multicast Destination Address Type Enable

Table 11-2176. Register Call Summary for CPSW_P1_TS_CTL2

Gigabit Ethernet MAC (EMAC) Subsystem • Time Sync Events: [0][1]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P1_TS_CTL2 Register (Offset = 2320h) [reset = 400h]: [0]

11.13.5.1.2.42 CPSW_P1_MAC_CONTROL Register (Offset = 2330h) [reset = 0h]

CPSW_P1_MAC_CONTROL is shown in Figure 11-970 and described in Table 11-2178.

Enet Port 1 Mac Control

Table 11-2177. CPSW_P1_MAC_CONTROL Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2330h

Figure 11-970. CPSW_P1_MAC_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							RX_CMF_EN
R-0h							R/W-0h
23	22	21	20	19	18	17	16
RX_CSF_EN	RX_CEF_EN	RESERVED	EXT_TX_FLOW_EN	EXT_RX_FLOW_EN	CTL_EN	GIG_FORCE	IFCTL_B
R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
IFCTL_A	RESERVED		CRC_TYPE	CMD_IDLE	RESERVED		
R/W-0h	R-0h		R/W-0h	R/W-0h	R-0h		
7	6	5	4	3	2	1	0
GIG	TX_PACE	GMII_EN	TX_FLOW_EN	RX_FLOW_EN	MTEST	LOOPBACK	FULLDUPLEX
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2178. CPSW_P1_MAC_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reads return 0 and writes have no effect.
24	RX_CMF_EN	R/W	0h	RX Copy MAC Control Frames Enable – Enables MAC control frames to be transferred to memory. MAC control frames are normally acted upon (if enabled), but not copied to memory. MAC control frames that are pause frames will be acted upon if enabled in the MacControl register, regardless of the value of RX_CMF_EN. Frames transferred to memory due toRX_CMF_EN will have the control bit set in their EOP buffer descriptor. • 0 – MAC control frames are filtered (but acted upon if enabled). • 1 – MAC control frames are transferred to memory.
23	RX_CSF_EN	R/W	0h	Enables frames or fragments shorter than 64 bytes to be copied to memory. Frames transferred to memory due to rx_csf_en will have the fragment or undersized bit set in their receive footer. Fragments are short frames that contain CRC/align/code errors and undersized are short frames without errors. • 0 –Short frames are filtered • 1 –Short frames are transferred to memory.
22	RX_CEF_EN	R/W	0h	RX Copy Error Frames Enable – Enables frames containing errors to be transferred to memory. The appropriate error bit will be set in the frame receive footer. Frames containing errors will be filtered when rx_cef _en is not set. • 0 – Frames containing errors are filtered. • 1 – Frames containing errors are transferred to memory.
21	RESERVED	R	0h	Reserved
20	EXT_TX_FLOW_EN	R/W	0h	External Transmit Flow Control Enable Enables the tx_flow_en to be selected from the EXT_TX_FLOW_EN input signal and not from the TX_FLOW_EN bit in this register.

Table 11-2178. CPSW_P1_MAC_CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	EXT_RX_FLOW_EN	R/W	0h	External Receive Flow Control Enable – Enables the rx_flow_en to be selected from the EXT_RX_FLOW_EN input signal and not from the RX_FLOW_EN bit in this register.
18	CTL_EN	R/W	0h	Control Enable – Enables the fullduplex and gigabit mode to be selected from the FULLDUPLEX_IN and GIG_IN input signals and not from the fullduplex and gig bits in this register. The FULLDUPLEX_MODE bit reflects the actual fullduplex mode selected.
17	GIG_FORCE	R/W	0h	Gigabit Mode Force – This bit is used to force the Enet Mac into gigabit mode if the input GMII_MTCLK has been stopped by the PHY.
16	IFCTL_B	R/W	0h	Interface Control B
15	IFCTL_A	R/W	0h	Interface Control A.Determines the RMII speed • 0 – 10 Mbps • 1 – 100 Mbps
14-13	RESERVED	R	0h	Reserved
12	CRC_TYPE	R/W	0h	Port CRC Type • 0 –Ethernet CRC • 1 – Castagnoli CRC
11	CMD_IDLE	R/W	0h	Command Idle • 0 – Idle not commanded • 1 – Idle Commanded (read idle in Enet_Pn_Mac_Status)
10-8	RESERVED	R	0h	Reserved
7	GIG	R/W	0h	Gigabit Mode • 0 – 10/100 mode • 1 – Gigabit mode (full duplex only) The GIG_OUT output is the value of this bit.
6	TX_PACE	R/W	0h	Transmit Pacing Enable • 0 – Transmit Pacing Disabled • 1 – Transmit Pacing Enabled
5	GMII_EN	R/W	0h	GMII Enable • 0 – GMII RX and TX held in reset • 1 – GMII RX and TX released from reset.
4	TX_FLOW_EN	R/W	0h	Transmit Flow Control Enable – Determines if incoming pause frames are acted upon in full-duplex mode. Incoming pause frames are not acted upon in half-duplex mode regardless of this bit setting. The RX_MBP_Enable bits determine whether or not received pause frames are transferred to memory. • 0 – Transmit Flow Control Disabled. Full-duplex mode – Incoming pause frames are not acted upon. • 1 – Transmit Flow Control Enabled . Full-duplex mode – Incoming pause frames are acted upon.
3	RX_FLOW_EN	R/W	0h	Receive Flow Control Enable • 0 – Receive Flow Control Disabled Half-duplex mode – No flow control generated collisions are sent. Full-duplex mode – No outgoing pause frames are sent. • 1 – Receive Flow Control Enabled Half-duplex mode – Collisions are initiated when receive flow control is triggered. Full-duplex mode – Outgoing pause frames are sent when receive flow control is triggered.
2	MTEST	R/W	0h	Manufacturing Test Mode – This bit must be set to allow writes to the BACKOFF_TEST and PAUSETIMER registers.

Table 11-2178. CPSW_P1_MAC_CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	LOOPBACK	R/W	0h	Loop Back Mode – Loopback mode forces internal fullduplex mode regardless of whether the fullduplex bit is set or not. The loopback bit should be changed only when GMII_EN is de-asserted. • 0 – Not looped back • 1 – Loop Back Mode enabled
0	FULLDUPLEX	R/W	0h	Full Duplex mode – Gigabit mode forces fullduplex mode regardless of whether the fullduplex bit is set or not. The FULLDUPLEX_OUT output is the value of this register bit • 0 – half duplex mode • 1 – full duplex mode

Table 11-2179. Register Call Summary for CPSW_P1_MAC_CONTROL

Gigabit Ethernet MAC (EMAC) Subsystem • Flow Control: [0][1][2][3][4][5][6][7][8] • Address Lookup Engine (ALE): [0][1] • MDIO Functional Description: [0] • Ethernet MAC Reset: [0] • Packet CRC Handling: [0][1]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P1_MAC_CONTROL Register (Offset = 2330h) [reset = 0h]: [0]

11.13.5.1.2.43 CPSW_P1_MAC_STATUS Register (Offset = 2334h) [reset = 0h]

CPSW_P1_MAC_STATUS is shown in Figure 11-971 and described in Table 11-2181.

Enet Port 1 Mac Status

Table 11-2180. CPSW_P1_MAC_STATUS Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2334h

Figure 11-971. CPSW_P1_MAC_STATUS Register

31	30	29	28	27	26	25	24
IDLE	RESERVED						
R-0	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	EXT_RX_FLOW_EN	EXT_TX_FLOW_EN	EXT_GIG	EXT_FULLDUPLEX	RESERVED	RX_FLOW_ACT	TX_FLOW_ACT
R-0h	R-	R-	R-	R-	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2181. CPSW_P1_MAC_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IDLE	R	0h	Enet IDLE – The Ethernet port is in the idle state (valid after an idle command) • 0 – The port is not in the idle state. • 1 - The port is in the idle state.
30-7	RESERVED	R	0h	Reserved
6	EXT_RX_FLOW_EN	R	-	External Receive Flow Control Enable – This is the value of the EXT_RX_FLOW_EN input bit.
5	EXT_TX_FLOW_EN	R	-	External Receive Flow Control Enable – This is the value of the EXT_TX_FLOW_EN input bit.
4	EXT_GIG	R	-	External GIG mode – This is the value of the EXT_GIG input bit.
3	EXT_FULLDUPLEX	R	-	External Fullduplex – This is the value of the EXT_FULLDUPLEX input bit.
2	RESERVED	R	0h	Reserved
1	RX_FLOW_ACT	R	0h	Receive Flow Control Active – When asserted, indicates that receive flow control is enabled and triggered.
0	TX_FLOW_ACT	R	0h	Transmit Flow Control Active When asserted, this bit indicates that the pause time period is being observed for a received pause frame. No new transmissions will begin while this bit is asserted except for the transmission of pause frames. Any transmission in progress when this bit is asserted will complete.

Table 11-2182. Register Call Summary for CPSW_P1_MAC_STATUS

Gigabit Ethernet MAC (EMAC) Subsystem • Ethernet MAC Reset: [0]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P1_MAC_STATUS Register (Offset = 2334h) [reset = 0h]: [0]

11.13.5.1.2.44 CPSW_P1_MAC_SOFT_RESET Register (Offset = 2338h) [reset = 0h]

CPSW_P1_MAC_SOFT_RESET is shown in Figure 11-972 and described in Table 11-2184.

Enet Port 1 Mac Soft Reset

Table 11-2183. CPSW_P1_MAC_SOFT_RESET Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2338h

Figure 11-972. CPSW_P1_MAC_SOFT_RESET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							SOFT_RESET
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2184. CPSW_P1_MAC_SOFT_RESET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	SOFT_RESET	R/W	0h	Software reset – Writing a one to this bit causes the Ethernet Mac logic to be reset. After writing a one to this bit, it may be polled to determine if the reset has occurred. If a one is read, the reset has not yet occurred. If a zero is read then reset has occurred.

Table 11-2185. Register Call Summary for CPSW_P1_MAC_SOFT_RESET

Gigabit Ethernet MAC (EMAC) Subsystem • Ethernet MAC Reset: [0][1] • Initialization and Configuration of EMAC Subsystem: [0]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P1_MAC_SOFT_RESET Register (Offset = 2338h) [reset = 0h]: [0]

11.13.5.1.2.45 CPSW_P1_MAC_BOFFTEST Register (Offset = 233Ch) [reset = 0h]

CPSW_P1_MAC_BOFFTEST is shown in Figure 11-973 and described in Table 11-2187.

Enet Port 1 Mac Backoff Test

Table 11-2186. CPSW_P1_MAC_BOFFTEST Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 233Ch

Figure 11-973. CPSW_P1_MAC_BOFFTEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED	PACEVAL					RNDNUM									
R-0h	R/W-0h					R/W-0h									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COLL_COUNT				RESERVED		TX_BACKOFF									
R-0h				R-0h		R-0h									

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2187. CPSW_P1_MAC_BOFFTEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30-26	PACEVAL	R/W	0h	Pacing Register Current Value. A non-zero value in this field indicates that transmit pacing is active. A transmit frame collision or deferral causes paceval to loaded with decimal 31, good frame transmissions (with no collisions or deferrals) cause paceval to be decremented down to zero. When paceval is nonzero, the transmitter delays 4 IPGs between new frame transmissions after each successfully transmitted frame that had no deferrals or collisions. Transmit pacing helps reduce “capture” effects improving overall network bandwidth.
25-16	RNDNUM	R/W	0h	Backoff Random Number Generator – This field allows the Backoff Random Number Generator to be read (or written in test mode only). This field can be written only when mtest has previously been set. Reading this field returns the generator’s current value. The value is reset to zero and begins counting on the clock after the de-assertion of reset
15-12	COLL_COUNT	R	0h	Collision Count – The number of collisions the current frame has experienced.
11-10	RESERVED	R	0h	Reserved
9-0	TX_BACKOFF	R	0h	Backoff Count – This field allows the current value of the backoff counter to be observed for test purposes. This field is loaded automatically according to the backoff algorithm, and is decremented by one for each slot time after the collision.

Table 11-2188. Register Call Summary for CPSW_P1_MAC_BOFFTEST

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P1_MAC_BOFFTEST Register (Offset = 233Ch) [reset = 0h]: [0]

11.13.5.1.2.46 CPSW_P1_MAC_RX_PAUSETIMER Register (Offset = 2340h) [reset = 0h]

CPSW_P1_MAC_RX_PAUSETIMER is shown in Figure 11-974 and described in Table 11-2190.

Enet Port 1 802.3 Receive Pause Timer

Table 11-2189. CPSW_P1_MAC_RX_PAUSETIMER Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2340h

Figure 11-974. CPSW_P1_MAC_RX_PAUSETIMER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RX_PAUSETIMER															
R-0h																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2190. CPSW_P1_MAC_RX_PAUSETIMER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	RX_PAUSETIMER	R/W	0h	RX Pause Timer Value – This field allows the contents of the receive pause timer to be observed (and written in test mode). The receive pause timer is loaded with 0xFF00 when the Enet port sends an outgoing pause frame (with pause time of 0xFFFF). The receive pause timer is decremented at slot time intervals. If the receive pause timer decrements to zero, then another outgoing pause frame will be sent and the load/decrement process will be repeated. This register is for 802.3 Based flow control and is not used for 802.1qbb Priority Based Flow Control (PFC).

Table 11-2191. Register Call Summary for CPSW_P1_MAC_RX_PAUSETIMER

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P1_MAC_RX_PAUSETIMER Register (Offset = 2340h) [reset = 0h]: [0]

11.13.5.1.2.47 CPSW_P1_MAC_TX_PAUSETIMER Register (Offset = 2370h) [reset = 0h]

CPSW_P1_MAC_TX_PAUSETIMER is shown in Figure 11-975 and described in Table 11-2193.

Enet Port 1 802.3 Tx Pause Timer

Table 11-2192. CPSW_P1_MAC_TX_PAUSETIMER Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 2370h

Figure 11-975. CPSW_P1_MAC_TX_PAUSETIMER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																TX_PAUSETIMER															
R-0h																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2193. CPSW_P1_MAC_TX_PAUSETIMER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	TX_PAUSETIMER	R/W	0h	802.3 Tx Pause Timer Value – This field allows the contents of the transmit pause timer to be observed (and written in test mode). The transmit pause timer is loaded by a received (incoming) pause frame, and then decremented, at slottime intervals, down to zero at which time Ethernet Port transmit frames are again enabled. This register is for 802.3 Based flow control and is not used for 802.1qbb Priority Based Flow Control (PFC).

Table 11-2194. Register Call Summary for CPSW_P1_MAC_TX_PAUSETIMER

EMAC Registers
- NSS_0_CFG_CPSW Registers: [0]
- CPSW_P1_MAC_TX_PAUSETIMER Register (Offset = 2370h) [reset = 0h]: [0]

11.13.5.1.2.48 CPSW_P1_MAC_EMCONTROL Register (Offset = 23A0h) [reset = 0h]

CPSW_P1_MAC_EMCONTROL is shown in Figure 11-976 and described in Table 11-2196.

Enet Port 1 Emulation Control

Table 11-2195. CPSW_P1_MAC_EMCONTROL Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 23A0h

Figure 11-976. CPSW_P1_MAC_EMCONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						SOFT	FREE
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2196. CPSW_P1_MAC_EMCONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	SOFT	R/W	0h	Emulation Soft Bit
0	FREE	R/W	0h	Emulation Free Bit

Table 11-2197. Register Call Summary for CPSW_P1_MAC_EMCONTROL

Gigabit Ethernet MAC (EMAC) Subsystem • Emulation Control: [0]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P1_MAC_EMCONTROL Register (Offset = 23A0h) [reset = 0h]: [0]

11.13.5.1.2.49 CPSW_P1_MAC_TX_GAP Register (Offset = 23A4h) [reset = Ch]

CPSW_P1_MAC_TX_GAP is shown in Figure 11-977 and described in Table 11-2199.

Enet Port 1 Tx Inter Packet Gap

Table 11-2198. CPSW_P1_MAC_TX_GAP Instances

Instance	Physical Address
NSS_0_CFG_CPSW	0422 23A4h

Figure 11-977. CPSW_P1_MAC_TX_GAP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																TX_GAP															
R-0h																R/W-Ch															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2199. CPSW_P1_MAC_TX_GAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	TX_GAP	R/W	Ch	Transmit Inter-Packet Gap This is the default gap value and only bits 8:0 are used. This can be increased from 12 to increase the gap between packets.

Table 11-2200. Register Call Summary for CPSW_P1_MAC_TX_GAP

Gigabit Ethernet MAC (EMAC) Subsystem • Ethernet MAC Sliver (CPGMAC_SL): [0]
EMAC Registers • NSS_0_CFG_CPSW Registers: [0] • CPSW_P1_MAC_TX_GAP Register (Offset = 23A4h) [reset = Ch]: [0]

11.13.5.1.3 NSS_0_CFG_ALE Registers

Table 11-2202 lists the memory-mapped registers for the NSS_0_CFG_ALE. All register offset addresses not listed in Table 11-2202 should be considered as reserved locations and the register contents should not be modified.

Table 11-2201. NSS_0_CFG_ALE Instances

Instance	Base Address
NSS_0_CFG_ALE	0423 E000h

Table 11-2202. NSS_0_CFG_ALE Registers

Offset	Acronym	Register Name	NSS_0_CFG_ALE Physical Address	Section
0h	ALE_IDVER	ID / Version	0423 E000h	Section 11.13.5.1.3.1
4h	ALE_STATUS	Status	0423 E004h	Section 11.13.5.1.3.2
8h	ALE_CONTROL	Control	0423 E008h	Section 11.13.5.1.2.2
10h	ALE_PRESCALE	Prescale	0423 E010h	Section 11.13.5.1.3.4
14h	ALE_AGING_TIMER	Aging Timer	0423 E014h	Section 11.13.5.1.3.5
20h	ALE_TABLE_CONTROL	Table Control	0423 E020h	Section 11.13.5.1.3.6
34h	ALE_TABLE_WORD2	Table Word 2	0423 E034h	Section 11.13.5.1.3.7
38h	ALE_TABLE_WORD1	Table Word 1	0423 E038h	Section 11.13.5.1.3.8
3Ch	ALE_TABLE_WORD0	Table Word 0	0423 E03Ch	Section 11.13.5.1.3.9
40h to 5Ch	ALE_PORT_CONTROL_0 to ALE_PORT_CONTROL_7	Port N Control	0423 E040h to 0423 E05Ch	Section 11.13.5.1.3.10
90h	ALE_UNKNOWN_VLAN	Unknown VLAN Member List	0423 E090h	Section 11.13.5.1.3.11
94h	ALE_UNKNOWN_UREG_MCAST_FLOOD	Unknown VLAN Multicast Flood Mask	0423 E094h	Section 11.13.5.1.3.12
98h	ALE_UNKNOWN_REG_MCAST_FLOOD	Unknown VLAN Multicast Flood Mask	0423 E098h	Section 11.13.5.1.3.12
9Ch	ALE_FORCE_UNTAGGED_EGRESS	Unknown VLAN Force Untagged Egress	0423 E09Ch	Section 11.13.5.1.3.14
C0h to CCh	ALE_VLAN_MASK_MUX_0 to ALE_VLAN_MASK_MUX_3	VLAN Mask Mux n Select	0423 E0C0h to 0423 E0CCh	Section 11.13.5.1.3.15
100h	ALE_POLICER_PORT_OUI	Specifies the port, frame priority, and OUI address index as well as match enables for port, frame priority, and OUI address matching	0423 E100h	Section 11.13.5.1.3.16
104h	ALE_POLICER_DA_SA	Specifies the match enable/match index for the L2 destination and L2 source addresses	0423 E104h	Section 11.13.5.1.3.17
108h	ALE_POLICER_VLAN	Specifies the match enable/match index for the Outer VLAN and Inner VLAN addresses	0423 E108h	Section 11.13.5.1.3.18
10Ch	ALE_POLICER_ETHERTYPE_IPSA	Specifies the match enable/match index for the Ether Type and IP Source address	0423 E10Ch	Section 11.13.5.1.3.19
110h	ALE_POLICER_IPDA	Specifies the match enable/match index for the IP Destination address	0423 E110h	Section 11.13.5.1.3.20

Table 11-2202. NSS_0_CFG_ALE Registers (continued)

Offset	Acronym	Register Name	NSS_0_CFG_ALE Physical Address	Section
120h	ALE_POLICER_TBL_CTL	The Policing Table Control is used to read or write the selected policing/classifier entry	0423 E120h	Section 11.13.5.1.3.21
134h	ALE_THREAD_DEF	The THREAD Mapping Default Value register is used to set the default thread ID when no classifier is matched	0423 E134h	Section 11.13.5.1.3.22
138h	ALE_THREAD_CTL	The THREAD Mapping Control register allows the highest matched classifier to return a particular thread ID for traffic sent to the host	0423 E138h	Section 11.13.5.1.3.23
13Ch	ALE_THREAD_VAL	The THREAD Mapping Value register is used to set the thread ID for a particular classifier entry	0423 E13Ch	Section 11.13.5.1.3.24

11.13.5.1.3.1 ALE_IDVER Register (Offset = 0h) [reset = 291104h]

ALE_IDVER is shown in and described in Table 11-2204.

ID / Version

Table 11-2203. ALE_IDVER Instances

Instance	Physical Address
NSS_0_CFG_ALE	0423 E000h

Figure 11-978. ALE_IDVER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 291104h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2204. ALE_IDVER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	291104h	TI internal data. Identifies revision of peripheral.

Table 11-2205. Register Call Summary for ALE_IDVER

EMAC Registers
- ALE_IDVER Register (Offset = 0h) [reset = 291104h]: [0]
- NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.2 ALE_STATUS Register (Offset = 4h) [reset = 140h]

ALE_STATUS is shown in Figure 11-979 and described in Table 11-2207.

Status

Table 11-2206. ALE_STATUS Instances

Instance	Physical Address
NSS_0_CFG_ALE	0423 E004h

Figure 11-979. ALE_STATUS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POLICERS_DIV_8								RESERVED	32_ENTRIES	RESERVED	ENTRIES_DIV_1024				
R-1h								R-0h	R-1h	R-0	R-0h				

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2207. ALE_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	POLICERS_DIV_8	R	1h	This is the number of policers the ALE implements divided by 8 (a value of 4 indicates 32 policers total).
7	RESERVED	R	0h	Reserved
6	32_ENTRIES	R	1h	When set indicates that there are 32 entries in the ALE table.
5	RESERVED	R	0h	Reserved
4-0	ENTRIES_DIV_1024	R	0h	This is the number of table entries total divided by 1024. A value of 1 indicates 1024 table entries. A value of 8 indicates 8192 table entries. A value of 0 indicates less than 1024.

Table 11-2208. Register Call Summary for ALE_STATUS

Gigabit Ethernet MAC (EMAC) Subsystem • Address Lookup Engine (ALE): [0]
EMAC Registers • ALE_STATUS Register (Offset = 4h) [reset = 140h]: [0] • NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.3 ALE_CONTROL Register (Offset = 8h) [reset = 0h]

ALE_CONTROL is shown in and described in Table 11-2210.

Control

Table 11-2209. ALE_CONTROL Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E008h

Figure 11-980. ALE_CONTROL Register

31	30	29	28	27	26	25	24
ENABLE	CLEAR_TABLE	AGE_OUT_NOW	RESERVED				
R/W-0h	R/W-0h	R/W-0h	R-0h				
23	22	21	20	19	18	17	16
UPD_BW_CTL			RESERVED				
R/W-0h			R-0h				
15	14	13	12	11	10	9	8
RESERVED		UVLAN_NO_LEARN	RESERVED			RESERVED	UNI_FLOOD_TO_HOST
R-0h		R/W-0h	R/W-0h			R-0h	R/W-0h
7	6	5	4	3	2	1	0
LEARN_NO_VID	EN_VID0_MODE	ENABLE_OUI_DENY	BYPASS	RATE_LIMIT_TX	VLAN_AWARE	ENABLE_AUTH_MODE	ENABLE_RATE_LIMIT
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2210. ALE_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ENABLE	R/W	0h	Enable ALE • 0 – Drop all packets • 1 – Enable ALE packet processing
30	CLEAR_TABLE	R/W	0h	Clear ALE address table – Setting this bit causes the ALE hardware to write all table bit values to zero. Software must perform a clear table operation as part of the ALE setup/configuration process. Setting this bit causes all ALE accesses to be held up for 64 clocks while the clear is performed. Access to all ALE registers will be blocked (wait states) until the 64 clocks have completed. This bit cannot be read as one because the read is blocked until the clear table is completed at which time this bit is cleared to zero.
29	AGE_OUT_NOW	R/W	0h	Age Out Address Table Now – Setting this bit causes the ALE hardware to remove (free up) any ageable table entry that does not have a set touch bit. This bit is cleared when the age out process has completed. This bit may be read. The age out process takes 4096 clocks best case (no ale packet processing during ageout) and 66550 clocks absolute worst case.
28-24	RESERVED	R	0h	Reserved

Table 11-2210. ALE_CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-21	UPD_BW_CTL	R/W	0h	The upd_bw_ctrl field allows for up to 8 times the rate in which adds, updates, touches, writes, and aging updates can occur • 0 - 350Mhz, 5M • 1 - 359Mhz, 11M • 2 - 367Mhz, 16M • 3 - 375Mhz, 22M • 4 - 384Mhz, 28M • 5 - 392Mhz, 34M • 6 - 400Mhz, 39M • 7 - 409Mhz, 45M
20-14	RESERVED	R	0h	Reserved
13	UVLAN_NO_LEARN	R/W	0h	Unknown VLAN No Learn – when set this will prevent source addresses of unknown VLANIDs from being automatically added into the look up table if learning is enabled.
12-10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Read as zero
8	UNI_FLOOD_TO_HOST	R/W	0h	Enable Port 0 Unicast Flood • 0 – unknown unicast packets are dropped to the host • 1 – unknown unicast packets flood to host also.
7	LEARN_NO_VID	R/W	0h	Learn No VID • 0 – VID is learned with the source address • 1 – VID is not learned with the source address (source address is not tied to VID). Determines the entry type.
6	EN_VID0_MODE	R/W	0h	Enable VLAN ID = 0 Mode • 0 – Process the priority tagged packet with VID = PORT_VLAN[11:0]. • 1 – Process the priority tagged packet with VID = 0.
5	ENABLE_OUI_DENY	R/W	0h	Enable OUI Deny Mode
4	BYPASS	R/W	0h	ALE Bypass • 0 – no bypass • 1 – bypass the ALE
3	RATE_LIMIT_TX	R/W	0h	Rate Limit Transmit Mode • 0 – Broadcast and multicast rate limit counters are received port based • 1 – Broadcast and multicast rate limit counters are transmit port based.
2	VLAN_AWARE	R/W	0h	ALE VLAN Aware • 0 – Simple switch rules. • 1 – VLAN aware rules. Packets forwarded based on VLAN membership.
1	ENABLE_AUTH_MODE	R/W	0h	Enable MAC Authorization Mode – Mac authorization mode requires that all table entries be made by the host software. There are no learned address in authorization mode and the packet will be dropped if the source address is not found and the destination address is not an address with the super table entry bit set. • 0 – The ALE is not in MAC authorization mode • 1 – The ALE is in MAC authorization mode
0	ENABLE_RATE_LIMIT	R/W	0h	Enable Broadcast and Multicast Rate Limit • 0 – Broadcast/Multicast rates not limited • 1 – Broadcast/Multicast packet reception limited to the port control register rate limit fields.

Table 11-2211. Register Call Summary for ALE_CONTROL

Gigabit Ethernet MAC (EMAC) Subsystem • Address Lookup Engine (ALE): [0][1][2]
EMAC Registers • ALE_CONTROL Register (Offset = 8h) [reset = 0h]: [0] • NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.4 ALE_PRESCALE Register (Offset = 10h) [reset = 0h]

ALE_PRESCALE is shown in Figure 11-981 and described in Table 11-2213.

Prescale

Table 11-2212. ALE_PRESCALE Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E010h

Figure 11-981. ALE_PRESCALE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												PRESCALE																			
R-0h												R/W-0h																			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2213. ALE_PRESCALE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-0	PRESCALE	R/W	0h	ALE Prescale – The input clock is divided by this value for use in the multicast/broadcast rate limiters. The minimum operating value is 0x10. The prescaler is off when the value is zero.

Table 11-2214. Register Call Summary for ALE_PRESCALE

EMAC Registers

- ALE_PRESCALE Register (Offset = 10h) [reset = 0h]: [0]
- NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.5 ALE_AGING_TIMER Register (Offset = 14h) [reset = 0h]

ALE_AGING_TIMER is shown in Figure 11-982 and described in Table 11-2216.

Aging Timer

Table 11-2215. ALE_AGING_TIMER Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E014h

Figure 11-982. ALE_AGING_TIMER Register

31	30	29	28	27	26	25	24
PRESCALE_1_DISABLE	PRESCALE_2_DISABLE	RESERVED					
R/W-0h	R/W-0h	R-0h					
23	22	21	20	19	18	17	16
AGING_TIMER							
R/W-0h							
15	14	13	12	11	10	9	8
AGING_TIMER							
R/W-0h							
7	6	5	4	3	2	1	0
AGING_TIMER							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2216. ALE_AGING_TIMER Register Field Descriptions

Bit	Field	Type	Reset	Description
31	PRESCALE_1_DISABLE	R/W	0h	ALE Prescaler 1 Disable: When set, removes 1,000 from the auto aging multiplier.
30	PRESCALE_2_DISABLE	R/W	0h	ALE Prescaler 2 Disable: When set, removes 1,000 from the auto aging multiplier.
29-24	RESERVED	R	0h	Reserved
23-0	AGING_TIMER	R/W	0h	ALE Aging Timer: When non-zero, auto-aging is enabled. The ale_aging_timer value (minus 1) times 1,000,000 is the number of clock cycles after which auto-aging will automatically be initiated. If either prescale_1_disable or prescale_2_disable is set then the multiplier is 1,000 instead of 1,000,000. If both prescale_1_disable and prescale_2_disable are set then the multiplier is 1 instead of 1,000,000. Auto aging is initiated each time the count reaches zero. When the ale_aging_timer is zero, auto-aging is disabled (but software can still initiate aging by setting age_out_now). There is no auto-aging if ale_enable is zero or if clear_table is set or if age_out_now is set (while the events are occurring).

Table 11-2217. Register Call Summary for ALE_AGING_TIMER

EMAC Registers

- ALE_AGING_TIMER Register (Offset = 14h) [reset = 0h]: [0]
- NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.6 ALE_TABLE_CONTROL Register (Offset = 20h) [reset = 0h]

ALE_TABLE_CONTROL is shown in Figure 11-983 and described in Table 11-2219.

Table Control

Table 11-2218. ALE_TABLE_CONTROL Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E020h

Figure 11-983. ALE_TABLE_CONTROL Register

31	30	29	28	27	26	25	24
WRITE_RDZ	RESERVED						
R/W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0							
15	14	13	12	11	10	9	8
RESERVED				ENTRY_POINTER			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
ENTRY_POINTER							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2219. ALE_TABLE_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	WRITE_RDZ	R/W	0h	Write Bit – This bit is always . Writing a 1 to this bit causes the three table word register values to be written to the entry_pointer location in the address table. Writing a 0 to this bit causes the three table word register values to be loaded from the entry_pointer location in the address table so that they may be subsequently read. A read of any ALE address location will be stalled until the read or write has completed.
30-14	RESERVED	R	0h	Reserved
13-0	ENTRY_POINTER	R/W	0h	Table Entry Pointer - The ENTRY_POINTER contains the table entry value that will be read/written with accesses to the table word registers. The number of entries in the table determine the width of the entry pointer field (y). y = 9 for 1024 entries.

Table 11-2220. Register Call Summary for ALE_TABLE_CONTROL

EMAC Registers
- ALE_TABLE_CONTROL Register (Offset = 20h) [reset = 0h]: [0]
- NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.7 ALE_TABLE_WORD2 Register (Offset = 34h) [reset = 0h]

ALE_TABLE_WORD2 is shown in Figure 11-984 and described in Table 11-2222.

Table Word 2

Table 11-2221. ALE_TABLE_WORD2 Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E034h

Figure 11-984. ALE_TABLE_WORD2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									ENTRY_70_64						
R-0h									R/W-						

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2222. ALE_TABLE_WORD2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6-0	ENTRY_70_64	R/W	-	Table entry bits 70 down to 64

Table 11-2223. Register Call Summary for ALE_TABLE_WORD2

EMAC Registers
- ALE_TABLE_WORD2 Register (Offset = 34h) [reset = 0h]: [0]
- NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.8 ALE_TABLE_WORD1 Register (Offset = 38h) [reset = 0h]

table_word1 is shown in Figure 11-985 and described in Table 11-2225.

Table Word 1

Table 11-2224. ALE_TABLE_WORD1 Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E038h

Figure 11-985. ALE_TABLE_WORD1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENTRY_63_32																															
R/W-																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2225. ALE_TABLE_WORD1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ENTRY_63_32	R/W	-	Table Entry Bits 63:32

Table 11-2226. Register Call Summary for ALE_TABLE_WORD1

EMAC Registers
- NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.9 ALE_TABLE_WORD0 Register (Offset = 3Ch) [reset = 0h]

ALE_TABLE_WORD0 is shown in Figure 11-986 and described in Table 11-2228.

Table Word 0

Table 11-2227. ALE_TABLE_WORD0 Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E03Ch

Figure 11-986. table_word0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENTRY_31_0																															
R/W-																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2228. ALE_TABLE_WORD0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ENTRY_31_0	R/W	-	Table Entry Bits 31:0

Table 11-2229. Register Call Summary for ALE_TABLE_WORD0

EMAC Registers
- ALE_TABLE_WORD0 Register (Offset = 3Ch) [reset = 0h]: [0]
- NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.10 ALE_PORT_CONTROL_0 to ALE_PORT_CONTROL_7 Register (Offset = 40h to 5Ch) [reset = 0h]

ALE_PORT_CONTROL_0 to ALE_PORT_CONTROL_7 is shown in Figure 11-987 and described in Table 11-2231.

Port N Control

Table 11-2230. ALE_PORT_CONTROL_0 to ALE_PORT_CONTROL_7 Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E040h to 0421 E05Ch

Figure 11-987. ALE_PORT_CONTROL_0 to ALE_PORT_CONTROL_7 Register

31	30	29	28	27	26	25	24
BCAST_LIMIT							
R/W-0h							
23	22	21	20	19	18	17	16
MCAST_LIMIT							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED			DISABLE_AUTH_MODE	RESERVED	RESERVED		
R-0h		R-0h	R/W-0h	R-0h	R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	RESERVED	NO_SA_UPDATE	NO_LEARN	VID_INGRESS_CHECK	DROP_UNTAGGED	PORT_STATE	
R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2231. ALE_PORT_CONTROL_0 to ALE_PORT_CONTROL_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	BCAST_LIMIT	R/W	0h	Broadcast Packet Rate Limit Each prescale pulse loads this field into the port broadcast rate limit counter. The port counters are decremented with each packet received or transmitted depending on whether the mode is transmit or receive. If the counters decrement to zero, then further packets are rate limited until the next prescale pulse. Broadcast rate limiting is enabled by a non-zero value in this field.
23-16	MCAST_LIMIT	R/W	0h	Multicast Packet Rate Limit Each prescale pulse loads this field into the port multicast rate limit counter. The port counters are decremented with each packet received or transmitted depending on whether the mode is transmit or receive. If the counters decrement to zero, then further packets are rate limited until the next prescale pulse. Multicast rate limiting is enabled by a non-zero value in this field.
15-13	RESERVED	R	0h	Reserved
12	DISABLE_AUTH_MODE	R/W	0h	Disable MAC Authorization Mode When set, this bit disables MAC authorization mode for this port (when enable_auth_mode in ALE_Control is set).
11-6	RESERVED	R	0h	Reads return 0.
5	NO_SA_UPDATE	R/W	0h	No Source Address Update When set an ingress packet on this port will not cause a matching source address entry port number to be changed (to this port). When cleared, an ingress packet on this port will cause a matching source address entry port number to be changed to this port number (if it was previously a different port).

Table 11-2231. ALE_PORT_CONTROL_0 to ALE_PORT_CONTROL_7 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	NO_LEARN	R/W	0h	No Learn Mode When set the port is disabled from learning source addresses.
3	VID_INGRESS_CHECK	R/W	0h	VLAN ID Ingress Check If VLAN found and the receive port is not a VLAN member then drop the packet.
2	DROP_UNTAGGED	R/W	0h	Drop Untagged Packets - Drop non-VLAN tagged ingress packets
1-0	PORT_STATE	R/W	0h	Port State • 0 – Disabled • 1 – Blocked • 2 – Learn • 3 – Forward

Table 11-2232. Register Call Summary for ALE_PORT_CONTROL_0

Gigabit Ethernet MAC (EMAC) Subsystem • Address Lookup Engine (ALE): [0]
EMAC Registers • ALE_PORT_CONTROL_0 to ALE_PORT_CONTROL_7 Register (Offset = 40h to 5Ch) [reset = 0h]: [0] • NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.11 ALE_UNKNOWN_VLAN Register (Offset = 90h) [reset = 0h]

ALE_UNKNOWN_VLAN is shown in Figure 11-988 and described in Table 11-2234.

Unknown VLAN Member List

Table 11-2233. ALE_UNKNOWN_VLAN Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E090h

Figure 11-988. ALE_UNKNOWN_VLAN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														LIST	
R-0h																														R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2234. ALE_UNKNOWN_VLAN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reads return 0 and writes have no effect.
1-0	LIST	R/W	0h	Unknown VLAN Member List

Table 11-2235. Register Call Summary for ALE_UNKNOWN_VLAN

EMAC Registers
- ALE_UNKNOWN_VLAN Register (Offset = 90h) [reset = 0h]: [0]
- NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.12 ALE_UNKNOWN_UREG_MCAST_FLOOD Register (Offset = 94h) [reset = 0h]

ALE_UNKNOWN_MCAST_FLOOD is shown in Figure 11-990 and described in Table 11-2240.

Unknown VLAN Unregistered Multicast Flood Mask

Table 11-2236. ALE_UNKNOWN_UREG_MCAST_FLOOD Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E094h

Figure 11-989. ALE_UNKNOWN_UREG_MCAST_FLOOD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														MASK	
R-0h																														R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2237. ALE_UNKNOWN_UREG_MCAST_FLOOD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	MASK	R/W	0h	Unknown VLAN Multicast Flood Mask

Table 11-2238. Register Call Summary for ALE_UNKNOWN_UREG_MCAST_FLOOD

EMAC Registers
- NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.13 ALE_UNKNOWN_REG_MCAST_FLOOD Register (Offset = 98h) [reset = 0h]

ALE_UNKNOWN_MCAST_FLOOD is shown in Figure 11-990 and described in Table 11-2240.

Unknown VLAN Registered Multicast Flood Mask

Table 11-2239. ALE_UNKNOWN_REG_MCAST_FLOOD Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E098h

Figure 11-990. ALE_UNKNOWN_REG_MCAST_FLOOD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														MASK	
R-0h																														R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2240. ALE_UNKNOWN_REG_MCAST_FLOOD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	MASK	R/W	0h	Unknown VLAN Registered Multicast Flood Mask

Table 11-2241. Register Call Summary for ALE_UNKNOWN_REG_MCAST_FLOOD

EMAC Registers
• NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.14 ALE_FORCE_UNTAGGED_EGRESS Register (Offset = 9Ch) [reset = 0h]

ALE_FORCE_UNTAGGED_EGRESS is shown in Figure 11-991 and described in Table 11-2243.

Unknown VLAN Force Untagged Egress

Table 11-2242. ALE_FORCE_UNTAGGED_EGRESS Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E09Ch

Figure 11-991. ALE_FORCE_UNTAGGED_EGRESS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														MASK	
R-0h																														R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2243. ALE_FORCE_UNTAGGED_EGRESS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	MASK	R/W	0h	Force Untagged Egress Mask

Table 11-2244. Register Call Summary for ALE_FORCE_UNTAGGED_EGRESS

EMAC Registers
- ALE_FORCE_UNTAGGED_EGRESS Register (Offset = 9Ch) [reset = 0h]: [0]
- NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.15 ALE_VLAN_MASK_MUX_0 to ALE_VLAN_MASK_MUX_3 Register (Offset = C0h to CCh) [reset = 0h]

ALE_VLAN_MASK_MUX_0 to ALE_VLAN_MASK_MUX_3 is shown in Figure 11-992 and described in Table 11-2246.

VLAN Mask Mux n Select

Table 11-2245. ALE_VLAN_MASK_MUX_0 to ALE_VLAN_MASK_MUX_3 Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E0C0h to 0421 E0CCh

Figure 11-992. ALE_VLAN_MASK_MUX_0 to ALE_VLAN_MASK_MUX_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														MASK	
R-0h																														R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2246. ALE_VLAN_MASK_MUX_0 to ALE_VLAN_MASK_MUX_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reads return 0 and writes have no effect.
1-0	MASK	R/W	0h	Force Untagged Egress Mask

Table 11-2247. Register Call Summary for ALE_VLAN_MASK_MUX_0

Gigabit Ethernet MAC (EMAC) Subsystem • Address Lookup Engine (ALE): [0][1]
EMAC Registers • ALE_VLAN_MASK_MUX_0 to ALE_VLAN_MASK_MUX_3 Register (Offset = C0h to CCh) [reset = 0h]: [0] • NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.16 ALE_POLICER_PORT_OUI Register (Offset = 100h) [reset = 0h]

ALE_POLICER_PORT_OUI is shown in Figure 11-993 and described in Table 11-2249.

Specifies the port, frame priority, and OUI address index as well as match enables for port, frame priority, and OUI address matching

Table 11-2248. ALE_POLICER_PORT_OUI Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E100h

Figure 11-993. ALE_POLICER_PORT_OUI Register

31	30	29	28	27	26	25	24
PORT_MEN	RESERVED		PORT_NUM				RESERVED
R/W-0h	R-0h		R/W-0h				R-0h
23	22	21	20	19	18	17	16
RESERVED				PRI_MEN	PRI_VAL		
R-0h				R/W-0h	R/W-0h		
15	14	13	12	11	10	9	8
OUI_MEN	RESERVED		OUI_INDEX				
R/W-0h	R-0h		R/W-0h				
7	6	5	4	3	2	1	0
OUI_INDEX							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2249. ALE_POLICER_PORT_OUI Register Field Descriptions

Bit	Field	Type	Reset	Description
31	PORT_MEN	R/W	0h	Enabled port match for the selected policing/classifier entry
30-29	RESERVED	R	0h	Reserved
28-25	PORT_NUM	R/W	0h	If enabled, specifies the port address to match for the selected policing/classifier entry
24-20	RESERVED	R	0h	Reserved
19	PRI_MEN	R/W	0h	Enabled frame priority match for the selected policing/classifier entry
18-16	PRI_VAL	R/W	0h	If enabled, specifies the frame priority to match for the selected policing/classifier entry
15	OUI_MEN	R/W	0h	Enabled frame OUI address match for the selected policing/classifier entry
14-13	RESERVED	R	0h	Reserved
12-0	OUI_INDEX	R/W	0h	If enabled, specifies the ALE OUI address lookup table index to match for the selected policing/classifier entry

Table 11-2250. Register Call Summary for ALE_POLICER_PORT_OUI

EMAC Registers
- ALE_POLICER_PORT_OUI Register (Offset = 100h) [reset = 0h]: [0]
- NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.17 ALE_POLICER_DA_SA Register (Offset = 104h) [reset = 0h]

ALE_POLICER_DA_SA is shown in Figure 11-994 and described in Table 11-2252.

Specifies the match enable/match index for the L2 destination and L2 source addresses

Table 11-2251. ALE_POLICER_DA_SA Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E104h

Figure 11-994. ALE_POLICER_DA_SA Register

31	30	29	28	27	26	25	24
DST_MEN	RESERVED		DST_INDEX				
R/W-0h	R-0h		R/W-0h				
23	22	21	20	19	18	17	16
DST_INDEX							
R/W-0h							
15	14	13	12	11	10	9	8
SRC_MEN	RESERVED		SRC_INDEX				
R/W-0h	R-0h		R/W-0h				
7	6	5	4	3	2	1	0
SRC_INDEX							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2252. ALE_POLICER_DA_SA Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DST_MEN	R/W	0h	Enabled frame L2 destination address match for the selected policing/classifier entry
30-29	RESERVED	R	0h	Reads return 0 and writes have no effect.
28-16	DST_INDEX	R/W	0h	If enabled, specifies the ALE L2 destination address lookup table index to match for the selected policing/classifier entry
15	SRC_MEN	R/W	0h	Enabled frame L2 source address match for the selected policing/classifier entry
14-13	RESERVED	R	0h	Reads return 0 and writes have no effect.
12-0	SRC_INDEX	R/W	0h	If enabled, specifies the ALE L2 source address lookup table index to match for the selected policing/classifier entry

Table 11-2253. Register Call Summary for ALE_POLICER_DA_SA

EMAC Registers
- ALE_POLICER_DA_SA Register (Offset = 104h) [reset = 0h]: [0]
- NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.18 ALE_POLICER_VLAN Register (Offset = 108h) [reset = 0h]

ALE_POLICER_VLAN is shown in Figure 11-995 and described in Table 11-2255.

Specifies the match enable/match index for the Outer VLAN and Inner VLAN addresses

Table 11-2254. ALE_POLICER_VLAN Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E108h

Figure 11-995. ALE_POLICER_VLAN Register

31	30	29	28	27	26	25	24
OVLAN_MEN	RESERVED		OVLAN_INDEX				
R/W-0h	R-0h		R/W-0h				
23	22	21	20	19	18	17	16
OVLAN_INDEX							
R/W-0h							
15	14	13	12	11	10	9	8
IVLAN_MEN	RESERVED		IVLAN_INDEX				
R/W-0h	R-0h		R/W-0h				
7	6	5	4	3	2	1	0
IVLAN_INDEX							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2255. ALE_POLICER_VLAN Register Field Descriptions

Bit	Field	Type	Reset	Description
31	OVLAN_MEN	R/W	0h	Enabled frame Outer VLAN address match for the selected policing/classifier entry
30-29	RESERVED	R	0h	Reserved
28-16	OVLAN_INDEX	R/W	0h	If enabled, specifies the ALE Outer VLAN address lookup table index to match for the selected policing/classifier entry
15	IVLAN_MEN	R/W	0h	Enabled frame Inner VLAN address match for the selected policing/classifier entry
14-13	RESERVED	R	0h	Reserved
12-0	IVLAN_INDEX	R/W	0h	If enabled, specifies the ALE Inner VLAN address lookup table index to match for the selected policing/classifier entry

Table 11-2256. Register Call Summary for ALE_POLICER_VLAN

EMAC Registers
- NSS_0_CFG_ALE Registers: [0]
- ALE_POLICER_VLAN Register (Offset = 108h) [reset = 0h]: [0]

11.13.5.1.3.19 ALE_POLICER_ETHERTYPE_IPSA Register (Offset = 10Ch) [reset = 0h]

policer_ethertype_ipsa is shown in Figure 11-996 and described in Table 11-2258.

Specifies the match enable/match index for the Ether Type and IP Source address

Table 11-2257. ALE_POLICER_ETHERTYPE_IPSA Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E10Ch

Figure 11-996. ALE_POLICER_ETHERTYPE_IPSA Register

31	30	29	28	27	26	25	24
ETHERTYPE_MEN	RESERVED		ETHERTYPE_INDEX				
R/W-0h	R-0h		R/W-0h				
23	22	21	20	19	18	17	16
ETHERTYPE_INDEX							
R/W-0h							
15	14	13	12	11	10	9	8
IPSRC_MEN	RESERVED		IPSRC_INDEX				
R/W-0h	R-0h		R/W-0h				
7	6	5	4	3	2	1	0
IPSRC_INDEX							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2258. ALE_POLICER_ETHERTYPE_IPSA Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ETHERTYPE_MEN	R/W	0h	Enabled frame Ether Type match for the selected policing/classifier entry
30-29	RESERVED	R	0h	Reserved
28-16	ETHERTYPE_INDEX	R/W	0h	If enabled, specifies the ALE Ether Type lookup table index to match for the selected policing/classifier entry
15	IPSRC_MEN	R/W	0h	Enabled frame IP Source address match for the selected policing/classifier entry
14-13	RESERVED	R	0h	Reserved
12-0	IPSRC_INDEX	R/W	0h	If enabled, specifies the ALE IP Source address lookup table index to match for the selected policing/classifier entry

Table 11-2259. Register Call Summary for ALE_POLICER_ETHERTYPE_IPSA

EMAC Registers

- NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.20 ALE_POLICER_IPDA Register (Offset = 110h) [reset = 0h]

ALE_POLICER_IPDA is shown in Figure 11-997 and described in Table 11-2261.

Specifies the match enable/match index for the IP Destination address

Table 11-2260. ALE_POLICER_IPDA Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E110h

Figure 11-997. ALE_POLICER_IPDA Register

31	30	29	28	27	26	25	24
IPDST_MEN	RESERVED		IPDST_INDEX				
R/W-0h	R-0h		R/W-0h				
23	22	21	20	19	18	17	16
IPDST_INDEX							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2261. ALE_POLICER_IPDA Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPDST_MEN	R/W	0h	Enabled frame IP Destination address match for the selected policing/classifier entry
30-29	RESERVED	R	0h	Reserved
28-16	IPDST_INDEX	R/W	0h	If enabled, specifies the ALE IP Destination address lookup table index to match for the selected policing/classifier entry
15-0	RESERVED	R	0h	Reserved

Table 11-2262. Register Call Summary for ALE_POLICER_IPDA

EMAC Registers
- ALE_POLICER_IPDA Register (Offset = 110h) [reset = 0h]: [0]
- NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.21 ALE_POLICER_TBL_CTL Register (Offset = 120h) [reset = 0h]

ALE_POLICER_TBL_CTL is shown in Figure 11-998 and described in Table 11-2264.

The Policing Table Control is used to read or write the selected policing/classifier entry

Table 11-2263. ALE_POLICER_TBL_CTL Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E120h

Figure 11-998. ALE_POLICER_TBL_CTL Register

31	30	29	28	27	26	25	24
WRITE_ENABLE	RESERVED						
R/W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				POL_TBL_INDEX			
R-0h				R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2264. ALE_POLICER_TBL_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	WRITE_ENABLE	R/W	0h	Setting this bit will write the POLICER_CFG0-7 to the pol_tbl_index selected policing/classifier entry
30-5	RESERVED	R	0h	Reads return 0 and writes have no effect.
4-0	POL_TBL_INDEX	R/W	0h	This field specifies the policing/classifier entry to be read or written

Table 11-2265. Register Call Summary for ALE_POLICER_TBL_CTL

EMAC Registers
- NSS_0_CFG_ALE Registers: [0]
- ALE_POLICER_TBL_CTL Register (Offset = 120h) [reset = 0h]: [0]

11.13.5.1.3.22 ALE_THREAD_DEF Register (Offset = 134h) [reset = 0h]

ALE_THREAD_DEF is shown in Figure 11-999 and described in Table 11-2267.

The THREAD Mapping Default Value register is used to set the default thread ID when no classifier is matched

Table 11-2266. ALE_THREAD_DEF Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E134h

Figure 11-999. ALE_THREAD_DEF Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
ENABLE	RESERVED						
R/W-0h	R-0h						
7	6	5	4	3	2	1	0
RESERVED		VALUE					
R-0h		R/W-0h					

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2267. ALE_THREAD_DEF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	ENABLE	R/W	0h	This field is the read or written enable for the Default Thread ID. When set, the switch will use the def_thread_val for the default thread ID if no classifier is matched. If clear, the switch will generate its own thread ID based on port and priority if there is no classifier match.
14-6	RESERVED	R	0h	Reserved
5-0	VALUE	R/W	0h	This field is the read or written default thread ID value that is used for host traffic.

Table 11-2268. Register Call Summary for ALE_THREAD_DEF

Gigabit Ethernet MAC (EMAC) Subsystem • Address Lookup Engine (ALE): [0][1]
EMAC Registers • ALE_THREAD_DEF Register (Offset = 134h) [reset = 0h]: [0] • NSS_0_CFG_ALE Registers: [0]

11.13.5.1.3.23 ALE_THREAD_CTL Register (Offset = 138h) [reset = 0h]

ALE_THREAD_CTL is shown in Figure 11-1000 and described in Table 11-2270.

The THREAD Mapping Control register allows the highest matched classifier to return a particular thread ID for traffic sent to the host

Table 11-2269. ALE_THREAD_CTL Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E138h

Figure 11-1000. ALE_THREAD_CTL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											ENTRY_PTR				
R-0h											R/W-0h				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2270. ALE_THREAD_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reads return 0 and writes have no effect.
4-0	ENTRY_PTR	R/W	0h	Policer Thread Entry Pointer. This field specificies the policer/classifier thread map entry to be read or written. This field is written with the policer/classifier number that will be read or written with a read or write to the ALE_Thread_Map register.

Table 11-2271. Register Call Summary for ALE_THREAD_CTL

Gigabit Ethernet MAC (EMAC) Subsystem • Address Lookup Engine (ALE): [0][1]
EMAC Registers • NSS_0_CFG_ALE Registers: [0] • ALE_THREAD_CTL Register (Offset = 138h) [reset = 0h]: [0]

11.13.5.1.3.24 ALE_THREAD_VAL Register (Offset = 13Ch) [reset = 0h]

ALE_THREAD_VAL is shown in Figure 11-1001 and described in Table 11-2273.

The THREAD Mapping Value register is used to set the thread ID for a particular classifier entry

Table 11-2272. ALE_THREAD_VAL Instances

Instance	Physical Address
NSS_0_CFG_ALE	0421 E13Ch

Figure 11-1001. ALE_THREAD_VAL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
ENABLE	RESERVED						
R/W-0h	R-0h						
7	6	5	4	3	2	1	0
RESERVED			VALUE				
R-0h			R/W-0h				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2273. ALE_THREAD_VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	ENABLE	R/W	0h	This field is the read or written enable for the Thread ID. When set, the switch will use the thread_val for the particular classifier match. If clear, the switch will generate its own thread ID based on port and priority.
14-6	RESERVED	R	0h	Reserved
5-0	VALUE	R/W	0h	This field is the read or written thread ID value that is used to map a classifier hit to thread ID for host traffic.

Table 11-2274. Register Call Summary for ALE_THREAD_VAL

Gigabit Ethernet MAC (EMAC) Subsystem • Address Lookup Engine (ALE): [0][1]
EMAC Registers • ALE_THREAD_VAL Register (Offset = 13Ch) [reset = 0h]: [0] • NSS_0_CFG_ALE Registers: [0]

11.13.5.1.4 NSS_0_CFG_CPTS Registers

Table 11-2276 lists the memory-mapped registers for the NSS_0_CFG_CPTS. All register offset addresses not listed in Table 11-2276 should be considered as reserved locations and the register contents should not be modified.

Table 11-2275. NSS_0_CFG_CPTS Instances

Instance	Base Address
NSS_0_CFG_CPTS	0423 D000h

Table 11-2276. NSS_0_CFG_CPTS Registers

Offset	Acronym	Register Name	NSS_0_CFG_CPTS Physical Address	Section
0h	CPTS_IDVER	Identification and Version Register	0423 D000h	Section 11.13.5.1.4.1
4h	CPTS_CONTROL	Time Sync Control Register	0423 D004h	Section 11.13.5.1.4.2
8h	CPTS_RFTCLK_SEL	RFTCLK Select Register	0423 D008h	Section 11.13.5.1.4.3
Ch	CPTS_TS_PUSH	Time Stamp Event Push Register	0423 D00Ch	Section 11.13.5.1.4.4
10h	CPTS_TS_LOAD_LOW_VAL	Time Stamp Load Value Register	0423 D010h	Section 11.13.5.1.4.5
14h	CPTS_TS_LOAD_EN	Time Stamp Load Enable Register	0423 D014h	Section 11.13.5.1.4.6
18h	CPTS_TS_COMP_LOW_VAL	Time Stamp Comparison Value Register	0423 D018h	Section 11.13.5.1.4.7
1Ch	CPTS_TS_COMP_LEN	Time Stamp Comparison Length Register	0423 D01Ch	Section 11.13.5.1.4.8
20h	CPTS_INTSTAT_RAW	Interrupt Status Register Raw	0423 D020h	Section 11.13.5.1.4.9
24h	CPTS_INTSTAT_MASKED	Interrupt Status Register Masked	0423 D024h	Section 11.13.5.1.4.10
28h	CPTS_INT_ENABLE	Interrupt Enable Register	0423 D028h	Section 11.13.5.1.4.11
2Ch	CPTS_TS_COMP_NUDGE	Timestamp Comparison Nudge Register	0423 D02Ch	Section 11.13.5.1.4.12
30h	CPTS_EVENT_POP	Event Pop Register	0423 D030h	Section 11.13.5.1.4.13
34h	CPTS_EVENT_0	Event_0 Register	0423 D034h	Section 11.13.5.1.4.14
38h	CPTS_EVENT_1	Event_1 Register	0423 D038h	Section 11.13.5.1.4.15
3Ch	CPTS_EVENT_2	Event_2 Register	0423 D03Ch	Section 11.13.5.1.4.16
40h	CPTS_EVENT_3	Event_3 Register	0423 D040h	Section 11.13.5.1.4.17
44h	CPTS_TS_LOAD_HIGH_VAL	Time Stamp Load High Value Register	0423 D044h	Section 11.13.5.1.4.18
48h	CPTS_TS_COMP_HIGH_VAL	Time Stamp Comparison High Value Register	0423 D048h	Section 11.13.5.1.4.19

11.13.5.1.4.1 CPTS_IDVER Register (Offset = 0h) [reset = 4E8A0107h]

CPTS_IDVER is shown in and described in Figure 11-1002.

Identification and Version Register

Table 11-2277. CPTS_IDVER Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D000h

Figure 11-1002. CPTS_IDVER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 4E8A0107h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2278. CPTS_IDVER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4E8A0107h	TI internal data. Identifies revision of peripheral.

Table 11-2279. Register Call Summary for CPTS_IDVER

EMAC Registers
- NSS_0_CFG_CPTS Registers: [0]
- CPTS_IDVER Register (Offset = 0h) [reset = 4E8A0107h]: [0]

11.13.5.1.4.2 CPTS_CONTROL Register (Offset = 4h) [reset = 4h]

CPTS_CONTROL is shown in and described in Table 11-2281.

Time Sync Control Register

Table 11-2280. CPTS_CONTROL Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D004h

Figure 11-1003. CPTS_CONTROL Register

31	30	29	28	27	26	25	24
TS_SYNC_SEL				RESERVED			
R/W-0h				R-0h			
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
HW8_TS_PUSH_EN	HW7_TS_PUSH_EN	HW6_TS_PUSH_EN	HW5_TS_PUSH_EN	HW4_TS_PUSH_EN	HW3_TS_PUSH_EN	HW2_TS_PUSH_EN	HW1_TS_PUSH_EN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	TS_COMP_TOG	64-BIT_MODE	SEQUENCE_EN	TSTAMP_EN	TS_COMP_POLARITY	INT_TEST	CPTS_EN
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2281. CPTS_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	TS_SYNC_SEL	R/W	0h	TS_SYNC output timestamp counter bit select • 0000 – TS_SYNC disabled • 0001 - 1111 – TS_SYNC is timestamp counter bits 31 (1111) down to 17 (0001)
27-16	RESERVED	R	0h	read as zero.
15	HW8_TS_PUSH_EN	R/W	0h	Hardware push 8 enable
14	HW7_TS_PUSH_EN	R/W	0h	Hardware push 7 enable
13	HW6_TS_PUSH_EN	R/W	0h	Hardware push 6 enable
12	HW5_TS_PUSH_EN	R/W	0h	Hardware push 5 enable
11	HW4_TS_PUSH_EN	R/W	0h	Hardware push 4 enable
10	HW3_TS_PUSH_EN	R/W	0h	Hardware push 3 enable
9	HW2_TS_PUSH_EN	R/W	0h	Hardware push 2 enable
8	HW1_TS_PUSH_EN	R/W	0h	Hardware push 1 enable
7	RESERVED	R	0h	Read as zero.
6	TS_COMP_TOG	R/W	0h	Timestamp Compare Toggle mode • 0 – TS_COMP is in non-toggle mode • 1 - TS_COMP is in toggle mode
5	64-BIT_MODE	R/W	0h	64-Bit Mode • 0 – The timestamp is 32-bits with the upper 32-bits forced to zero. • 1 - The timestamp is 64-bits.

Table 11-2281. CPTS_CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	SEQUENCE_EN	R/W	0h	Sequence Enable • 0 – The timestamp value increments with the selected RFTCLK • 1 - The timestamp for received packets is the sequence number of the received packet (first packet is 0, second packet is 1, etc).
3	TSTAMP_EN	R/W	0h	Host Receive Timestamp Enable • 0 – Timestamps are disabled on received packets to host • 1 - Timestamps enabled on received packets to host (cpts_en must be set)
2	TS_COMP_POLARITY	R/W	1h	TS_COMP polarity • 0 – TS_COMP is asserted low • 1 - TS_COMP is asserted high
1	INT_TEST	R/W	0h	Interrupt test - When set, this bit allows the raw interrupt to be written to facilitate interrupt test.
0	CPTS_EN	R/W	0h	Time sync enable – When disabled (cleared to zero), the RCLK domain is held in reset.

Table 11-2282. Register Call Summary for CPTS_CONTROL

Gigabit Ethernet MAC (EMAC) Subsystem • Time Sync Events: [0] • 64-bit Time Stamp Value: [0] • Timestamp Sync Output: [0][1] • CPTS Initialization: [0][1] • Timestamp Compare Output: [0][1][2]
EMAC Registers • CPTS_RFTCLK_SEL Register (Offset = 8h) [reset = 0h]: [0] • NSS_0_CFG_CPTS Registers: [0] • CPTS_CONTROL Register (Offset = 4h) [reset = 4h]: [0]

11.13.5.1.4.3 CPTS_RFTCLK_SEL Register (Offset = 8h) [reset = 0h]

CPTS_RFTCLK_SEL is shown in Figure 11-1004 and described in Table 11-2284.

RFTCLK Select Register

Table 11-2283. CPTS_RFTCLK_SEL Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D008h

Figure 11-1004. CPTS_RFTCLK_SEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											RFTCLK_SEL				
R-0h											R/W-0h				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2284. CPTS_RFTCLK_SEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Read as zero.
4-0	RFTCLK_SEL	R/W	0h	Reference clock select: • 0h = NSS/IEP PLL /4 • 1h = CHIP_CLK1 /3 • 2h = TIMI0 pin • 3h = TIMI1 pin • 4h = CHIP_CLK1 /2 • 8h = CPTS_REFCLK_P/N pin • Others = Reserved This RFTCLK_SEL value can be written only when the CPTS_EN bit and the TSTAMP_EN bit are cleared to zero in the CPTS_CONTROL register.

Table 11-2285. Register Call Summary for CPTS_RFTCLK_SEL

Gigabit Ethernet MAC (EMAC) Subsystem • EMAC Integration: [0][1] • CPTS Initialization: [0]
EMAC Registers • CPTS_RFTCLK_SEL Register (Offset = 8h) [reset = 0h]: [0] • NSS_0_CFG_CPTS Registers: [0]

11.13.5.1.4.4 CPTS_TS_PUSH Register (Offset = Ch) [reset = 0h]

ts_push is shown in Figure 11-1005 and described in Table 11-2287.

Time Stamp Event Push Register

Table 11-2286. CPTS_TS_PUSH Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D00Ch

Figure 11-1005. CPTS_TS_PUSH Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							TS_PUSH
R-0h							W-0h

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 11-2287. CPTS_TS_PUSH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Read as zero.
0	TS_PUSH	W	0h	Time stamp event push – When a logic high is written to this bit a time stamp event is pushed onto the event FIFO. The time stamp value is the time of the write of this register, not the time of the event read. The time stamp value can then be read on interrupt via the event registers. Software should not push a second time stamp event onto the event FIFO until the first time stamp value has been read from the event FIFO (there should be only one time stamp event in the event FIFO at any given time). This bit is write only and always reads zero.

Table 11-2288. Register Call Summary for CPTS_TS_PUSH

Gigabit Ethernet MAC (EMAC) Subsystem • Time Sync Events: [0]
EMAC Registers • NSS_0_CFG_CPTS Registers: [0]

11.13.5.1.4.5 CPTS_TS_LOAD_LOW_VAL Register (Offset = 10h) [reset = 0h]

CPTS_TS_LOAD_LOW_VAL is shown in Figure 11-1006 and described in Table 11-2290.

Time Stamp Load Value Register

Table 11-2289. CPTS_TS_LOAD_LOW_VAL Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D010h

Figure 11-1006. CPTS_TS_LOAD_LOW_VAL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS_LOAD_VAL																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2290. CPTS_TS_LOAD_LOW_VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TS_LOAD_VAL	R/W	0h	Time stamp load value– Writing the ts_load_en bit causes TS_LOAD_VAL[63:0] to be written into the time stamp. The time stamp value is read by initiating a time stamp push event, not by reading this register. When reading this register, the value read is not the time stamp, but is the value that was last written to this register.

Table 11-2291. Register Call Summary for CPTS_TS_LOAD_LOW_VAL

Gigabit Ethernet MAC (EMAC) Subsystem • 32-bit Time Stamp Value: [0] • 64-bit Time Stamp Value: [0]
EMAC Registers • NSS_0_CFG_CPTS Registers: [0] • CPTS_TS_LOAD_LOW_VAL Register (Offset = 10h) [reset = 0h]: [0]

11.13.5.1.4.6 CPTS_TS_LOAD_EN Register (Offset = 14h) [reset = 0h]

CPTS_TS_LOAD_EN is shown in Figure 11-1007 and described in Table 11-2293.

Time Stamp Load Enable Register

Table 11-2292. CPTS_TS_LOAD_EN Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D014h

Figure 11-1007. CPTS_TS_LOAD_EN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							TS_LOAD_EN
R-0h							W-0h

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 11-2293. CPTS_TS_LOAD_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Read as zero
0	TS_LOAD_EN	W	0h	Time stamp load enable – Writing a one to this bit enables the time stamp value to be written with the value in TS_LOAD_VAL[63:0]. This bit is write only and will be cleared by the hardware after one clock. The upper 32- bits of the timestamp are forced to zero in 32-bit mode.

Table 11-2294. Register Call Summary for CPTS_TS_LOAD_EN

Gigabit Ethernet MAC (EMAC) Subsystem • 32-bit Time Stamp Value: [0] • 64-bit Time Stamp Value: [0]
EMAC Registers • NSS_0_CFG_CPTS Registers: [0] • CPTS_TS_LOAD_EN Register (Offset = 14h) [reset = 0h]: [0]

11.13.5.1.4.7 CPTS_TS_COMP_LOW_VAL Register (Offset = 18h) [reset = 0h]

CPTS_TS_COMP_LOW_VAL in Figure 11-1008 and described in Table 11-2296.

Time Stamp Comparison Value Register

Table 11-2295. CPTS_TS_COMP_LOW_VAL Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D018h

Figure 11-1008. CPTS_TS_LOW_COMP_VAL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS_COMP_VAL																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2296. CPTS_TS_COMP_LOW_VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TS_COMP_VAL	R/W	0h	Time Stamp Comparison Low Value – Writing a non-zero value to the TS_Comp_Length[15:0] register causes a pulse of TS_Comp_Length RCLK periods on the TS_COMP output and a comparison event when the time_stamp counter value is equivalent to ts_comp_val.

Table 11-2297. Register Call Summary for CPTS_TS_COMP_LOW_VAL

Gigabit Ethernet MAC (EMAC) Subsystem • Timestamp Compare Event: [0][1] • Timestamp Compare Output: [0][1]
EMAC Registers • NSS_0_CFG_CPTS Registers: [0] • CPTS_TS_COMP_LOW_VAL Register (Offset = 18h) [reset = 0h]: [0]

11.13.5.1.4.8 CPTS_TS_COMP_LEN Register (Offset = 1Ch) [reset = 0h]

CPTS_TS_COMP_LEN is shown in Figure 11-1009 and described in Table 11-2299.

Time Stamp Comparison Length Register

Table 11-2298. CPTS_TS_COMP_LEN Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D01Ch

Figure 11-1009. CPTS_TS_COMP_LEN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								TS_COMP_LENGTH																							
R-0h								R/W-0h																							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2299. CPTS_TS_COMP_LEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Read as zero
23-0	TS_COMP_LENGTH	R/W	0h	Time stamp comparison length - Writing a non-zero value to this field enables the time stamp comparison event and output. This value should be zero when the TS_Comp_Low_Val and TS_Comp_High_Val registers are written.

Table 11-2300. Register Call Summary for CPTS_TS_COMP_LEN

Gigabit Ethernet MAC (EMAC) Subsystem • Timestamp Compare Event: [0][1] • Timestamp Compare Output: [0][1][2][3][4]
EMAC Registers • NSS_0_CFG_CPTS Registers: [0] • CPTS_TS_COMP_LEN Register (Offset = 1Ch) [reset = 0h]: [0]

11.13.5.1.4.9 CPTS_INTSTAT_RAW Register (Offset = 20h) [reset = 0h]

CPTS_INTSTAT_RAW is shown in Figure 11-1010 and described in Table 11-2302.

Interrupt Status Register Raw

Table 11-2301. CPTS_INTSTAT_RAW Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D020h

Figure 11-1010. CPTS_INTSTAT_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							TS_PEND_RAW
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2302. CPTS_INTSTAT_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	TS_PEND_RAW	R/W	0h	TS_PEND_RAW int read (before enable). Writable when int_test = 1 A one in this bit indicates that there are one or more events in the event FIFO.

Table 11-2303. Register Call Summary for CPTS_INTSTAT_RAW

Gigabit Ethernet MAC (EMAC) Subsystem • Statistics Interrupt: [0] • CPTS Interrupt Handling: [0][1]
EMAC Registers • CPTS_INTSTAT_RAW Register (Offset = 20h) [reset = 0h]: [0] • NSS_0_CFG_CPTS Registers: [0]

11.13.5.1.4.10 CPTS_INTSTAT_MASKED Register (Offset = 24h) [reset = 0h]

CPTS_INTSTAT_MASKED is shown in Figure 11-1011 and described in Table 11-2305.

Interrupt Status Register Masked

Table 11-2304. CPTS_INTSTAT_MASKED Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D024h

Figure 11-1011. CPTS_INTSTAT_MASKED Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							TS_PEND
R-0h							R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2305. CPTS_INTSTAT_MASKED Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	TS_PEND	R	0h	TS_PEND masked interrupt read (after enable)

Table 11-2306. Register Call Summary for CPTS_INTSTAT_MASKED

Gigabit Ethernet MAC (EMAC) Subsystem • Statistics Interrupt: [0]
EMAC Registers • NSS_0_CFG_CPTS Registers: [0] • CPTS_INTSTAT_MASKED Register (Offset = 24h) [reset = 0h]: [0]

11.13.5.1.4.11 CPTS_INT_ENABLE Register (Offset = 28h) [reset = 0h]

CPTS_INT_ENABLE is shown in Figure 11-1012 and described in Table 11-2308.

Interrupt Enable Register

Table 11-2307. CPTS_INT_ENABLE Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 F008h

Figure 11-1012. CPTS_INT_ENABLE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							TS_PEND_EN
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2308. CPTS_INT_ENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	RESERVED
0	TS_PEND_EN	R/W	0h	TS_PEND masked interrupt enable.

Table 11-2309. Register Call Summary for CPTS_INT_ENABLE

Gigabit Ethernet MAC (EMAC) Subsystem • Statistics Interrupt: [0] • CPTS Interrupt Handling: [0][1] • CPTS Initialization: [0]
EMAC Registers • NSS_0_CFG_CPTS Registers: [0] • CPTS_INT_ENABLE Register (Offset = 28h) [reset = 0h]: [0]

11.13.5.1.4.12 CPTS_TS_COMP_NUDGE Register (Offset = 2Ch) [reset = 0h]

CPTS_TS_COMP_NUDGE is shown in Figure 11-1012 and described in Table 11-2308.

Interrupt Enable Register

Table 11-2310. CPTS_TS_COMP_NUDGE Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 F02Ch

Figure 11-1013. CPTS_TS_COMP_NUDGE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								TS_COMP_LENGTH							
R-0h																								R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2311. CPTS_TS_COMP_NUDGE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Read as zero.
7-0	TS_COMP_NUDGE	R/W	0h	Timestamp Comparison Nudge Value. This two's complement number is added to the ts_comp_length[23:0] value to increase or decrease the TS_COMP length by the ts_comp_nudge amount. Only a single high or low time is adjusted and the ts_comp_nudge value is cleared to zero when the nudge has occurred.

Table 11-2312. Register Call Summary for CPTS_TS_COMP_NUDGE

Gigabit Ethernet MAC (EMAC) Subsystem • Timestamp Compare Output: [0][1]
EMAC Registers • NSS_0_CFG_CPTS Registers: [0] • CPTS_TS_COMP_NUDGE Register (Offset = 2Ch) [reset = 0h]: [0]

11.13.5.1.4.13 CPTS_EVENT_POP Register (Offset = 30h) [reset = 0h]

CPTS_EVENT_POP is shown in Figure 11-1014 and described in Table 11-2314.

Event Pop Register

Table 11-2313. CPTS_EVENT_POP Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D030h

Figure 11-1014. CPTS_EVENT_POP Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EVENT_POP
R-0h							W-0h

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 11-2314. CPTS_EVENT_POP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Read as zero.
0	EVENT_POP	W	0h	Event pop - When a logic high is written to this bit an event is popped off the event FIFO. The event FIFO pop occurs as part of the interrupt process after the event has been read from the Event_0-3 registers. Popping an event discards the event and causes the next event, if any, to be moved to the top of the FIFO ready to be read by software on interrupt.

Table 11-2315. Register Call Summary for CPTS_EVENT_POP

Gigabit Ethernet MAC (EMAC) Subsystem • CPTS Interrupt Handling: [0][1]
EMAC Registers • NSS_0_CFG_CPTS Registers: [0] • CPTS_EVENT_POP Register (Offset = 30h) [reset = 0h]: [0]

11.13.5.1.4.14 CPTS_EVENT_0 Register (Offset = 34h) [reset = 0h]

event_low is shown in Figure 11-1015 and described in Table 11-2317.

Event Low Register

Table 11-2316. CPTS_EVENT_0 Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D034h

Figure 11-1015. CPTS_EVENT_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIME_STAMP																															
R-																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2317. CPTS_EVENT_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TIME_STAMP	R	-	Time Stamp – The timestamp is valid for transmit, receive, and time stamp push event types. The timestamp value is not valid for counter roll event types.

Table 11-2318. Register Call Summary for CPTS_EVENT_0

Gigabit Ethernet MAC (EMAC) Subsystem • Time Sync Events: [0][1] • CPTS Interrupt Handling: [0][1]
EMAC Registers • NSS_0_CFG_CPTS Registers: [0]

11.13.5.1.4.15 CPTS_EVENT_1 Register (Offset = 38h) [reset = 0h]

CPTS_EVENT_1 is shown in Figure 11-1016 and described in Table 11-2320.

Event Middle Register

Table 11-2319. CPTS_EVENT_1 Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D038h

Figure 11-1016. CPTS_EVENT_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED			PORT_NUMBER					EVENT_TYPE				MESSAGE_TYPE			
R-			R-					R-				R-			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEQUENCE_ID															
R-															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2320. CPTS_EVENT_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	-	Reserved
28-24	PORT_NUMBER	R	-	Port number - indicates the port number (encoded) of an Ethernet event or the encoded hardware timestamp number.
23-20	EVENT_TYPE	R	-	Event type • 0000 – Time Stamp Push Event • 0001 – Time Stamp Rollover Event • 0010 – Time Stamp Half Rollover Event • 0011 – Hardware Time Stamp Push Event • 0100 – Ethernet Receive Event • 0101 – Ethernet Transmit Event • 0110 – Time Stamp Compare Event • 0111 – Host Transmit Event • 1000-1111 - Reserved
19-16	MESSAGE_TYPE	R	-	Message type – The message type value that was contained in an Ethernet transmit or receive time sync packet. This field is valid only for Ethernet transmit or receive events.
15-0	SEQUENCE_ID	R	-	Sequence ID – The 16-bit sequence id is the value that was contained in an Ethernet transmit or receive time sync packet. This field is valid only for Ethernet transmit or receive events.

Table 11-2321. Register Call Summary for CPTS_EVENT_1

Gigabit Ethernet MAC (EMAC) Subsystem • Time Sync Events: [0][1]
EMAC Registers • NSS_0_CFG_CPTS Registers: [0] • CPTS_EVENT_1 Register (Offset = 38h) [reset = 0h]: [0]

11.13.5.1.4.16 CPTS_EVENT_2 Register (Offset = 3Ch) [reset = 0h]

CPTS_EVENT_2 is shown in Figure 11-1017 and described in Table 11-2323.

Event High Register

Table 11-2322. CPTS_EVENT_2 Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D03Ch

Figure 11-1017. CPTS_EVENT_HIGH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DOMAIN							
R-																								R-							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2323. CPTS_EVENT_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	-	RESERVED
7-0	DOMAIN	R	-	Domain – The 8-bit domain is the value that was contained in an Ethernet transmit or receive time sync packet. This field is valid only for Ethernet transmit or receive events.

Table 11-2324. Register Call Summary for CPTS_EVENT_2

EMAC Registers

- NSS_0_CFG_CPTS Registers: [0]
- CPTS_EVENT_2 Register (Offset = 3Ch) [reset = 0h]: [0]

11.13.5.1.4.17 CPTS_EVENT_3 Register (Offset = 40h) [reset = 0h]

CPTS_EVENT_3 is shown in Figure 11-1017 and described in Table 11-2323.

Event High Register

Table 11-2325. CPTS_EVENT_3 Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D040h

Figure 11-1018. CPTS_EVENT_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIME_STAMP																															
R-																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2326. CPTS_EVENT_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TIME_STAMP	R	-	Time Stamp – The timestamp upper 32-bits are valid for transmit, receive, and time stamp push event types. This value is zero in 32-bit mode.

Table 11-2327. Register Call Summary for CPTS_EVENT_3

Gigabit Ethernet MAC (EMAC) Subsystem • Time Sync Events: [0][1] • CPTS Interrupt Handling: [0][1]
EMAC Registers • NSS_0_CFG_CPTS Registers: [0] • CPTS_EVENT_3 Register (Offset = 40h) [reset = 0h]: [0]

11.13.5.1.4.18 CPTS_TS_LOAD_HIGH_VAL Register (Offset = 44h) [reset = 0h]

CPTS_TS_LOAD_HIGH_VAL is shown in Figure 11-1017 and described in Table 11-2323.

Table 11-2328. CPTS_TS_LOAD_HIGH_VAL Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D044h

Figure 11-1019. CPTS_TS_LOAD_HIGH_VAL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS_LOAD_VAL																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-2329. CPTS_TS_LOAD_HIGH_VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TS_LOAD_VAL	RW	0h	Time Stamp Load high Value – Writing the ts_load_en bit causes the value contained in this register (and the CPTS_TS_LOAD_VAL[63:0]) to be written into the time stamp. The time stamp value is read by initiating a time stamp push event, not by reading this register. When reading this register, the value read is not the time stamp, but is the value that was last written to this register. This value is unused in 32-bit mode

Table 11-2330. Register Call Summary for CPTS_TS_LOAD_HIGH_VAL

Gigabit Ethernet MAC (EMAC) Subsystem • 64-bit Time Stamp Value: [0]
EMAC Registers • CPTS_TS_LOAD_HIGH_VAL Register (Offset = 44h) [reset = 0h]: [0] • NSS_0_CFG_CPTS Registers: [0]

11.13.5.1.4.19 CPTS_TS_COMP_HIGH_VAL Register (Offset = 48h) [reset = 0h]

CPTS_TS_COMP_HIGH_VAL is shown in Figure 11-1017 and described in Table 11-2323.

Table 11-2331. CPTS_TS_COMP_HIGH_VAL Instances

Instance	Physical Address
NSS_0_CFG_CPTS	0423 D044h

Figure 11-1020. CPTS_TS_COMP_HIGH_VAL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS_COMP_VAL																															
RW-0h																															

LEGEND: RW = Read/Write; *-n* = value after reset

Table 11-2332. CPTS_TS_COMP_VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TS_COMP_VAL	RW	0h	Time Stamp Comparison High Value – Writing a non-zero value to the TS_Comp_Length[15:0] register causes a pulse of TS_Comp_Length RCLK periods on the TS_COMP output and a comparison event when the time_stamp counter value is equivalent to ts_comp_val[63:0]. This value is unused in 32-bit mode. The upper 32-bits in this register should be written before the lower 32-bits in the TS_Comp_Low_Val register.

Table 11-2333. Register Call Summary for CPTS_TS_COMP_HIGH_VAL

Gigabit Ethernet MAC (EMAC) Subsystem • Timestamp Compare Event: [0][1]
EMAC Registers • NSS_0_CFG_CPTS Registers: [0] • CPTS_TS_COMP_HIGH_VAL Register (Offset = 48h) [reset = 0h]: [0]

11.13.5.1.5 NSS_0_CFG_MDIO Registers

Table 11-2335 lists the memory-mapped registers for the NSS_0_CFG_MDIO. All register offset addresses not listed in Table 11-2335 should be considered as reserved locations and the register contents should not be modified.

Table 11-2334. NSS_0_CFG_MDIO Instances

Instance	Base Address
NSS_0_CFG_MDIO	0420 0F00h

Table 11-2335. NSS_0_CFG_MDIO Registers

Offset	Acronym	Register Name	NSS_0_CFG_MDIO Physical Address	Section
0h	MDIO_VERSION	MDIO Revision	0420 0F00h	Section 11.13.5.1.5.1
4h	MDIO_CONTROL	MDIO Control register	0420 0F04h	Section 11.13.5.1.5.2
8h	MDIO_ALIVE	PHY Alive Status Register	0420 0F08h	Section 11.13.5.1.5.3
Ch	MDIO_LINK	PHY Link Status	0420 0F0Ch	Section 11.13.5.1.5.4
10h	MDIO_LINK_INT_RAW	MDIO Link Status Change Interrupt Register, raw value	0420 0F10h	Section 11.13.5.1.5.5
14h	MDIO_LINK_INT_MASKED	MDIO Link Status Change Interrupt Register	0420 0F14h	Section 11.13.5.1.5.6
20h	MDIO_USER_INT_RAW	MDIO User Command Complete Interrupt, raw value	0420 0F20h	Section 11.13.5.1.5.7
24h	MDIO_USER_INT_MASKED	MDIO User Command Complete Interrupt	0420 0F24h	Section 11.13.5.1.5.8
28h	MDIO_USER_INT_MASK_SET	MDIO User Command Complete Interrupt Mask Set	0420 0F28h	Section 11.13.5.1.5.9
2Ch	MDIO_USER_INT_MASK_CLEAR	MDIO User Command Complete Interrupt Mask Clear	0420 0F2Ch	Section 11.13.5.1.5.10
80h	MDIO_USER_ACCESS_0	MDIO User Access	0420 0F80h	Section 11.13.5.1.5.11
84h	MDIO_USER_PHY_SEL_0	MDIO User PHY Select	0420 0F84h	Section 11.13.5.1.5.12
84h	MDIO_USER_ACCESS_1	MDIO User Access	0420 0F84h	Section 11.13.5.1.5.11
88h	MDIO_USER_PHY_SEL_1	MDIO User PHY Select	0420 0F88h	Section 11.13.5.1.5.12

11.13.5.1.5.1 MDIO_VERSION Register (Offset = 0h) [reset = 00070105h]

MDIO_VERSION is shown in Figure 11-1021 and described in Table 11-2337.

Table 11-2336. MDIO_VERSION Instances

Instance	Physical Address
NSS_0_CFG_MDIO	0420 0F00h

Figure 11-1021. MDIO_VERSION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 00070105h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2337. MDIO_VERSION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	00070105h	TI internal data. Identifies revision of peripheral.

Table 11-2338. Register Call Summary for MDIO_VERSION

EMAC Registers
- NSS_0_CFG_MDIO Registers: [0]
- MDIO_VERSION Register (Offset = 0h) [reset = 00070105h]: [0]

11.13.5.1.5.2 MDIO_CONTROL Register (Offset = 4h) [reset = 810000FFh]

MDIO_CONTROL is shown in Figure 11-1022 and described in Table 11-2340.

Table 11-2339. MDIO_CONTROL Instances

Instance	Physical Address
NSS_0_CFG_MDIO	0420 0F04h

Figure 11-1022. MDIO_CONTROL Register

31	30	29	28	27	26	25	24
IDLE	ENABLE	RESERVED	HIGHEST_USER_CHANNEL				
R-1h	R/W-0h	R-0h	R-1h				
23	22	21	20	19	18	17	16
RESERVED			PREAMBLE	FAULT	FAULT_DETECT_ENABLE	INT_TEST_ENABLE	RESERVED
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h
15	14	13	12	11	10	9	8
CLKDIV							
R/W-FFh							
7	6	5	4	3	2	1	0
CLKDIV							
R/W-FFh							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2340. MDIO_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IDLE	R	1h	MDIO state machine idle indicator. 0: State machine is running. 1: State machine is in idle state.
30	ENABLE	R/W	0h	Enable control. If the MDIO state machine is active at the time it is disabled, it will complete the current operation before halting and setting the IDLE bit. If using byte access, the ENABLE bit has to be the last bit written in this register. 0: Disables the MDIO state machine. 1: Enable the MDIO state machine.
29	RESERVED	R	0h	Reserved
28-24	HIGHEST_USER_CHANNEL	R	1h	Highest user channel. This field specifies the highest user access channel that is available in the module. E.g. 0 = channel 0 only. 1 = channels 0 and 1, and so on. The device itself may not pin out all channels. See Environment section for details.
23-21	RESERVED	R	0h	Reserved
20	PREAMBLE	R/W	0h	Preamble disable. 0: Standard MDIO preamble is used. 1: Disables this device from sending MDIO frame preambles.
19	FAULT	R/W	0h	Fault indicator. This bit is set to 1 if the MDIO pins fail to read back what the device is driving onto them. This indicates a physical layer fault and the module state machine is reset. Writing a 1 to it clears this bit. 0: No failure. 1: Physical layer fault; the MDIO state machine is reset.
18	FAULT_DETECT_ENABLE	R/W	0h	Fault detect enable. This bit has to be set to 1 to enable the physical layer fault detection. 0: Disables the physical layer fault detection. 1: Enables the physical layer fault detection.

Table 11-2340. MDIO_CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	INT_TEST_ENABLE	R/W	0h	Interrupt test enable. This bit can be set to 1 to enable the host to set the USERINT and LINKINT bits for test purposes. 0: Interrupt bits are not set. 1: Enables the host to set the USERINT and LINKINT bits for test purposes.
16	RESERVED	R	0h	Reserved
15-0	CLKDIV	R/W	FFh	Clock divider. This field specifies the division ratio between ICLK and the frequency of MDCLK. MDCLK is disabled when CLKDIV is set to 0. MDCLK frequency = ICLK frequency/(CLKDIV+1).

Table 11-2341. Register Call Summary for MDIO_CONTROL

Gigabit Ethernet MAC (EMAC) Subsystem • MDIO Functional Description: [0] • Initializing the MDIO Module: [0][1] • Interface Clocking: [0]
EMAC Registers • NSS_0_CFG_MDIO Registers: [0] • MDIO_CONTROL Register (Offset = 4h) [reset = 810000FFh]: [0] • MDIO_USER_INT_MASKED Register (Offset = 24h) [reset = 0h]: [0] • MDIO_LINK_INT_MASKED Register (Offset = 14h) [reset = 0h]: [0] • MDIO_USER_INT_RAW Register (Offset = 20h) [reset = 0h]: [0]

11.13.5.1.5.3 MDIO_ALIVE Register (Offset = 8h) [reset = 0h]

MDIO_ALIVE is shown in Figure 11-1023 and described in Table 11-2343.

Table 11-2342. MDIO_ALIVE Instances

Instance	Physical Address
NSS_0_CFG_MDIO	0420 0F08h

Figure 11-1023. MDIO_ALIVE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALIVE																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2343. MDIO_ALIVE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ALIVE	R/W	0h	MDIO alive. Bit is set if the most recent access to the PHY was acknowledged by the PHY, the bit is reset if the PHY fails to acknowledge the access. Both the user and polling accesses to a PHY will cause the corresponding ALIVE bit to be updated. The ALIVE bits are only meant to be used to give an indication of the presence or not of a PHY with the corresponding address. Writing a 1 to any bit will clear it, writing a 0 has no effect.

Table 11-2344. Register Call Summary for MDIO_ALIVE

Gigabit Ethernet MAC (EMAC) Subsystem • MDIO Functional Description: [0][1] • Initializing the MDIO Module: [0]
EMAC Registers • NSS_0_CFG_MDIO Registers: [0] • MDIO_ALIVE Register (Offset = 8h) [reset = 0h]: [0]

11.13.5.1.5.4 MDIO_LINK Register (Offset = Ch) [reset = 0h]

MDIO_LINK is shown in Figure 11-1024 and described in Table 11-2346.

Table 11-2345. MDIO_LINK Instances

Instance	Physical Address
NSS_0_CFG_MDIO	0420 0F0Ch

Figure 11-1024. MDIO_LINK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LINK																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2346. MDIO_LINK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LINK	R	0h	MDIO link state. This register is updated after a read of the Generic Status Register of a PHY. The bit is set if the PHY has link and the PHY acknowledges the read transaction. The bit is reset if the PHY indicates it does not have link or fails to acknowledge the read transaction. Writes to the register have no effect.

Table 11-2347. Register Call Summary for MDIO_LINK

Gigabit Ethernet MAC (EMAC) Subsystem • MDIO Functional Description: [0] • Initializing the MDIO Module: [0]
EMAC Registers • NSS_0_CFG_MDIO Registers: [0] • MDIO_LINK Register (Offset = Ch) [reset = 0h]: [0]

11.13.5.1.5.5 MDIO_LINK_INT_RAW Register (Offset = 10h) [reset = 0h]

MDIO_LINK_INT_RAW is shown in Figure 11-1025 and described in Table 11-2349.

Table 11-2348. MDIO_LINK_INT_RAW Instances

Instance	Physical Address
NSS_0_CFG_MDIO	0420 0F10h

Figure 11-1025. MDIO_LINK_INT_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						LINKINTRAW	
R-0h						R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2349. MDIO_LINK_INT_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	LINKINTRAW	R/W	0h	MDIO link change event raw value. Channels 1:0

Table 11-2350. Register Call Summary for MDIO_LINK_INT_RAW

Gigabit Ethernet MAC (EMAC) Subsystem • MDIO Functional Description: [0]
EMAC Registers • NSS_0_CFG_MDIO Registers: [0] • MDIO_LINK_INT_RAW Register (Offset = 10h) [reset = 0h]: [0]

11.13.5.1.5.6 MDIO_LINK_INT_MASKED Register (Offset = 14h) [reset = 0h]

MDIO_LINK_INT_MASKED is shown in Figure 11-1026 and described in Table 11-2352.

Table 11-2351. MDIO_LINK_INT_MASKED Instances

Instance	Physical Address
NSS_0_CFG_MDIO	0420 0F14h

Figure 11-1026. MDIO_LINK_INT_MASKED Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						LINKINTMASKED	
R-0h						R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2352. MDIO_LINK_INT_MASKED Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	LINKINTMASKED	R/W	0h	MDIO link change interrupt, masked value. When asserted 1, a bit indicates that there was an MDIO link change event (i.e. change in the MDIO Link register) corresponding to the PHY address in the MDIO_USERPHYSEL register and the LINKINTENB bit was set. LINKINTMASKED[0] corresponds to MDIO_USERPHYSEL0. Writing a 1 will clear the interrupt and writing 0 has no effect. If the INTTESTENB bit in the MDIO_CONTROL register is set, the host may set the LINKINT bits to a 1. This mode may be used for test purposes.

Table 11-2353. Register Call Summary for MDIO_LINK_INT_MASKED

Gigabit Ethernet MAC (EMAC) Subsystem • MDIO Functional Description: [0] • MDIO Interrupts: [0]
EMAC Registers • NSS_0_CFG_MDIO Registers: [0] • MDIO_LINK_INT_MASKED Register (Offset = 14h) [reset = 0h]: [0]

11.13.5.1.5.7 MDIO_USER_INT_RAW Register (Offset = 20h) [reset = 0h]

MDIO_USER_INT_RAW is shown in Figure 11-1027 and described in Table 11-2355.

Table 11-2354. MDIO_USER_INT_RAW Instances

Instance	Physical Address
NSS_0_CFG_MDIO	0420 0F20h

Figure 11-1027. MDIO_USER_INT_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						USERINTRAW	
R-0h						R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2355. MDIO_USER_INT_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	USERINTRAW	R/W	0h	Raw value of MDIO user command complete event for the MDIO_USERACCESS register. When asserted 1, a bit indicates that the previously scheduled PHY read or write command using that MDIO_USERACCESS register has completed. Writing a 1 will clear the event and writing 0 has no effect. If the INTTESTENB bit in the MDIO_CONTROL register is set, the host may set the USERINTRAW bits to a 1. This mode may be used for test purposes.

Table 11-2356. Register Call Summary for MDIO_USER_INT_RAW

Gigabit Ethernet MAC (EMAC) Subsystem • Reading Data From a PHY Register: [0] • Writing Data To a PHY Register: [0] • MDIO Functional Description: [0]
EMAC Registers • NSS_0_CFG_MDIO Registers: [0] • MDIO_USER_INT_RAW Register (Offset = 20h) [reset = 0h]: [0]

11.13.5.1.5.8 MDIO_USER_INT_MASKED Register (Offset = 24h) [reset = 0h]

MDIO_USER_INT_MASKED is shown in Figure 11-1028 and described in Table 11-2358.

Table 11-2357. MDIO_USER_INT_MASKED Instances

Instance	Physical Address
NSS_0_CFG_MDIO	0420 0F24h

Figure 11-1028. MDIO_USER_INT_MASKED Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						USERINTMASKED	
R-0h						R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2358. MDIO_USER_INT_MASKED Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	USERINTMASKED	R/W	0h	Masked value of MDIO user command complete interrupt for the MDIO_USERACCESS register. When asserted 1, a bit indicates that the previously scheduled PHY read or write command using MDIO_USERACCESS register has completed and the corresponding USERINTMASKSET bit is set to 1. Writing a 1 will clear the interrupt and writing 0 has no effect. If the INTTESTENB bit in the MDIO_CONTROL register is set, the host may set the USERINTMASKED bits to a 1. This mode may be used for test purposes.

Table 11-2359. Register Call Summary for MDIO_USER_INT_MASKED

Gigabit Ethernet MAC (EMAC) Subsystem • Reading Data From a PHY Register: [0] • Writing Data To a PHY Register: [0] • MDIO Functional Description: [0] • MDIO Interrupts: [0]
EMAC Registers • NSS_0_CFG_MDIO Registers: [0] • MDIO_USER_INT_MASKED Register (Offset = 24h) [reset = 0h]: [0]

11.13.5.1.5.9 MDIO_USER_INT_MASK_SET Register (Offset = 28h) [reset = 0h]

MDIO_USER_INT_MASK_SET is shown in Figure 11-1029 and described in Table 11-2361.

Table 11-2360. MDIO_USER_INT_MASK_SET Instances

Instance	Physical Address
NSS_0_CFG_MDIO	0420 0F28h

Figure 11-1029. MDIO_USER_INT_MASK_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						USERINTMASKSET	
R-0h						R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2361. MDIO_USER_INT_MASK_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	USERINTMASKSET	R/W	0h	MDIO user interrupt mask set for USERINTMASKED. Writing a bit to 1 will enable MDIO user command complete interrupts for MDIO_USERACCESS register. MDIO user interrupt for MDIO_USERACCESS register is disabled if the bit is 0. Writing a 0 to this register has no effect.

Table 11-2362. Register Call Summary for MDIO_USER_INT_MASK_SET

Gigabit Ethernet MAC (EMAC) Subsystem • Reading Data From a PHY Register: [0] • Writing Data To a PHY Register: [0] • MDIO Functional Description: [0] • Initializing the MDIO Module: [0] • MDIO Interrupts: [0]
EMAC Registers • NSS_0_CFG_MDIO Registers: [0] • MDIO_USER_INT_MASK_SET Register (Offset = 28h) [reset = 0h]: [0]

11.13.5.1.5.10 MDIO_USER_INT_MASK_CLEAR Register (Offset = 2Ch) [reset = 0h]

MDIO_USER_INT_MASK_CLEAR is shown in Figure 11-1030 and described in Table 11-2364.

Table 11-2363. MDIO_USER_INT_MASK_CLEAR Instances

Instance	Physical Address
NSS_0_CFG_MDIO	0420 0F2Ch

Figure 11-1030. MDIO_USER_INT_MASK_CLEAR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						USERINTMASKCLR	
R-0h						R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2364. MDIO_USER_INT_MASK_CLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	USERINTMASKCLR	R/W	0h	MDIO user command complete interrupt mask clear for USERINTMASKED. Writing a bit to 1 will disable further user command complete interrupts for MDIO_USERACCESS register. Writing a 0 to this register has no effect.

Table 11-2365. Register Call Summary for MDIO_USER_INT_MASK_CLEAR

EMAC Registers
- NSS_0_CFG_MDIO Registers: [0]
- MDIO_USER_INT_MASK_CLEAR Register (Offset = 2Ch) [reset = 0h]: [0]

11.13.5.1.5.11 MDIO_USER_ACCESS_0 to MDIO_USER_ACCESS_1 Register (Offset = 80h + [i * 4h], where i = 0 to 1) [reset = 0h]

MDIO_USER_ACCESS_0 to MDIO_USER_ACCESS_1 is shown in Figure 11-1031 and described in Table 11-2367.

Table 11-2366. MDIO_USER_ACCESS_0 to MDIO_USER_ACCESS_1 Instances

Instance	Physical Address
NSS_0_CFG_MDIO	0420 0F80h + [i * 4h], where i = 0 to 1

Figure 11-1031. MDIO_USER_ACCESS_0 to MDIO_USER_ACCESS_1 Register

31	30	29	28	27	26	25	24
GO	WRITE	ACK	RESERVED			REGADR	
R/W-0h	R/W-0h	R/W-0h	R-0h			R/W-0h	
23	22	21	20	19	18	17	16
REGADR			PHYADR				
R/W-0h			R/W-0h				
15	14	13	12	11	10	9	8
DATA							
R/W-0h							
7	6	5	4	3	2	1	0
DATA							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2367. MDIO_USER_ACCESS_0 to MDIO_USER_ACCESS_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GO	R/W	0h	Go. Writing a 1 to this bit causes the MDIO state machine to perform an MDIO access when it is convenient for it to do so, this is not an instantaneous process. Writing a 0 to this bit has no effect. This bit is writeable only if the MDIO state machine is enabled. This bit will self clear when the requested access has been completed. Any writes to the MDIO_USERACCESS register are blocked when the GO bit is 1. If byte access is being used, the GO bit should be written last.
30	WRITE	R/W	0h	Write enable. Setting this bit to a 1 causes the MDIO transaction to be a register write, otherwise it is a register read.
29	ACK	R/W	0h	Acknowledge. This bit is set if the PHY acknowledged the read transaction.
28-26	RESERVED	R	0h	Reserved
25-21	REGADR	R/W	0h	Register address. Specifies the PHY register to be accessed for this transaction.
20-16	PHYADR	R/W	0h	PHY address. Specifies the PHY to be accesses for this transaction.
15-0	DATA	R/W	0h	User data. The data value read from or to be written to the specified PHY register.

Table 11-2368. Register Call Summary for MDIO_USER_ACCESS_0

Gigabit Ethernet MAC (EMAC) Subsystem

- Reading Data From a PHY Register: [0][1][2][3][4][5]
- Writing Data To a PHY Register: [0][1][2][3][4]
- MDIO Functional Description: [0][1][2][3][4]
- Initializing the MDIO Module: [0][1][2]
- MDIO Interrupts: [0]

Table 11-2368. Register Call Summary for MDIO_USER_ACCESS_0 (continued)

EMAC Registers
• NSS_0_CFG_MDIO Registers: [0] • MDIO_USER_ACCESS_0 to MDIO_USER_ACCESS_1 Register (Offset = 80h + [i * 4h], where i = 0 to 1) [reset = 0h]: [0]

11.13.5.1.5.12 MDIO_USER_PHY_SEL_0 to MDIO_USER_PHY_SEL_1 Register (Offset = 84h + [i * 4h], where i = 0 to 1) [reset = 0h]

MDIO_USER_PHY_SEL_0 to MDIO_USER_PHY_SEL_1 is shown in Figure 11-1032 and described in Table 11-2370.

Table 11-2369. MDIO_USER_PHY_SEL_0 to MDIO_USER_PHY_SEL_1 Instances

Instance	Physical Address
NSS_0_CFG_MDIO	0420 0F84h + [i * 4h], where i = 0 to 1

Figure 11-1032. MDIO_USER_PHY_SEL_0 to MDIO_USER_PHY_SEL_1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
LINKSEL	LINKINT_ENABLE	RESERVED	PHYADR_MON				
R/W-0h	R/W-0h	R-0h	R/W-0h				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2370. MDIO_USER_PHY_SEL_0 to MDIO_USER_PHY_SEL_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	LINKSEL	R/W	0h	Link status determination select. 0: link status is determined by the MDIO state machine 1: link status using the MLINK pin (NOT PINNED OUT. DO NOT USE!)
6	LINKINT_ENABLE	R/W	0h	Link change interrupt enable. Set to 1 to enable link change status interrupts for PHY address specified in PHYADDRMON. Link change interrupts are disabled if this bit is set to 0. 0: Link change interrupts are disabled. 1: Link change status interrupts for PHY address specified in PHYADDRMON bits are enabled.
5	RESERVED	R	0h	Reserved
4-0	PHYADR_MON	R/W	0h	PHY address whose link status is monitored

Table 11-2371. Register Call Summary for MDIO_USER_PHY_SEL_0

Gigabit Ethernet MAC (EMAC) Subsystem • MDIO Functional Description: [0][1] • Initializing the MDIO Module: [0] • MDIO Interrupts: [0]
EMAC Registers • NSS_0_CFG_MDIO Registers: [0] • MDIO_USER_PHY_SEL_0 to MDIO_USER_PHY_SEL_1 Register (Offset = 84h + [i * 4h], where i = 0 to 1) [reset = 0h]: [0]

11.13.5.1.6 NSS_0_CFG_ECC Registers

lists the memory-mapped registers for the NSS_0_CFG_ECC. All register offset addresses not listed in should be considered as reserved locations and the register contents should not be modified.

Table 11-2372. NSS_0_CFG_ECC Instances

Instance	Base Address
NSS_0_CFG_NAVSS_ECC	0400 0800h
NSS_0_CFG_EMAC_ECC	0423 F000h

Table 11-2373. NSS_0_CFG_ECC Registers

Offset	Acronym	Register Name	NSS_0_CFG_NAVSS_ECC Physical Address	NSS_0_CFG_EMAC_ECC Physical Address	Section
0h	ECC_REVISION	The Revision Register contains the ID and revision information.	0400 0800h	0423 F000h	Section 11.13.5.1.6.1
8h	ECC_VECTOR	ECC RAM ID to select the ECC RAM to control or read status.	0400 0808h	0423 F008h	Section 11.13.5.1.6.2
Ch	ECC_MISC_STATUS	Miscellaneous status register.	0400 080Ch	0423 F00Ch	Section 11.13.5.1.6.3
10h	ECC_WRAPPER_REVISION	The Revision Register contains the ID and revision information for the ECC wrapper.	0400 0810h	0423 F010h	Section 11.13.5.1.6.4
14h	ECC_CONTROL	The Control Register controls the ECC control bits for the selected ECC RAM.	0400 0814h	0423 F014h	Section 11.13.5.1.6.5
18h	ECC_ERROR_CONTROL1	This register contains ECC error control bits for the selected ECC RAM.	0400 0818h	0423 F018h	Section 11.13.5.1.6.6
1Ch	ECC_ERROR_CONTROL2	This register contains ECC error control bits for the selected ECC RAM.	0400 081Ch	0423 F01Ch	Section 11.13.5.1.6.7
20h	ECC_ERROR_STATUS1	This register contains ECC status bits for the selected ECC RAM.	0400 0820h	0423 F020h	Section 11.13.5.1.6.8
24h	ECC_ERROR_STATUS2	This register contains ECC status bits for the selected ECC RAM.	0400 0824h	0423 F024h	Section 11.13.5.1.6.9
3Ch	ECC_EOI	This is the ECC_EOI register for the interrupt to the host.	0400 083Ch	0423 F03Ch	Section 11.13.5.1.6.10
40h to 7Ch	ECC_INT_STATUS_0 to ECC_INT_STATUS_15	These are the raw level interrupt status bits where each bit corresponds to the pending status from an ECC RAM.	0400 0840h to 0400 087Ch	0423 F040h to 0423 F07Ch	Section 11.13.5.1.6.11
80h to BCh	ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15	These are interrupt enables associated with the interrupt from each of the ECC RAMs. Writing a 1 to a bit position in the register enables the interrupt from the associated ECC RAM.	0400 0880h to 0400 08BCh	0423 F080h to 0423 F0BCh	Section 11.13.5.1.6.12
C0h to FCh	ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15	These are interrupt enable clear bits associated with the interrupt from each of the ECC RAMs. Writing a 1 to a bit position in the register disables the interrupt from the associated ECC RAM.	0400 08C0h to 0400 08FCh	0423 F0C0h to 0423 F0FCh	Section 11.13.5.1.6.13

11.13.5.1.6.1 ECC_REVISION Register (Offset = 0h) [reset = 4E100001h]

ECC_REVISION is shown in and described in Table 11-2375.

The Revision Register contains the ID and revision information.

Table 11-2374. ECC_REVISION Instances

Instance	Physical Address
NSS_0_CFG_NAVSS_ECC	0400 0800h
NSS_0_CFG_EMAC_ECC	0423 F000h

Figure 11-1033. ECC_REVISION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 4E100001h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2375. ECC_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4E100001h	TI internal data. Identifies revision of peripheral.

Table 11-2376. Register Call Summary for ECC_REVISION

EMAC Registers • ECC_REVISION Register (Offset = 0h) [reset = 4E100001h]: [0] • NSS_0_CFG_ECC Registers: [0]
Memory Error Detection and Correction • ECC Wrapper and ECC Aggregator: [0]

11.13.5.1.6.2 ECC_VECTOR Register (Offset = 8h) [reset = 0h]

ECC_VECTOR is shown in Figure 11-1034 and described in Table 11-2378.

ECC RAM ID to select the ECC RAM to control or read status.

Table 11-2377. ECC_VECTOR Instances

Instance	Physical Address
NSS_0_CFG_NAVSS_ECC	0400 0808h
NSS_0_CFG_EMAC_ECC	0423 F008h

Figure 11-1034. ECC_VECTOR Register

31	30	29	28	27	26	25	24
RESERVED							READ_DONE
R-							R-0h
23	22	21	20	19	18	17	16
READ_ADDRESS							
R/W-0h							
15	14	13	12	11	10	9	8
TRIGGER_READ	RESERVED					RAM_ID	
R/W-0h	R-					R/W-0h	
7	6	5	4	3	2	1	0
RAM_ID							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2378. ECC_VECTOR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R		Reserved
24	READ_DONE	R	0h	Status indicating that the serial VBUS read is complete
23-16	READ_ADDRESS	R/W	0h	Read address. Can be any of the registers 0x10 - 0x24
15	TRIGGER_READ	R/W	0h	Trigger a read operation to the specified read address that requires a serial VBUS access
14-11	RESERVED	R		Reserved
10-0	RAM_ID	R/W	0h	ECC RAM ID to select which ECC RAM to control or read status from

Table 11-2379. Register Call Summary for ECC_VECTOR

EMAC Registers • ECC_VECTOR Register (Offset = 8h) [reset = 0h]: [0] • NSS_0_CFG_ECC Registers: [0]
Networking Subsystem (NSS) • Memory Error Detection and Correction: [0]
Memory Error Detection and Correction • ECC Interrupts: [0][1][2] • Reads to ECC Control and Status Registers: [0][1][2] • ECC Wrapper and ECC Aggregator: [0][1][2][3]

11.13.5.1.6.3 ECC_MISC_STATUS Register (Offset = Ch) [reset = 0h]

ECC_MISC_STATUS is shown in Figure 11-1035 and described in Table 11-2381.

Miscellaneous status register.

Table 11-2380. ECC_MISC_STATUS Instances

Instance	Physical Address
NSS_0_CFG_NAVSS_ECC	0400 080Ch
NSS_0_CFG_EMAC_ECC	0423 F00Ch

Figure 11-1035. ECC_MISC_STATUS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					NUM_RAMS										
R-0h																					R/W-										

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2381. ECC_MISC_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Reserved
10-0	NUM_RAMS	R/W		Number of ECC RAMs serviced by the aggregator

Table 11-2382. Register Call Summary for ECC_MISC_STATUS

EMAC Registers • NSS_0_CFG_ECC Registers: [0] • ECC_MISC_STATUS Register (Offset = Ch) [reset = 0h]: [0]
Memory Error Detection and Correction • ECC Wrapper and ECC Aggregator: [0]

11.13.5.1.6.4 ECC_WRAPPER_REVISION Register (Offset = 10h) [reset = 4E100001 h]

ECC_WRAPPER_REVISION is shown in Figure 6-57 and described in Table 11-2384.

The Revision Register contains the ID and revision information for the ECC wrapper.

Table 11-2383. ECC_WRAPPER_REVISION Instances

Instance	Physical Address
NSS_0_CFG_NAVSS_ECC	0400 0810h
NSS_0_CFG_EMAC_ECC	0423 F010h

Figure 11-1036. ECC_WRAPPER_REVISION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 4E100001h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2384. ECC_WRAPPER_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4E100001h	TI internal data. Identifies revision of peripheral.

Table 11-2385. Register Call Summary for ECC_WRAPPER_REVISION

EMAC Registers • ECC_WRAPPER_REVISION Register (Offset = 10h) [reset = 4E100001 h]: [0] • NSS_0_CFG_ECC Registers: [0]
Memory Error Detection and Correction • Reads to ECC Control and Status Registers: [0]

11.13.5.1.6.5 ECC_CONTROL Register (Offset = 14h) [reset = 7h]

ECC_CONTROL is shown in Figure 11-1037 and described in Table 11-2387.

The Control Register controls the ECC control bits for the selected ECC RAM.

Table 11-2386. ECC_CONTROL Instances

Instance	Physical Address
NSS_0_CFG_NAVSS_ECC	0400 0814h
NSS_0_CFG_EMAC_ECC	0423 F014h

Figure 11-1037. ECC_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED	ERROR_ONCE	FORCE_N_ROW	FORCE_DED	FORCE_SEC	ENABLE_RMW	ECC_CHECK	ECC_ENABLE
R-	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h	R/W-1h	R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2387. ECC_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R		Reserved
6	ERROR_ONCE	R/W	0h	If this bit is set, the force_sec/force_ded will inject an error to the specified row only once. The force_sec bit will be cleared once a writeback happens. If writeback is not enabled, this error will be cleared the cycle following the read when the data is corrected. For double-bit errors, the force_ded bit will be cleared the cycle following the double-bit error. Any subsequent reads will not force an error.
5	FORCE_N_ROW	R/W	0h	Force single/double-bit error on the next RAM read
4	FORCE_DED	R/W	0h	Force double-bit error. Cleared the cycle following the error if error_once is asserted.
3	FORCE_SEC	R/W	0h	Force single-bit error. Cleared on a writeback or the cycle following the error if error_once is asserted
2	ENABLE_RMW	R/W	1h	Enable read-modify-write on partial word writes
1	ECC_CHECK	R/W	1h	Enable ECC check. ECC is completely bypassed if both ecc_enable and ecc_check are 0
0	ECC_ENABLE	R/W	1h	Enable ECC generation

Table 11-2388. Register Call Summary for ECC_CONTROL

EMAC Registers • ECC_CONTROL Register (Offset = 14h) [reset = 7h]: [0] • NSS_0_CFG_ECC Registers: [0]
Networking Subsystem (NSS) • Memory Error Detection and Correction: [0]

11.13.5.1.6.6 ECC_ERROR_CONTROL1 Register (Offset = 18h) [reset = 0h]

ECC_ERROR_CONTROL1 is shown in Figure 11-1038 and described in Table 11-2390.

This register contains ECC error control bits for the selected ECC RAM.

Table 11-2389. ECC_ERROR_CONTROL1 Instances

Instance	Physical Address
NSS_0_CFG_NAVSS_ECC	0400 0818h
NSS_0_CFG_EMAC_ECC	0423 F018h

Figure 11-1038. ECC_ERROR_CONTROL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_BIT1																ECC_ROW															
R/W-0h																R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2390. ECC_ERROR_CONTROL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	ECC_BIT1	R/W	0h	Data bit that needs to be flipped when force_sec or force_ded is set
15-0	ECC_ROW	R/W	0h	Row address where force_sec or force_ded needs to be applied. This is ignored if force_n_row is set.

Table 11-2391. Register Call Summary for ECC_ERROR_CONTROL1

EMAC Registers • ECC_ERROR_CONTROL1 Register (Offset = 18h) [reset = 0h]: [0] • NSS_0_CFG_ECC Registers: [0]
Memory Error Detection and Correction • Packet Header ECC: [0]

11.13.5.1.6.7 ECC_ERROR_CONTROL2 Register (Offset = 1Ch) [reset = 0h]

ECC_ERROR_CONTROL2 is shown in Figure 11-1039 and described in Table 11-2393.

This register contains ECC error control bits for the selected ECC RAM.

Table 11-2392. ECC_ERROR_CONTROL2 Instances

Instance	Physical Address
NSS_0_CFG_NAVSS_ECC	0400 081Ch
NSS_0_CFG_EMAC_ECC	0423 F01Ch

Figure 11-1039. ECC_ERROR_CONTROL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ECC_BIT2															
R-																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2393. ECC_ERROR_CONTROL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R		Reserved
15-0	ECC_BIT2	R/W	0h	Data bit that needs to be flipped when force_ded is set

Table 11-2394. Register Call Summary for ECC_ERROR_CONTROL2

EMAC Registers
- NSS_0_CFG_ECC Registers: [0]
- ECC_ERROR_CONTROL2 Register (Offset = 1Ch) [reset = 0h]: [0]

11.13.5.1.6.8 ECC_ERROR_STATUS1 Register (Offset = 20h) [reset = 0h]

ECC_ERROR_STATUS1 is shown in Figure 11-1040 and described in Table 11-2396.

This register contains ECC status bits for the selected ECC RAM.

Table 11-2395. ECC_ERROR_STATUS1 Instances

Instance	Physical Address
NSS_0_CFG_NAVSS_ECC	0400 0820h
NSS_0_CFG_EMAC_ECC	0423 F020h

Figure 11-1040. ECC_ERROR_STATUS1 Register

31	30	29	28	27	26	25	24
ECC_ROW							
R-0h							
23	22	21	20	19	18	17	16
ECC_ROW							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					CLR_ECC_OTHER	CLR_ECC_DED	CLR_ECC_SEC
R-					W1toCl-0h	W1toCl-0h	W1toCl-0h
7	6	5	4	3	2	1	0
RESERVED					ECC_OTHER	ECC_DED	ECC_SEC
R-					R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; W1toCl = Write 1 to Clear Bit; *-n* = value after reset

Table 11-2396. ECC_ERROR_STATUS1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	ECC_ROW	R	0h	Indicates the row/address where the single or double-bit error occurred
15-11	RESERVED	R		Reserved
10	CLR_ECC_OTHER	W1toCl	0h	1 indicates a pending double-bit error. Writing a 1 clears the status bit.
9	CLR_ECC_DED	W1toCl	0h	1 indicates a pending successive single-bit error. Writing a 1 clears the status bit.
8	CLR_ECC_SEC	W1toCl	0h	1 indicates a pending single-bit error. Writing a 1 clears the status bit.
7-3	RESERVED	R		Reserved
2	ECC_OTHER	R/W	0h	1 indicates that successive single-bit errors have occurred while a writeback is still pending. Software can also write a 1 to set the pending status.
1	ECC_DED	R/W	0h	1 indicates a double-bit error. Software can also write a 1 to set the pending status.
0	ECC_SEC	R/W	0h	1 indicates a single-bit error. Software can also write a 1 to set the pending status.

Table 11-2397. Register Call Summary for ECC_ERROR_STATUS1

EMAC Registers
- ECC_ERROR_STATUS1 Register (Offset = 20h) [reset = 0h]: [0]
- NSS_0_CFG_ECC Registers: [0]

Table 11-2397. Register Call Summary for ECC_ERROR_STATUS1 (continued)

Memory Error Detection and Correction • ECC Interrupts: [0][1][2][3]

11.13.5.1.6.9 ECC_ERROR_STATUS2 Register (Offset = 24h) [reset = 0h]

ECC_ERROR_STATUS2 is shown in Figure 11-1041 and described in Table 11-2399.

This register contains ECC status bits for the selected ECC RAM.

Table 11-2398. ECC_ERROR_STATUS2 Instances

Instance	Physical Address
NSS_0_CFG_NAVSS_ECC	0400 0824h
NSS_0_CFG_EMAC_ECC	0423 F024h

Figure 11-1041. ECC_ERROR_STATUS2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ECC_BIT1															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2399. ECC_ERROR_STATUS2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	ECC_BIT1	R	0h	Indicates the data bit that is in error

Table 11-2400. Register Call Summary for ECC_ERROR_STATUS2

EMAC Registers • NSS_0_CFG_ECC Registers: [0] • ECC_ERROR_STATUS2 Register (Offset = 24h) [reset = 0h]: [0]
Memory Error Detection and Correction • ECC Interrupts: [0][1] • Reads to ECC Control and Status Registers: [0]

11.13.5.1.6.10 ECC_EOI Register (Offset = 3Ch) [reset = 0h]

ECC_EOI is shown in Figure 11-1042 and described in Table 11-2402.

This is the ECC_EOI register for the interrupt to the host.

Table 11-2401. ECC_EOI Instances

Instance	Physical Address
NSS_0_CFG_NAVSS_ECC	0400 083Ch
NSS_0_CFG_EMAC_ECC	0423 F03Ch

Figure 11-1042. ECC_EOI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EOI_WR
R-0h							W1toCl-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2402. ECC_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EOI_WR	R/W	0h	Write to this register indicates that software has acknowledged the pending interrupt and the next interrupt can be sent to the host

Table 11-2403. Register Call Summary for ECC_EOI

EMAC Registers • NSS_0_CFG_ECC Registers: [0][1] • ECC_EOI Register (Offset = 3Ch) [reset = 0h]: [0][1]
Memory Error Detection and Correction • ECC Interrupts: [0]

11.13.5.1.6.11 ECC_INT_STATUS_0 to ECC_INT_STATUS_15 Register (Offset = 40h to 7Ch) [reset = 0h]

ECC_INT_STATUS_0 to ECC_INT_STATUS_15 is shown in Figure 11-1043 and described in Table 11-2405.

These are the raw level interrupt status bits where each bit corresponds to the pending status from an ECC RAM.

Table 11-2404. ECC_INT_STATUS_0 to ECC_INT_STATUS_15 Instances

Instance	Physical Address
NSS_0_CFG_NAVSS_ECC	0400 0840h to 0400 087Ch
NSS_0_CFG_EMAC_ECC	0423 F040h to 0423 F07Ch

Figure 11-1043. ECC_INT_STATUS_0 to ECC_INT_STATUS_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITMASK																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2405. ECC_INT_STATUS_0 to ECC_INT_STATUS_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITMASK	R	0h	Level interrupt status from each ECC RAM: • 0=not pending • 1=pending status

Table 11-2406. Register Call Summary for ECC_INT_STATUS_0

EMAC Registers • ECC_INT_STATUS_0 to ECC_INT_STATUS_15 Register (Offset = 40h to 7Ch) [reset = 0h]: [0] • NSS_0_CFG_ECC Registers: [0]
Memory Error Detection and Correction • ECC Interrupts: [0]

11.13.5.1.6.12 ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15 Register (Offset = 80h to BCh) [reset = 0h]

ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15 is shown in Figure 11-1044 and described in Table 11-2408.

These are interrupt enables associated with the interrupt from each of the ECC RAMs. Writing a 1 to a bit position in the register enables the interrupt from the associated ECC RAM.

Table 11-2407. ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15 Instances

Instance	Physical Address
NSS_0_CFG_NAVSS_ECC	0400 0880h to 0400 08BCh
NSS_0_CFG_EMAC_ECC	0423 F080h to 0423 F0BCh

Figure 11-1044. ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITMASK																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 11-2408. ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITMASK	W	0h	Write 1 to enable interrupt from the associated ECC RAM

Table 11-2409. Register Call Summary for ECC_INT_ENABLE_0

EMAC Registers • NSS_0_CFG_ECC Registers: [0] • ECC_INT_ENABLE_0 to ECC_INT_ENABLE_15 Register (Offset = 80h to BCh) [reset = 0h]: [0]
Memory Error Detection and Correction • ECC Interrupts: [0]

11.13.5.1.6.13 ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15 Register (Offset = C0h to FCh) [reset = 0h]

ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15 is shown in Figure 11-1045 and described in Table 11-2411.

These are interrupt enable clear bits associated with the interrupt from each of the ECC RAMs. Writing a 1 to a bit position in the register disables the interrupt from the associated ECC RAM.

Table 11-2410. ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15 Instances

Instance	Physical Address
NSS_0_CFG_NAVSS_ECC	0400 08C0h to 0400 08FCh
NSS_0_CFG_EMAC_ECC	0423 F0C0h to 0423 F0FCh

Figure 11-1045. ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITMASK																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 11-2411. ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITMASK	W	0h	Write 1 to disable interrupt from the associated ECC RAM

Table 11-2412. Register Call Summary for ECC_INT_CLEAR_0

EMAC Registers
- NSS_0_CFG_ECC Registers: [0]
- ECC_INT_CLEAR_0 to ECC_INT_CLEAR_15 Register (Offset = C0h to FCh) [reset = 0h]: [0]

11.13.5.2 NAVSS Registers

11.13.5.2.1 NSS_0_CFG_NAVSS_CFG Registers

Table 11-2414 lists the memory-mapped registers for the NSS_0_N. All register offset addresses not listed in Table 11-2414 should be considered as reserved locations and the register contents should not be modified.

Table 11-2413. NSS_0_CFG_NAVSS_CFG Instances

Instance	Base Address
NSS_0_CFG_NAVSS_CFG	0400 0000h

Table 11-2414. NSS_0_CFG_NAVSS_CFG Registers

Offset	Acronym	Register Name	NSS_0_CFG_NAVSS_CFG Physical Address	Section
0h	REVISION_REGISTER	NAVSS Revision register	0400 0000h	Section 11.13.5.2.1.1

11.13.5.2.1.1 REVISION_REGISTER Register (Offset = 0h) [reset = 4EFC1900h]

REVISION_REGISTER is shown in Figure 11-1046 and described in .

Table 11-2415. REVISION_REGISTER Instances

Instance	Physical Address
NSS_0_CFG_NAVSS_CFG	0400 0000h

Figure 11-1046. REVISION_REGISTER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 4EFC1900h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2416. REVISION_REGISTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4EFC 1900h	TI internal data. Identifies revision of peripheral.

Table 11-2417. Register Call Summary for REVISION_REGISTER

NAVSS Registers
- NSS_0_CFG_NAVSS_CFG Registers: [0]
- REVISION_REGISTER Register (Offset = 0h) [reset = 4EFC1900h]: [0]

11.13.5.2.2 NSS_0_INTD Registers

lists the memory-mapped registers for the NSS_0_INTD. All register offset addresses not listed in Table 11-2505 should be considered as reserved locations and the register contents should not be modified.

Table 11-2418. NSS_0_CFG_INTD Instances

Instance	Base Address
NSS_0_CFG_INTD	0400 1000h

Table 11-2419. NSS_0_CFG_INTD Registers

Offset	Acronym	Register Name	NSS_0_CFG_INTD Physical Address	Section
0h	INTD_REVISION_REG	Revision Register	0400 1000h	Section 11.13.5.2.2.1
4h	INTD_CONTROL_REG	End of Interrupt Register	0400 1004h	Section 11.13.5.2.2.2
10h	INTD_EOI_REG	Interrupt Vector Register	0400 1010h	Section 11.13.5.2.2.3
140h	INTD_INTR_VECTOR_REG	Linking RAM Region 0 Size Register	0400 1140h	Section 11.13.5.2.2.4
200h	INTD_STATUS_REG0	Status Register 0	0400 1200h	Section 11.13.5.2.2.5
280h	INTD_STATUS_CLR_REG0	Status Clear Register 0	0400 1280h	Section 11.13.5.2.2.6
300h	INTD_INTCNT_REG0	Interrupt Counter Register for HOST_CNT_INT00CDMA0_STARVE_INTR_INT	0400 1300h	Section 11.13.5.2.2.7
480h	INTD_INTR_VECTOR_REG_HOST	Interrupt Vector for host	0400 1480h	Section 11.13.5.2.2.8

11.13.5.2.2.1 INTD_REVISION_REG Register (Offset = 0h) [reset = 1E83A100h]

INTD_REVISION_REG is shown in Figure 11-1047 and described in Table 11-2421.

Table 11-2420. INTD_REVISION_REG Instances

Instance	Physical Address
NSS_0_CFG_INTD	0400 1000h

Figure 11-1047. INTD_REVISION_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 1E83A100h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2421. INTD_REVISION_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	1E83A100h	TI internal data. Identifies revision of peripheral.

Table 11-2422. Register Call Summary for INTD_REVISION_REG

NAVSS Registers
- INTD_REVISION_REG Register (Offset = 0h) [reset = 1E83A100h]: [0]
- NSS_0_INTD Registers: [0]

11.13.5.2.2.2 INTD_CONTROL_REG Register (Offset = 4h) [reset = 0h]

INTD_CONTROL_REG is shown in and described in Table 11-2424.

Table 11-2423. INTD_CONTROL_REG Instances

Instance	Physical Address
NSS_0_CFG_INTD	0400 1004h

Figure 11-1048. INTD_CONTROL_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R - 0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2424. INTD_CONTROL_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	0h	Reserved

Table 11-2425. Register Call Summary for INTD_CONTROL_REG

NAVSS Registers
- INTD_CONTROL_REG Register (Offset = 4h) [reset = 0h]: [0]
- NSS_0_INTD Registers: [0]

11.13.5.2.2.3 INTD_EOI_REG Register (Offset = 10h) [reset = 0h]

INTD_EOI_REG is shown in Figure 11-1049 and described in Table 11-2427.

Table 11-2426. INTD_EOI_REG Instances

Instance	Physical Address
NSS_0_CFG_INTD	0400 1010h

Figure 11-1049. INTD_EOI_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								EOI_VECTOR							
R-0h																								R/W-0h							

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 11-2427. INTD_EOI_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	W	0h	Reserved
7-0	EOI_VECTOR	R	0h	End of Interrupt Vector

Table 11-2428. Register Call Summary for INTD_EOI_REG

NAVSS Registers
- INTD_EOI_REG Register (Offset = 10h) [reset = 0h]: [0]
- NSS_0_INTD Registers: [0]

11.13.5.2.2.4 INTD_INTR_VECTOR_REG Register (Offset = 14h) [reset = 0h]

INTD_INTR_VECTOR_REG is shown in Figure 11-1050 and described in Table 11-2430.

Table 11-2429. INTD_INTR_VECTOR_REG Instances

Instance	Physical Address
NSS_0_CFG_INTD	0400 1014h

Figure 11-1050. INTD_INTR_VECTOR_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTR_VECTOR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2430. INTD_INTR_VECTOR_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INTR_VECTOR	R	0h	Interrupt Vector Register. The register has the latest prioritized interrupt values.

Table 11-2431. Register Call Summary for INTD_INTR_VECTOR_REG

NAVSS Registers
- INTD_INTR_VECTOR_REG Register (Offset = 14h) [reset = 0h]: [0]
- NSS_0_INTD Registers: [0]

11.13.5.2.2.5 INTD_STATUS_REG0 Register (Offset = 200h) [reset = 0h]

INTD_STATUS_REG0 is shown in Figure 11-1051 and described in Table 11-2433.

Table 11-2432. INTD_STATUS_REG0 Instances

Instance	Physical Address
NSS_0_CFG_INTD	0400 1200h

Figure 11-1051. INTD_STATUS_REG0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						STATUS_HOST_INT09SA_UL_PKA_IN_INTR	STATUS_HOST_INT08SA_UL_TRNG_IN_INTR
R-0h						R-0h	R-0h
7	6	5	4	3	2	1	0
STATUS_HOST_INT07MDIO_USER1_IN_INTR	STATUS_HOST_INT06MDIO_USER0_IN_INTR	STATUS_HOST_INT05MDIO_LINK1_IN_INTR	STATUS_HOST_INT04MDIO_LINK0_IN_INTR	STATUS_HOST_INT03ESW_EVNT_PEND_IN_INTR	STATUS_HOST_INT02ESW_STAT_PEND1_IN_INTR	STATUS_HOST_INT01ESW_STAT_PEND0_IN_INTR	STATUS_HOST_INT00CDMA0_STARVE_INTR_INT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	W1TS - 0h

LEGEND: R = Read Only; W1TS = Write 1 to Set Bit; -*n* = value after reset

Table 11-2433. INTD_STATUS_REG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	STATUS_HOST_INT09SA_UL_PKA_IN_INTR	R	0h	Status for INT09SA_UL_PKA_IN_INTR
8	STATUS_HOST_INT08SA_UL_TRNG_IN_INTR	R	0h	Status for INT08SA_UL_TRNG_IN_INTR
7	STATUS_HOST_INT07MDIO_USER1_IN_INTR	R	0h	Status for INT07MDIO_USER1_IN_INTR
6	STATUS_HOST_INT06MDIO_USER0_IN_INTR	R	0h	Status for INT06MDIO_USER0_IN_INTR
5	STATUS_HOST_INT05MDIO_LINK1_IN_INTR	R	0h	Status for INT05MDIO_LINK1_IN_INTR
4	STATUS_HOST_INT04MDIO_LINK0_IN_INTR	R	0h	Status for INT04MDIO_LINK0_IN_INTR
3	STATUS_HOST_INT03ESW_EVNT_PEND_IN_INTR	R	0h	Status for INT03ESW_EVNT_PEND_IN_INTR
2	STATUS_HOST_INT02ESW_STAT_PEND1_IN_INTR	R	0h	Status for INT02ESW_STAT_PEND1_IN_INTR
1	STATUS_HOST_INT01ESW_STAT_PEND0_IN_INTR	R	0h	Status for INT01ESW_STAT_PEND0_IN_INTR
0	STATUS_HOST_INT00CDMA0_STARVE_INTR_INT	W1TS	0h	Status (write 1 to set) for INT00CDMA0_STARVE_INTR_INT

Table 11-2434. Register Call Summary for INTD_STATUS_REG0

NAVSS Registers
• NSS_0_INTD Registers: [0] • INTD_STATUS_REG0 Register (Offset = 200h) [reset = 0h]: [0]

11.13.5.2.2.6 INTD_STATUS_CLR_REG0 Register (Offset = 280h) [reset = 0h]

INTD_STATUS_CLR_REG0 is shown in Figure 11-1052 and described in Table 11-2436.

Table 11-2435. INTD_STATUS_CLR_REG0 Instances

Instance	Physical Address
NSS_0_CFG_INTD	0400 1280h

Figure 11-1052. INTD_STATUS_CLR_REG0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															STATUS_HOST_INT00CDMA0_STARVE_INTR_INT_CLR
R - 0h																															W1TC - 0h

LEGEND: R = Read Only; W1TC = Write 1 to Clear Bit; *-n* = value after reset

Table 11-2436. INTD_STATUS_CLR_REG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	STATUS_HOST_INT00CDMA0_STARVE_INTR_INT_CLR	W1TC	0h	Status (write 1 to clear) for INT00CDMA0_STARVE_INTR_INT

Table 11-2437. Register Call Summary for INTD_STATUS_CLR_REG0

NAVSS Registers
- NSS_0_INTD Registers: [0]
- INTD_STATUS_CLR_REG0 Register (Offset = 280h) [reset = 0h]: [0]

11.13.5.2.2.7 INTD_INTCNT_REG0 Register (Offset = 300h) [reset = 0h]

INTD_INTCNT_REG0 is shown in Figure 11-1053 and described in Table 11-2439.

Table 11-2438. INTD_INTCNT_REG0 Instances

Instance	Physical Address
NSS_0_CFG_INTD	0400 1300h

Figure 11-1053. INTD_INTCNT_REG0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								INTCNT_HOST_CNT_INT00CDMA0_STARVE_INTR_INT							
R-0h																								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2439. INTD_INTCNT_REG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	INTCNT_HOST_CNT_INT00CDMA0_STARVE_INTR_INT	R	0h	Interrupt Count for HOST_CNT_INT00CDMA0_STARVE_INTR_INT (write to decrement)

Table 11-2440. Register Call Summary for INTD_INTCNT_REG0

NAVSS Registers
- INTD_INTCNT_REG0 Register (Offset = 300h) [reset = 0h]: [0]
- NSS_0_INTD Registers: [0]

11.13.5.2.2.8 INTD_INTR_VECTOR_REG_HOST (Offset = 480h) [reset = 0h]

INTD_INTR_VECTOR_REG_HOST is shown in Figure 11-1054 and described in .

Table 11-2441. INTD_INTR_VECTOR_REG_HOST Instances

Instance	Physical Address
NSS_0_CFG_INTD	0400 1480h

Figure 11-1054. INTD_INTR_VECTOR_REG_HOST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTR_VECTOR_HOST																															
R - 0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2442. INTD_INTR_VECTOR_REG_HOST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INTR_VECTOR_HOST	R	0h	Interrupt Vector

11.13.5.2.3 CDMA Registers

11.13.5.2.3.1 NSS_0_CFG_CPPIDMA0_CONFIG Registers

Table 11-2444 lists the memory-mapped registers for the NSS_0_CFG_CPPIDMA0_CONFIG. All register offset addresses not listed in Table 11-2444 should be considered as reserved locations and the register contents should not be modified.

Table 11-2443. NSS_0_CFG_CPPIDMA0_CONFIG Instances

Instance	Base Address
NSS_0_CFG_CPPIDMA0_CONFIG	0401 0000h

Table 11-2444. NSS_0_CFG_CPPIDMA0_CONFIG Registers

Offset	Acronym	Register Name	NSS_0_CFG_CPPIDMA0_CONFIG Physical Address	Section
0h	CDMA_REVISION_REG	Revision Register	0401 0000h	Section 11.13.5.2.3.1.1
4h	CDMA_PERF_CONTROL_REG	Performance Control Register	0401 0004h	Section 11.13.5.2.3.1.2
8h	CDMA_EMULATION_CONTROL_REG	Emulation Control Register	0401 0008h	Section 11.13.5.2.3.1.3
Ch	CDMA_PRIORITY_CONTROL_REG	Priority Control Register	0401 000Ch	Section 11.13.5.2.3.1.4
10h to 1Ch	CDMA_QM_BASE_ADDRESS_REG_0 to CDMA_QM_BASE_ADDRESS_REG_3	Queue Manager 0 Base Address to Queue Manager 3 Base Address Registers	0401 0010h to 0401 001Ch	Section 11.13.5.2.3.1.5

11.13.5.2.3.1.1 CDMA_REVISION_REG Register (Offset = 0h) [reset = 4E5AA900h]

CDMA_REVISION_REG is shown in Figure 11-1055 and described in Table 11-2446.

Return to Summary Table.

Table 11-2445. CDMA_REVISION_REG Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_CONFIG	0401 0000h

Figure 11-1055. CDMA_REVISION_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 4E5AA900h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2446. CDMA_REVISION_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4E5AA900h	TI internal data. Identifies revision of peripheral.

Table 11-2447. Register Call Summary for CDMA_REVISION_REG

NAVSS Registers
- CDMA_REVISION_REG Register (Offset = 0h) [reset = 4E5AA900h]: [0]
- NSS_0_CFG_CPPIDMA0_CONFIG Registers: [0]

11.13.5.2.3.1.2 CDMA_PERF_CONTROL_REG Register (Offset = 4h) [reset = 200000h]

CDMA_PERF_CONTROL_REG is shown in Figure 11-1056 and described in Table 11-2449.

Return to Summary Table.

Table 11-2448. CDMA_PERF_CONTROL_REG Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_CONFIG	0401 0004h

Figure 11-1056. CDMA_PERF_CONTROL_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED										WARB_FIFO_DEPTH					
R-0h										R/W-20h					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMEOUT_CNT															
R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2449. CDMA_PERF_CONTROL_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved
21-16	WARB_FIFO_DEPTH	R/W	20h	This field sets the depth of the write arbitration FIFO which stores write transaction information between the command arbiter and write data arbiters in the Bus Interface Unit. This value can be set to a range of 1 to 32. Setting this field to smaller values will cause prevent the CDMAHP from having an excess of write transactions outstanding whose data is still waiting to be transferred. System performance can suffer if write commands are allowed to be issued long before the corresponding write data will be transferred. This field allows the command count to be optimized based on system dynamics.
15-0	TIMEOUT_CNT	R/W	0h	This field sets the timeout duration in clock cycles. This field controls the minimum amount of time that an Rx channel will be required to wait when it encounters a buffer starvation condition and the Rx error handling bit is set to 1 (packet is to be preserved - no discard). If the Rx error handling bit in the flow table is cleared, this field will have no effect on the Rx operation. When this field is set to 0, the Rx engine will not force an Rx channel to wait after encountering a starvation event (the feature is disabled). When this field is set to a value other than 0, the Rx engine will force any channel whose associated flow had the Rx error handling bit asserted and which encounters starvation to wait for at least the specified # of clock cycles before coming into context again to retry the access to the QM. This is intended to control potentially debilitating effects on the QM performance that can becaused by the CDMAHP modules continually polling the QM. The exact # of clock cycles between QM access attempts is not important and will not be exact. The only guarantee is that the # of cycles waited will be at least as large as the TIMEOUT_CNT.

Table 11-2450. Register Call Summary for CDMA_PERF_CONTROL_REG

NAVSS Registers
- NSS_0_CFG_CPPIDMA0_CONFIG Registers: [0]
- CDMA_PERF_CONTROL_REG Register (Offset = 4h) [reset = 200000h]: [0]

11.13.5.2.3.1.3 CDMA_EMULATION_CONTROL_REG Register (Offset = 8h) [reset = 8000 0000h]

CDMA_EMULATION_CONTROL_REG is shown in Figure 11-1057 and described in Table 11-2452.

Return to Summary Table.

Table 11-2451. CDMA_EMULATION_CONTROL_REG Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_CONFIG	0401 0008h

Figure 11-1057. CDMA_EMULATION_CONTROL_REG Register

31	30	29	28	27	26	25	24
LOOPBACK_EN	RESERVED						
R/W-1h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						SOFT	FREE
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2452. CDMA_EMULATION_CONTROL_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	LOOPBACK_EN	R/W	1h	Streaming Interface loopback enable
30-2	RESERVED	R	0h	Reserved
1	SOFT	R/W	0h	Soft
0	FREE	R/W	0h	Free

Table 11-2453. Register Call Summary for CDMA_EMULATION_CONTROL_REG

Gigabit Ethernet MAC (EMAC) Subsystem • Emulation Control: [0]
NAVSS Registers • NSS_0_CFG_CPPIDMA0_CONFIG Registers: [0] • CDMA_EMULATION_CONTROL_REG Register (Offset = 8h) [reset = 8000 0000h]: [0]

11.13.5.2.3.1.4 CDMA_PRIORITY_CONTROL_REG Register (Offset = Ch) [reset = 0h]

CDMA_PRIORITY_CONTROL_REG is shown in Figure 11-1058 and described in Table 11-2455.

Return to Summary Table.

Table 11-2454. CDMA_PRIORITY_CONTROL_REG Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_CONFIG	0401 000Ch

Figure 11-1058. CDMA_PRIORITY_CONTROL_REG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					RX_PRIORITY		
R-0h					R/W-0h		
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					TX_PRIORITY		
R-0h					R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2455. CDMA_PRIORITY_CONTROL_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18-16	RX_PRIORITY	R/W	0h	Rx priority: This field contains the 3-bit value which will be output on the mem_cpriority and mem_cepriority outputs during all Rx transactions.
15-3	RESERVED	R	0h	Reserved
2-0	TX_PRIORITY	R/W	0h	Tx priority: This field contains the 3-bit value which will be output on the mem_cpriority and mem_cepriority outputs during all Tx transactions.

Table 11-2456. Register Call Summary for CDMA_PRIORITY_CONTROL_REG

NAVSS Registers

- CDMA_PRIORITY_CONTROL_REG Register (Offset = Ch) [reset = 0h]: [0]
- NSS_0_CFG_CPPIDMA0_CONFIG Registers: [0]

11.13.5.2.3.1.5 CDMA_QM_BASE_ADDRESS_REG_0 to CDMA_QM_BASE_ADDRESS_REG_3 Register (Offset = 10h to 1Ch) [reset = 0h]

CDMA_QM_BASE_ADDRESS_REG_0 to CDMA_QM_BASE_ADDRESS_REG_3 is shown in Figure 11-1059 and described in Table 11-2458.

Return to Summary Table.

Table 11-2457. CDMA_QM_BASE_ADDRESS_REG_0 to CDMA_QM_BASE_ADDRESS_REG_3 Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_CONFIG	0401 0010h to 0401 001Ch

Figure 11-1059. CDMA_QM_BASE_ADDRESS_REG_0 to CDMA_QM_BASE_ADDRESS_REG_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QMI_BASE[1]																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

(1) i = 0 to 3

Table 11-2458. CDMA_QM_BASE_ADDRESS_REG_0 to CDMA_QM_BASE_ADDRESS_REG_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QMI_BASE	R/W	0h	Provides a programmable pointer to the base address of the queues region for this instance of the Queue Manager

Table 11-2459. Register Call Summary for CDMA_QM_BASE_ADDRESS_REG_0

NAVSS Registers
- NSS_0_CFG_CPPIDMA0_CONFIG Registers: [0]
- CDMA_QM_BASE_ADDRESS_REG_0 to CDMA_QM_BASE_ADDRESS_REG_3 Register (Offset = 10h to 1Ch) [reset = 0h]: [0]

11.13.5.2.3.2 NSS_0_CFG_CPPIDMA0_TX_SCHEDULER Registers

Table 11-2461 lists the memory-mapped registers for the NSS_0_CFG_CPPIDMA0_TX_SCHEDULER. All register offset addresses not listed in Table 11-2461 should be considered as reserved locations and the register contents should not be modified.

Table 11-2460. NSS_0_CFG_CPPIDMA0_TX_SCHEDULER Instances

Instance	Base Address
NSS_0_CFG__cppidma0_tx_scheduler	0401 0100h

Table 11-2461. NSS_0_CFG_CPPIDMA0_TX_SCHEDULER Registers

Offset	Acronym	Register Name	NSS_0_CFG_CPPIDMA0_TX_SCHEDULER Physical Address	Section
0h to 20h	CDMA_TX_CHANNEL_SCHEDULER_CONFIG_REG_0 to CDMA_TX_CHANNEL_SCHEDULER_CONFIG_REG_8	Tx Channel 0 Scheduler Config Register to Tx Channel 8 Scheduler Config Register	0401 0100h to 0401 0020h	Section 11.13.5.2.3.2.1

11.13.5.2.3.2.1 CDMA_TX_CHANNEL_SCHEDULER_CONFIG_REG_0 to CDMA_TX_CHANNEL_SCHEDULER_CONFIG_REG_8 Register (Offset = 0h to 20h) [reset = 0h]

CDMA_TX_CHANNEL_SCHEDULER_CONFIG_REG_0 to CDMA_TX_CHANNEL_SCHEDULER_CONFIG_REG_8 is shown in Figure 11-1060 and described in Table 11-2463.

Table 11-2462. CDMA_TX_CHANNEL_SCHEDULER_CONFIG_REG_0 to CDMA_TX_CHANNEL_SCHEDULER_CONFIG_REG_8 Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_TX_SCHEDULER	4010100h to 4010020h

Figure 11-1060. CDMA_TX_CHANNEL_SCHEDULER_CONFIG_REG_0 to CDMA_TX_CHANNEL_SCHEDULER_CONFIG_REG_8 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						PRIORITY	
R-0h						R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2463. CDMA_TX_CHANNEL_SCHEDULER_CONFIG_REG_0 to CDMA_TX_CHANNEL_SCHEDULER_CONFIG_REG_8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	PRIORITY	R/W	0h	Tx scheduling priority: • 0=high • 1=medium-high • 3=low

Table 11-2464. Register Call Summary for CDMA_TX_CHANNEL_SCHEDULER_CONFIG_REG_0

NAVSS Registers
- CDMA_TX_CHANNEL_SCHEDULER_CONFIG_REG_0 to CDMA_TX_CHANNEL_SCHEDULER_CONFIG_REG_8 Register (Offset = 0h to 20h) [reset = 0h]: [0]
- NSS_0_CFG_CPPIDMA0_TX_SCHEDULER Registers: [0]

11.13.5.2.3.3 NSS_0_CFG_CPPIDMA0_TX_CHANNEL_CFG Registers

Table 11-2466 lists the memory-mapped registers for the NSS_0_CFG_CPPIDMA0_TX_CHANNEL_CFG. All register offset addresses not listed in Table 11-2466 should be considered as reserved locations and the register contents should not be modified.

Table 11-2465. NSS_0_CFG_CPPIDMA0_TX_CHANNEL_CFG Instances

Instance	Base Address
NSS_0_CFG_CPPIDMA0_TX_CHANNEL_CFG	0401 1000h

Table 11-2466. NSS_0_CFG_CPPIDMA0_TX_CHANNEL_CFG Registers

Offset	Acronym	Register Name	NSS_0_CFG__cppidma0_tx_channel_cfg Physical Address	Section
0h + (i * 20h)[1]	CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_0 to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_8	Tx Channel i Global Config Register A	0401 1000h + (i * 20h)	Section 11.13.5.2.3.3.1
4h + (i * 20h)	CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_0 to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_8	Tx Channel i Global Config Register B	0401 1004h + (i * 20h)	Section 11.13.5.2.3.3.2

(1) i = 0 to 8

11.13.5.2.3.3.1 CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_0 to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_8 Register (Offset = 0h + [i * 20h], where i = 0 to 8) [reset = 0h]

CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_0 to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_8 is shown in Figure 11-1061 and described in Table 11-2468.

Table 11-2467. CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_0 to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_8 Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_TX_CHANNEL_CFG	0401 1000h + [i * 20h], where i = 0 to 8

Figure 11-1061. CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_0 to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_8 Register

31	30	29	28	27	26	25	24
TX_ENABLE	TX_TEARDOWN	TX_PAUSE	RESERVED				
R/W-0h	R/W-0h	R/W-0h	R-0h				
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2468. CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_0 to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	TX_ENABLE	R/W	0h	Tx channel enable: • 0 = Channel is disabled • 1 = Channel is enabled Disabling a channel halts operation on the channel after the current block transfer is completed. Disabling a channel in the middle of a packet transfer may result in underflow conditions in the attached application block and data loss.
30	TX_TEARDOWN	R/W	0h	Setting this bit will request the channel to be torn down. This field will remain set after a channel teardown is complete.
29	TX_PAUSE	R/W	0h	Setting this bit will request the channel to pause processing at the next packet boundary. This is a more graceful method of halting processing than disabling the channel as it will not allow any current packets to underflow.
28-0	RESERVED	R	0h	Reserved

Table 11-2469. Register Call Summary for CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_0

Navigator Subsystem (NAVSS)
- Transmit Channel Real Time Control/Status: [0]
- Transmit DMA Controller Channel Setup (All Packet Types): [0]
- Transmit Channel Teardown (All Packet Types): [0][1]

Table 11-2469. Register Call Summary for CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_0 (continued)

NAVSS Registers
• NSS_0_CFG_CPPIDMA0_TX_CHANNEL_CFG Registers: [0] • CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_0 to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_A_8 Register (Offset = 0h + [i * 20h], where i = 0 to 8) [reset = 0h]: [0]

11.13.5.2.3.3.2 CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_0 to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_8 Register (Offset = 4h + [i * 0h], where i = 0 to 8) [reset = 0h]

CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_0 to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_8 is shown in Figure 11-1062 and described in Table 11-2471.

Table 11-2470. CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_0 to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_8 Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_TX_CHANNEL_CFG	0401 1004h + [i * 20h], where i = 0 to 8

Figure 11-1062. CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_0 to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_8 Register

31	30	29	28	27	26	25	24
RESERVED	TX_FILT_EINFO	TX_FILT_PSWORDS	RESERVED				TX_AIF_MONO_MODE
R-0h	W-0h	W-0h	R-0h				W-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 11-2471. CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_0 to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30	TX_FILT_EINFO	W	0h	This field controls whether or not the DMA controller will pass the extended packet information fields, if present, from the descriptor to the back end application • 0 = DMA controller will pass extended packet info words if they are present in the descriptor • 1 = DMA controller will filter extended packet info words
29	TX_FILT_PSWORDS	W	0h	This field controls whether or not the DMA controller will pass the protocol specific words, if present, from the descriptor to the back end application • 0 = DMA controller will pass PS words if present in descriptor • 1 = DMA controller will filter PS words
28-25	RESERVED	R	0h	Reserved
24	TX_AIF_MONO_MODE	W	0h	This field when set indicates that all monolithic packets which will be transferred on this channel will be formatted in an optimal configuration as needed by the Antenna Interface Peripheral • 0 = Normal Mode • 1 = AIF specific
23-0	RESERVED	R	0h	Reserved

Table 11-2472. Register Call Summary for CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_0

Navigator Subsystem (NAVSS) • Transmit Channel Configuration: [0]
NAVSS Registers • NSS_0_CFG_CPPIDMA0_TX_CHANNEL_CFG Registers: [0] • CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_0 to CDMA_TX_CHANNEL_GLOBAL_CONFIG_REG_B_8 Register (Offset = 4h + [i * 0h], where i = 0 to 8) [reset = 0h]: [0]

11.13.5.2.3.4 NSS_0_CFG_CPPIDMA0_RX_CHANNEL_CFG Registers

Table 11-2474 lists the memory-mapped registers for the NSS_0_CFG_CPPIDMA0_RX_CHANNEL_CFG. All register offset addresses not listed in Table 11-2474 should be considered as reserved locations and the register contents should not be modified.

Table 11-2473. NSS_0_CFG_CPPIDMA0_RX_CHANNEL_CFG Instances

Instance	Base Address
NSS_0_CFG_CPPIDMA0_RX_CHANNEL_CFG	0401 2000h

Table 11-2474. NSS_0_CFG_CPPIDMA0_RX_CHANNEL_CFG Registers

Offset	Acronym	Register Name	NSS_0_CFG_CPPIDMA0_RX_CHANNEL_CFG Physical Address	Section
0h + (i * 20h)	CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_0 to CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_25	Rx Channel 0 Global Config Register A to Rx Channel 25 Global Config Register A	0401 2000h + (i * 20h)	Section 11.13.5.2.3.4.1

11.13.5.2.3.4.1 CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_0 to CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_25 Register (Offset = 0h + [i * 20h], where i = 0 to 25) [reset = 0h]

CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_0 to CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_25 is shown in Figure 11-1063 and described in Table 11-2476.

Table 11-2475. CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_0 to CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_25 Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_RX_CHANNEL_CFG	0401 2000h + [i * 20h], where i = 0 to 25

Figure 11-1063. CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_0 to CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_25 Register

31	30	29	28	27	26	25	24
RX_ENABLE	RX_TEARDOWN	RX_PAUSE	RESERVED				
R/W-0h	R/W-0h	R/W-0h	R-0h				
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2476. CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_0 to CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_25 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RX_ENABLE	R/W	0h	Tx channel enable: • 0 = Channel is disabled • 1 = Channel is enabled
30	RX_TEARDOWN	R/W	0h	This field indicates whether or not an Rx teardown operation is complete. This field should be cleared when a channel is initialized. This field will be set after a channel teardown is complete.
29	RX_PAUSE	R/W	0h	Setting this bit will request the channel to pause processing at the next packet boundary. This is a more graceful method of halting processing than disabling the channel as it will not allow any current packets to overflow.
28-0	RESERVED	R	0h	Reserved

Table 11-2477. Register Call Summary for CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_0

Navigator Subsystem (NAVSS) • Receive Channel Teardown: [0][1] • Rx DMA Real-Time Control/Status: [0] • Receive Channel Setup (All Packet Types): [0][1]
NAVSS Registers • CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_0 to CDMA_RX_CHANNEL_GLOBAL_CONFIG_REG_25 Register (Offset = 0h + [i * 20h], where i = 0 to 25) [reset = 0h]: [0] • NSS_0_CFG_CPPIDMA0_RX_CHANNEL_CFG Registers: [0]

11.13.5.2.3.5 NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG Registers

Table 11-2479 lists the memory-mapped registers for the NSS_0_CFG_CPPIDMA0_CDMA_RX_FLOW_CFG. All register offset addresses not listed in Table 11-2479 should be considered as reserved locations and the register contents should not be modified.

Table 11-2478. NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG Instances

Instance	Base Address
NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG	0401 3000h

Table 11-2479. NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG Registers

Offset	Acronym	Register Name	NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG Physical Address	Section
0h + (i * 20h)[1]	CDMA_RX_FLOW_CONFIG_REG_A_0 to CDMA_RX_FLOW_CONFIG_REG_A_31	Rx Flow i Config Register A	0401 3000h + (i * 20h)	Section 11.13.5.2.3.5.1
4h+ (i * 20h)	CDMA_RX_FLOW_CONFIG_REG_B_0 to CDMA_RX_FLOW_CONFIG_REG_B_31	Rx Flow i Config Register B	0401 3004h + (i * 20h)	Section 11.13.5.2.3.5.2
8h+ (i * 20h)	CDMA_RX_FLOW_CONFIG_REG_C_0 to CDMA_RX_FLOW_CONFIG_REG_C_31	Rx Flow i Config Register C	0401 3008h + (i * 20h)	Section 11.13.5.2.3.5.3
Ch+ (i * 20h)	CDMA_RX_FLOW_CONFIG_REG_D_0 to CDMA_RX_FLOW_CONFIG_REG_D_31	Rx Flow i Config Register D	0401 300Ch + (i * 20h)	Section 11.13.5.2.3.5.4
10h+ (i * 20h)	CDMA_RX_FLOW_CONFIG_REG_E_0 to CDMA_RX_FLOW_CONFIG_REG_E_31	Rx Flow i Config Register E	0401 3010h + (i * 20h)	Section 11.13.5.2.3.5.5
14h+ (i * 20h)	CDMA_RX_FLOW_CONFIG_REG_F_0 to CDMA_RX_FLOW_CONFIG_REG_F_31	Rx Flow i Config Register F	0401 3014h + (i * 20h)	Section 11.13.5.2.3.5.6
18h+ (i * 20h)	CDMA_RX_FLOW_CONFIG_REG_G_0 to CDMA_RX_FLOW_CONFIG_REG_G_31	Rx Flow i Config Register G	0401 3018h + (i * 20h)	Section 11.13.5.2.3.5.7
1Ch+ (i * 20h)	CDMA_RX_FLOW_CONFIG_REG_H_0 to CDMA_RX_FLOW_CONFIG_REG_H_31	Rx Flow i Config Register H	0401 301Ch + (i * 20h)	Section 11.13.5.2.3.5.8

[1] i = 0 to 31

11.13.5.2.3.5.1 CDMA_RX_FLOW_CONFIG_REG_A_0 to CDMA_RX_FLOW_CONFIG_REG_A_31 Register (Offset = 0h + [i * 20h], where i = 0 to 31) [reset = 0h]

CDMA_RX_FLOW_CONFIG_REG_A_0 to CDMA_RX_FLOW_CONFIG_REG_A_31 is shown in Figure 11-1064 and described in Table 11-2481.

Table 11-2480. CDMA_RX_FLOW_CONFIG_REG_A_0 to CDMA_RX_FLOW_CONFIG_REG_A_31 Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG	0401 3000h + [i * 20h], where i = 0 to 31

Figure 11-1064. CDMA_RX_FLOW_CONFIG_REG_A_0 to CDMA_RX_FLOW_CONFIG_REG_A_31 Register

31	30	29	28	27	26	25	24
RESERVED	RX_EINFO_PRESENT	RX_PSINFO_PRESENT	RX_ERROR_HANDLING	RX_DESC_TYPE		RX_PS_LOCATION	RX_SOP_OFFSET
R-0h	W-0h	W-0h	W-0h	W-0h		W-0h	W-0h
23	22	21	20	19	18	17	16
RX_SOP_OFFSET							
W-0h							
15	14	13	12	11	10	9	8
RESERVED		RX_DEST_QMGR		RX_DEST_QNUM			
R-0h		W-0h		W-0h			
7	6	5	4	3	2	1	0
RX_DEST_QNUM							
W-0h							

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 11-2481. CDMA_RX_FLOW_CONFIG_REG_A_0 to CDMA_RX_FLOW_CONFIG_REG_A_31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30	RX_EINFO_PRESENT	W	0h	Rx extended packet info block present. This bit controls whether or not the Extended Packet Info Block will be present in the Rx Packet Descriptor. If this bit is clear, the port DMA will clear the Extended Packet Info Present bit in the PD and will drop any Timestamp or SW Data words that are presented from the back end application. If this bit is set, the port DMA will set the Extended Packet Info Block Present bit in the PD and will copy any Timestamp or SW Data words that are presented across the Rx streaming interface into the Extended Packet Info Block words in the descriptor. If no Timestamp or SW Data words are presented from the back end application, the port DMA will overwrite the fields in the PD with zeroes.
29	RX_PSINFO_PRESENT	W	0h	Rx protocol specific words present. This bit controls whether or not the Protocol Specific words will be present in the Rx Packet Descriptor. If this bit is clear, the port DMA will set the PS word count to 0 in the PD and will drop any PS words that are presented from the back end application. If this bit is set, the port DMA will set the PS word count to the value given by the back end application and will copy the PS words from the back end application to the location.

Table 11-2481. CDMA_RX_FLOW_CONFIG_REG_A_0 to CDMA_RX_FLOW_CONFIG_REG_A_31 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
28	RX_ERROR_HANDLING	W	0h	Rx error handling mode: • 0 = Starvation errors result in dropping packet and reclaiming any used descriptor or buffer resources back to the original queues/pools they were allocated to • 1 = Starvation errors result in subsequent re-try of the descriptor allocation operation In this mode, the DMA will save it's internal operational state back to the internal state RAM without issuing an advance operation to it's internal FIFO buffers. This results in the DMA reinitiating the data transfer at a later time with the intention that additional free buffers and/or descriptors will have been added.
27-26	RX_DESC_TYPE	W	0h	Rx descriptor type: • 00 = RESERVED • 01 = Host • 10 = Monolithic • 11 = RESERVED
25	RX_PS_LOCATION	W	0h	Rx protocol specific location. If this bit is cleared, the DMA will clear the Protocol Specific Region Location bit in the PD and will place the Protocol Specific Words at the end of the Packet Descriptor. If this bit is set, the DMA will set the Protocol Specific Region Location bit in the PD and will place the Protocol Specific Words at the beginning of the data buffer. When this mode is used, it is required that the resulting target data buffer pointer (which is calculated by adding theHOST_RX_SOP_OFFSET to the original buffer pointer in the Packet Descriptor) is aligned to a 32-bit boundary to avoid unwanted buffer truncation as the DMA will round up to the next 32-bit aligned boundary.
24-16	RX_SOP_OFFSET	W	0h	Rx start of packet offset. This field specifies the number of bytes that are to be skipped in the SOP buffer before beginning to write the payload or protocol specific bytes(if they are in the sop buffer). This value must be less than the minimum size of a buffer in the system. Valid values are 0 - 255 bytes. Note that for monolithic packets the value of this field must always be initialized to be greater than or equal to 4 times the maximum number of protocol specific 32-bit words that are required by any of the packet types that will be transferred by this channel. This is important as the primary purpose of this field is to ensure that space is left in the descriptor to place the protocol specific information without overwriting or being overwritten by the Rx data. The secondary purpose of this field is to allow space to be left prior to the data in the descriptor in case header information needs to be added as the packet is passed thorough the system.
15-14	RESERVED	R	0h	Reserved
13-12	RX_DEST_QMGR	W	0h	Rx destination queue manager. This field indicates the default receive queue manager that this channel should use.
11-0	RX_DEST_QNUM	W	0h	Rx destination queue. This field indicates the default receive queue that packets on this flow should be placed onto.

Table 11-2482. Register Call Summary for CDMA_RX_FLOW_CONFIG_REG_A_0

Navigator Subsystem (NAVSS) • Receive Flow Configuration: [0] • Descriptor Starvation: [0]
NAVSS Registers • NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG Registers: [0] • CDMA_RX_FLOW_CONFIG_REG_A_0 to CDMA_RX_FLOW_CONFIG_REG_A_31 Register (Offset = 0h + [i * 20h], where i = 0 to 31) [reset = 0h]: [0]

11.13.5.2.3.5.2 CDMA_RX_FLOW_CONFIG_REG_B_0 to CDMA_RX_FLOW_CONFIG_REG_B_31 Register (Offset = 4h + [i * 20h], where i = 0 to 31) [reset = 0h]

CDMA_RX_FLOW_CONFIG_REG_B_0 to CDMA_RX_FLOW_CONFIG_REG_B_31 is shown in Figure 11-1065 and described in Table 11-2484.

Table 11-2483. CDMA_RX_FLOW_CONFIG_REG_B_0 to CDMA_RX_FLOW_CONFIG_REG_B_31 Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG	0401 3004h + [i * 20h], where i = 0 to 31

Figure 11-1065. CDMA_RX_FLOW_CONFIG_REG_B_0 to CDMA_RX_FLOW_CONFIG_REG_B_31 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RX_SRC_TAG_HI								RX_SRC_TAG_LO							
R-0h								R-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX_DEST_TAG_HI								RX_DEST_TAG_LO							
R-0h								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2484. CDMA_RX_FLOW_CONFIG_REG_B_0 to CDMA_RX_FLOW_CONFIG_REG_B_31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RX_SRC_TAG_HI	R	0h	Rx source tag high byte constant value. This is the value to insert into bits 15:8 of the source tag if the RX_SRC_TAG_LO_SEL is set to 1.
23-16	RX_SRC_TAG_LO	R	0h	Rx source tag low byte constant value. This is the value to insert into bits 7:0 of the source tag if the RX_SRC_TAG_LO_SEL is set to 1.
15-8	RX_DEST_TAG_HI	R	0h	Rx destination tag high byte constant value. This is the value to insert into bits 15:8 of the destination tag if the RX_DEST_TAG_HI_SEL is set to 1.
7-0	RX_DEST_TAG_LO	R	0h	Rx destination tag low byte constant value. This is the value to insert into bits 7:0 of the destination tag if the RX_DEST_TAG_LO_SEL is set to 1.

Table 11-2485. Register Call Summary for CDMA_RX_FLOW_CONFIG_REG_B_0

Navigator Subsystem (NAVSS) • Receive Flow Configuration: [0]
NAVSS Registers • NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG Registers: [0] • CDMA_RX_FLOW_CONFIG_REG_B_0 to CDMA_RX_FLOW_CONFIG_REG_B_31 Register (Offset = 4h + [i * 20h], where i = 0 to 31) [reset = 0h]: [0]

11.13.5.2.3.5.3 CDMA_RX_FLOW_CONFIG_REG_C_0 to CDMA_RX_FLOW_CONFIG_REG_C_31 Register (Offset = 8h + [i * 20h], where i = 0 to 31) [reset = 0h]

CDMA_RX_FLOW_CONFIG_REG_C_0 to CDMA_RX_FLOW_CONFIG_REG_C_31 is shown in Figure 11-1066 and described in Table 11-2487.

Table 11-2486. CDMA_RX_FLOW_CONFIG_REG_C_0 to CDMA_RX_FLOW_CONFIG_REG_C_31 Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG	0401 3008h + [i * 20h], where i = 0 to 31

Figure 11-1066. CDMA_RX_FLOW_CONFIG_REG_C_0 to CDMA_RX_FLOW_CONFIG_REG_C_31 Register

31	30	29	28	27	26	25	24
RESERVED	RX_SRC_TAG_HI_SEL			RESERVED	RX_SRC_TAG_LO_SEL		
R-0h	W-0h			R-0h	W-0h		
23	22	21	20	19	18	17	16
RESERVED	RX_DEST_TAG_HI_SEL			RESERVED	RX_DEST_TAG_LO_SEL		
R-0	W-0h			R-0h	W-0h		
15	14	13	12	11	10	9	8
RESERVED							
R-0							
7	6	5	4	3	2	1	0
RESERVED				RX_SIZE_THRESH_EN			
R-0h				W-0h			

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 11-2487. CDMA_RX_FLOW_CONFIG_REG_C_0 to CDMA_RX_FLOW_CONFIG_REG_C_31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30-28	RX_SRC_TAG_HI_SEL	W	0h	Rx source tag high byte selector. This field specifies the source for bits 7:0 of the source tag field in the output packet descriptor. • 000 = do not overwrite • 001 = overwrite with value given in RX_SRC_TAG_HI • 010 = overwrite with FLOW_ID[7:0] from back end application • 011 = overwrite with FLOW_ID[15:8] from back end application • 100 = overwrite with SRC_TAG[7:0] from back end application • 101 = overwrite with SRC_TAG[15:8] from back end application • 110-111 = RESERVED
27	RESERVED	R	0h	Reserved
26-24	RX_SRC_TAG_LO_SEL	W	0h	Rx source tag low byte selector. This field specifies the source for bits 7:0 of the source tag field in the output packet descriptor. • 000 = do not overwrite • 001 = overwrite with value given in RX_SRC_TAG_LO • 010 = overwrite with FLOW_ID[7:0] from back end application • 011 = overwrite with FLOW_ID[15:8] from back end application • 100 = overwrite with SRC_TAG[7:0] from back end application • 101 = overwrite with SRC_TAG[15:8] from back end application • 110-111 = RESERVED
23	RESERVED	R	0h	Reserved

Table 11-2487. CDMA_RX_FLOW_CONFIG_REG_C_0 to CDMA_RX_FLOW_CONFIG_REG_C_31 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22-20	RX_DEST_TAG_HI_SEL	W	0h	Rx destination tag high byte selector. This field specifies the source for bits 15:8 of the source tag field in the word 1 of the output PD. • 000 = do not overwrite • 001 = overwrite with value given in RX_DEST_TAG_HI • 010 = overwrite with FLOW_ID[7:0] from back end application • 011 = overwrite with FLOW_ID[15:8] from back end application • 100 = overwrite with DEST_TAG[7:0] from back end application • 101 = overwrite with DEST_TAG[15:8] from back end application • 110-111 = RESERVED
19	RESERVED	R	0h	Reserved
18-16	RX_DEST_TAG_LO_SEL	W	0h	Rx destination tag low byte selector. This field specifies the source for bits 7:0 of the source tag field in word 1 of the output PD. • 000 = do not overwrite • 001 = overwrite with value given in RX_DEST_TAG_LO • 010 = overwrite with FLOW_ID[7:0] from back end application • 011 = overwrite with FLOW_ID[15:8] from back end application • 100 = overwrite with DEST_TAG[7:0] from back end application • 101 = overwrite with DEST_TAG[15:8] from back end application • 110-111 = RESERVED
15-4	RESERVED	R	0h	Reserved
3-0	RX_SIZE_THRESH_EN	W	0h	Rx packet sized based free buffer queue enables. These bits control whether or not the flow will compare the packet size received from the back end application against the RX_SIZE_THRESHN fields to determine which FDQ to allocate the SOP buffer from. Each bit in this field corresponds to 1 of the 4 potential size thresholds that can be compared against. Bit 0 corresponds to RX_SIZE_THRESH0 and bit 3 corresponds to RX_SIZE_THRESH3. • 000 = do not use the threshold • 001 = Use the thresholds to select between the 4 different potential SOP FDQs. If none of the thresholds are enabled, the DMA controller in the port will allocate the SOP buffer from the queue specified by the RX_FDQ0_SZ0_QMGR and RX_FDQ0_SZ0_QNUM fields. Support for packet size based FDQ selection is OPTIONAL. If the port does not implement this feature, the bits of this field will be hardcoded to 0 and will not be writable by the host.

Table 11-2488. Register Call Summary for CDMA_RX_FLOW_CONFIG_REG_C_0

Navigator Subsystem (NAVSS) • Receive Flow Configuration: [0]
NAVSS Registers • NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG Registers: [0] • CDMA_RX_FLOW_CONFIG_REG_C_0 to CDMA_RX_FLOW_CONFIG_REG_C_31 Register (Offset = 8h + [i * 20h], where i = 0 to 31) [reset = 0h]: [0]

11.13.5.2.3.5.4 CDMA_RX_FLOW_CONFIG_REG_D_0 to CDMA_RX_FLOW_CONFIG_REG_D_31 Register (Offset = Ch + [i * 20h], where i = 0 to 31) [reset = 0h]

CDMA_RX_FLOW_CONFIG_REG_D_0 to CDMA_RX_FLOW_CONFIG_REG_D_31 is shown in Figure 11-1067 and described in Table 11-2490.

Table 11-2489. CDMA_RX_FLOW_CONFIG_REG_D_0 to CDMA_RX_FLOW_CONFIG_REG_D_31 Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG	0401 300Ch + [i * 20h], where i = 0 to 31

Figure 11-1067. CDMA_RX_FLOW_CONFIG_REG_D_0 to CDMA_RX_FLOW_CONFIG_REG_D_31 Register

31	30	29	28	27	26	25	24
RESERVED		RX_FDQ0_SZ0_QMGR		RX_FDQ0_SZ0_QNUM			
R-0h		W-0h		W-0h			
23	22	21	20	19	18	17	16
RX_FDQ0_SZ0_QNUM							
W-0h							
15	14	13	12	11	10	9	8
RESERVED		RX_FDQ1_QMGR		RX_FDQ1_QNUM			
R-0h		W-0h		W-0h			
7	6	5	4	3	2	1	0
RX_FDQ1_QNUM							
W-0h							

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 11-2490. CDMA_RX_FLOW_CONFIG_REG_D_0 to CDMA_RX_FLOW_CONFIG_REG_D_31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29-28	RX_FDQ0_SZ0_QMGR	W	0h	Rx free descriptor 0 queue manager index - size 0. This field specifies which Queue Manager should be used for the 1st Rx buffer in a packet whose size is less than or equal to the RX_SIZE0 value.
27-16	RX_FDQ0_SZ0_QNUM	W	0h	Rx free descriptor 0 queue index - size 0. This field specifies which Free Descriptor Queue should be used for the 1st Rx buffer in a packet whose size is less than or equal to the RX_SIZE0 value.
15-14	RESERVED	R	0h	Reserved
13-12	RX_FDQ1_QMGR	W	0h	Rx free descriptor 1 queue manager index. This field specifies which Queue Manager should be used for the 2nd Rx buffer in a host type packet.
11-0	RX_FDQ1_QNUM	W	0h	Rx free descriptor 1 queue index. This field specifies which Free Descriptor Queue should be used for the 2nd Rx buffer in a host type packet.

Table 11-2491. Register Call Summary for CDMA_RX_FLOW_CONFIG_REG_D_0

Navigator Subsystem (NAVSS) • Receive Flow Configuration: [0]
NAVSS Registers • NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG Registers: [0] • CDMA_RX_FLOW_CONFIG_REG_D_0 to CDMA_RX_FLOW_CONFIG_REG_D_31 Register (Offset = Ch + [i * 20h], where i = 0 to 31) [reset = 0h]: [0]

11.13.5.2.3.5.5 CDMA_RX_FLOW_CONFIG_REG_E_0 to CDMA_RX_FLOW_CONFIG_REG_E_31 Register (Offset = 10h + [i * 20h], where i = 0 to 31) [reset = 0h]

CDMA_RX_FLOW_CONFIG_REG_E_0 to CDMA_RX_FLOW_CONFIG_REG_E_31 is shown in Figure 11-1068 and described in Table 11-2493.

Table 11-2492. CDMA_RX_FLOW_CONFIG_REG_E_0 to CDMA_RX_FLOW_CONFIG_REG_E_31 Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG	0401 3010h + [i * 20h], where i = 0 to 31

Figure 11-1068. CDMA_RX_FLOW_CONFIG_REG_E_0 to CDMA_RX_FLOW_CONFIG_REG_E_31 Register

31	30	29	28	27	26	25	24
RESERVED		RX_FDQ2_QMGR		RX_FDQ2_QNUM			
R-0h		W-0h		W-0h			
23	22	21	20	19	18	17	16
RX_FDQ2_QNUM							
W-0h							
15	14	13	12	11	10	9	8
RESERVED		RX_FDQ3_QMGR		RX_FDQ3_QNUM			
R-0h		W-0h		W-0h			
7	6	5	4	3	2	1	0
RX_FDQ3_QNUM							
W-0h							

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 11-2493. CDMA_RX_FLOW_CONFIG_REG_E_0 to CDMA_RX_FLOW_CONFIG_REG_E_31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29-28	RX_FDQ2_QMGR	W	0h	Rx free descriptor 2 queue manager index. This field specifies which Queue Manager should be used for the 3rd Rx buffer in a host type packet.
27-16	RX_FDQ2_QNUM	W	0h	Rx free descriptor 2 queue index. This field specifies which Free Descriptor Queue should be used for the 3rd Rx buffer in a host type packet
15-14	RESERVED	R	0h	Reserved
13-12	RX_FDQ3_QMGR	W	0h	Rx free descriptor 3 queue manager index. This field specifies which Queue Manager should be used for the 4th or later Rx buffers in a host type packet
11-0	RX_FDQ3_QNUM	W	0h	Rx free descriptor 3 queue index. This field specifies which Free Descriptor Queue should be used for the 4th or later Rx buffers in a host type packet

Table 11-2494. Register Call Summary for CDMA_RX_FLOW_CONFIG_REG_E_0

Navigator Subsystem (NAVSS) • Receive Flow Configuration: [0]
NAVSS Registers • NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG Registers: [0] • CDMA_RX_FLOW_CONFIG_REG_E_0 to CDMA_RX_FLOW_CONFIG_REG_E_31 Register (Offset = 10h + [i * 20h], where i = 0 to 31) [reset = 0h]: [0]

11.13.5.2.3.5.6 CDMA_RX_FLOW_CONFIG_REG_F_0 to CDMA_RX_FLOW_CONFIG_REG_F_31 Register (Offset = 14h + [i * 20h], where i = 0 to 31) [reset = 0h]

CDMA_RX_FLOW_CONFIG_REG_F_0 to CDMA_RX_FLOW_CONFIG_REG_F_31 is shown in Figure 11-1069 and described in Table 11-2496.

Table 11-2495. CDMA_RX_FLOW_CONFIG_REG_F_0 to CDMA_RX_FLOW_CONFIG_REG_F_31 Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG	0401 3014h + [i * 20h], where i = 0 to 31

Figure 11-1069. CDMA_RX_FLOW_CONFIG_REG_F_0 to CDMA_RX_FLOW_CONFIG_REG_F_31 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX_SIZE_THRESH0																RX_SIZE_THRESH1															
W-0h																W-0h															

LEGEND: W = Write Only; *-n* = value after reset

Table 11-2496. CDMA_RX_FLOW_CONFIG_REG_F_0 to CDMA_RX_FLOW_CONFIG_REG_F_31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RX_SIZE_THRESH0	W	0h	Rx packet size threshold 0. This value is left shifted by 5 bits and compared against the packet size to determine which free descriptor queue should be used for the SOP buffer in the packet. If the packet size is less than or equal to the value given in this threshold, the DMA controller in the port will allocate the SOP buffer from the queue given by the RX_FDQ0_SZ0_QMGR and RX_FDQ0_SZ0_QNUM fields. This field is OPTIONAL.
15-0	RX_SIZE_THRESH1	W	0h	Rx packet size threshold 1. This value is left shifted by 5 bits and compared against the packet size to determine which free descriptor queue should be used for the SOP buffer in the packet. If the packet size is greater than theRX_SIZE_THRESH0 but is less than or equal to the value given in this threshold, the DMA controller in the port will allocate the SOP buffer from the queue given by the RX_FDQ0_SZ1_QMGR and RX_FDQ0_SZ1_QNUM fields. If enabled, this value must be greater than the value given in theRX_SIZE_THRESH0 field. This field is optional.

Table 11-2497. Register Call Summary for CDMA_RX_FLOW_CONFIG_REG_F_0

Navigator Subsystem (NAVSS) • Receive Flow Configuration: [0]
NAVSS Registers • NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG Registers: [0] • CDMA_RX_FLOW_CONFIG_REG_F_0 to CDMA_RX_FLOW_CONFIG_REG_F_31 Register (Offset = 14h + [i * 20h], where i = 0 to 31) [reset = 0h]: [0]

11.13.5.2.3.5.7 CDMA_RX_FLOW_CONFIG_REG_G_0 to CDMA_RX_FLOW_CONFIG_REG_G_31 Register (Offset = 18h + [i * 20h], where i = 0 to 31) [reset = 0h]

CDMA_RX_FLOW_CONFIG_REG_G_0 to CDMA_RX_FLOW_CONFIG_REG_G_31 is shown in Figure 11-1070 and described in Table 11-2499.

Table 11-2498. CDMA_RX_FLOW_CONFIG_REG_G_0 to CDMA_RX_FLOW_CONFIG_REG_G_31 Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG	0401 3018h + [i * 20h], where i = 0 to 31

Figure 11-1070. CDMA_RX_FLOW_CONFIG_REG_G_0 to CDMA_RX_FLOW_CONFIG_REG_G_31 Register

31	30	29	28	27	26	25	24
RX_SIZE_THRESH2							
W-0h							
23	22	21	20	19	18	17	16
RX_SIZE_THRESH2							
W-0h							
15	14	13	12	11	10	9	8
RESERVED		RX_FDQ0_SZ1_QMGR		RX_FDQ0_SZ1_QNUM			
R-0h		W-0h		W-0h			
7	6	5	4	3	2	1	0
RX_FDQ0_SZ1_QNUM							
W-0h							

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 11-2499. CDMA_RX_FLOW_CONFIG_REG_G_0 to CDMA_RX_FLOW_CONFIG_REG_G_31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RX_SIZE_THRESH2	W	0h	Rx packet size threshold 2. This value is left shifted by 5 bits and compared against the packet size to determine which free descriptor queue should be used for the SOP buffer in the packet. If the packet size is less than or equal to the value given in this threshold, the DMA controller in the port will allocate the SOP buffer from the queue given by the RX_FDQ0_SZ2_QMGR and RX_FDQ0_SZ2_QNUM fields. If enabled, this value must be greater than the value given in the RX_SIZE_THRESH1 field. This field is optional.
15-14	RESERVED	R	0h	Reserved
13-12	RX_FDQ0_SZ1_QMGR	W	0h	Rx free descriptor 0 queue manager index - size 1. This field specifies which Queue Manager should be used for the 1st Rx buffer in a packet whose size is less than or equal to the RX_SIZE0 value. This field is optional.
11-0	RX_FDQ0_SZ1_QNUM	W	0h	Rx free descriptor 0 queue index - size 1. This field specifies which Free Descriptor Queue should be used for the 1st Rx buffer in a packet whose size is less than or equal to the RX_SIZE0 value. This field is optional.

Table 11-2500. Register Call Summary for CDMA_RX_FLOW_CONFIG_REG_G_0

Navigator Subsystem (NAVSS) • Receive Flow Configuration: [0]
NAVSS Registers • NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG Registers: [0] • CDMA_RX_FLOW_CONFIG_REG_G_0 to CDMA_RX_FLOW_CONFIG_REG_G_31 Register (Offset = 18h + [i * 20h], where i = 0 to 31) [reset = 0h]: [0]

11.13.5.2.3.5.8 CDMA_RX_FLOW_CONFIG_REG_H_0 to CDMA_RX_FLOW_CONFIG_REG_H_31 Register (Offset = 1Ch + [i * 20h], where i = 0 to 31) [reset = 0h]

CDMA_RX_FLOW_CONFIG_REG_H_0 to CDMA_RX_FLOW_CONFIG_REG_H_31 is shown in Figure 11-1071 and described in Table 11-2502.

Table 11-2501. CDMA_RX_FLOW_CONFIG_REG_H_0 to CDMA_RX_FLOW_CONFIG_REG_H_31 Instances

Instance	Physical Address
NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG	0401 301Ch + [i * 20h], where i = 0 to 31

Figure 11-1071. CDMA_RX_FLOW_CONFIG_REG_H_0 to CDMA_RX_FLOW_CONFIG_REG_H_31 Register

31	30	29	28	27	26	25	24
RESERVED		RX_FDQ0_SZ2_QMGR		RX_FDQ0_SZ2_QNUM			
R-0h		W-0h		W-0h			
23	22	21	20	19	18	17	16
RX_FDQ0_SZ2_QNUM							
W-0h							
15	14	13	12	11	10	9	8
RESERVED		RX_FDQ0_SZ3_QMGR		RX_FDQ0_SZ3_QNUM			
R-0h		W-0h		W-0h			
7	6	5	4	3	2	1	0
RX_FDQ0_SZ3_QNUM							
W-0h							

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 11-2502. CDMA_RX_FLOW_CONFIG_REG_H_0 to CDMA_RX_FLOW_CONFIG_REG_H_31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29-28	RX_FDQ0_SZ2_QMGR	W	0h	Rx free descriptor 0 queue manager index - size 2. This field specifies which Queue Manager should be used for the 1st Rx buffer in a packet whose size is less than or equal to the RX_SIZE2 value. This field is optional.
27-16	RX_FDQ0_SZ2_QNUM	W	0h	Rx free descriptor 0 queue index - size 2. This field specifies which Free Descriptor Queue should be used for the 1st Rx buffer in a packet whose size is less than or equal to the RX_SIZE0 value. This field is optional.
15-14	RESERVED	R	0h	Reserved
13-12	RX_FDQ0_SZ3_QMGR	W	0h	Rx free descriptor 0 queue manager index - size 3. This field specifies which Queue Manager should be used for the 1st Rx buffer in a packet whose size was not less than or equal to the RX_SIZE3 value. This field is optional
11-0	RX_FDQ0_SZ3_QNUM	W	0h	Rx free descriptor 0 queue index - size 3. This field specifies which Free Descriptor Queue should be used for the 1st Rx buffer in a packet whose size is less than or equal to the RX_SIZE0 value. This field is optional.

Table 11-2503. Register Call Summary for CDMA_RX_FLOW_CONFIG_REG_H_0

Navigator Subsystem (NAVSS) • Receive Flow Configuration: [0]
NAVSS Registers • NSS_0_CFG_CPPIDMA0_RX_FLOW_CFG Registers: [0] • CDMA_RX_FLOW_CONFIG_REG_H_0 to CDMA_RX_FLOW_CONFIG_REG_H_31 Register (Offset = 1Ch + [i * 20h], where i = 0 to 31) [reset = 0h]: [0]

11.13.5.2.4 QM Registers

11.13.5.2.4.1 NSS_0_CFG_QMGR0_CFG Registers

Table 11-2505 lists the memory-mapped registers for the NSS_0_CFG_QMGR0_CFG. All register offset addresses not listed in Table 11-2505 should be considered as reserved locations and the register contents should not be modified.

Table 11-2504. NSS_0_CFG_QMGR0_CFG Instances

Instance	Base Address
NSS_0_CFG_QMGR0_CFG	0404 0000h

Table 11-2505. NSS_0_CFG_QMGR0_CFG Registers

Offset	Acronym	Register Name	NSS_0_CFG_QMGR0_CFG Physical Address	Section
0h	QM_REVISION_REG	Revision Register	0404 0000h	Section 11.13.5.2.4.1.1
8h	QM_QUEUE_DIVERSION_REG	Queue Diversion Register	0404 0008h	Section 11.13.5.2.4.1.2
Ch	QM_LINKING_RAM_REGION_0_BASE_ADDRESS_REG	Linking RAM Region 0 Base Address Register	0404 000Ch	Section 11.13.5.2.4.1.3
10h	QM_LINKING_RAM_REGION_0_SIZE_REG	Linking RAM Region 0 Size Register	0404 0010h	Section 11.13.5.2.4.1.4
14h	QM_LINKING_RAM_REGION_1_BASE_ADDRESS_REG	Linking RAM Region 1 Base Address Register	0404 0014h	Section 11.13.5.2.4.1.5
20h to 5Ch	QM_FREE_DESCRIPTOR_STARVE_COUNT_REG_0 to QM_FREE_DESCRIPTOR_STARVE_COUNT_REG_15	Free Descriptor/Buffer Starvation Count Register 0 to 15	0404 0014h to 0404 0000h	Section 11.13.5.2.4.1.6

11.13.5.2.4.1.1 QM_REVISION_REG Register (Offset = 0h) [reset = 8EFC1900h]

QM_REVISION_REG is shown in Figure 11-1072 and described in Table 11-2507.

Table 11-2506. QM_REVISION_REG Instances

Instance	Physical Address
NSS_0_CFG_QMGR0_CFG	0404 0000h

Figure 11-1072. QM_REVISION_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 8EFC1900h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2507. QM_REVISION_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	8EFC1900h	TI internal data. Identifies revision of peripheral.

Table 11-2508. Register Call Summary for QM_REVISION_REG

NAVSS Registers
- NSS_0_CFG_QMGR0_CFG Registers: [0]
- QM_REVISION_REG Register (Offset = 0h) [reset = 8EFC1900h]: [0]

11.13.5.2.4.1.2 QM_QUEUE_DIVERSION_REG Register (Offset = 8h) [reset = 0h]

QM_QUEUE_DIVERSION_REG is shown in Figure 11-1073 and described in Table 11-2510.

Table 11-2509. QM_QUEUE_DIVERSION_REG Instances

Instance	Physical Address
NSS_0_CFG_QMGR0_CFG	0404 0008h

Figure 11-1073. QM_QUEUE_DIVERSION_REG Register

31	30	29	28	27	26	25	24
HEAD_TAIL	RESERVED	DEST_QNUM					
W-0h	R-0h	W-0h					
23	22	21	20	19	18	17	16
DEST_QNUM							
W-0h							
15	14	13	12	11	10	9	8
RESERVED		SOURCE_QNUM					
R-0h		W-0h					
7	6	5	4	3	2	1	0
SOURCE_QNUM							
W-0h							

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 11-2510. QM_QUEUE_DIVERSION_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	HEAD_TAIL	W	0h	Indicates whether queue contents should be merged on to head or tail of destination queue. Clear this field for head and set for tail.
30	RESERVED	R	0h	Reserved
29-16	DEST_QNUM	W	0h	Destination queue number
15-14	RESERVED	R	0h	Reserved
13-0	SOURCE_QNUM	W	0h	Source queue number

Table 11-2511. Register Call Summary for QM_QUEUE_DIVERSION_REG

Navigator Subsystem (NAVSS) • Diverting Queued Packets from One Queue to Another: [0]
NAVSS Registers • NSS_0_CFG_QMGR0_CFG Registers: [0] • QM_QUEUE_DIVERSION_REG Register (Offset = 8h) [reset = 0h]: [0]

11.13.5.2.4.1.3 QM_LINKING_RAM_REGION_0_BASE_ADDRESS_REG Register (Offset = Ch) [reset = 0h]

QM_LINKING_RAM_REGION_0_BASE_ADDRESS_REG is shown in Figure 11-1074 and described in Table 11-2513.

Table 11-2512. QM_LINKING_RAM_REGION_0_BASE_ADDRESS_REG Instances

Instance	Physical Address
NSS_0_CFG_QMGR0_CFG	0404 000Ch

Figure 11-1074. QM_LINKING_RAM_REGION_0_BASE_ADDRESS_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REGION0_BASE																														RESERVED	
R/W-0h																														R/W-0h	

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2513. QM_LINKING_RAM_REGION_0_BASE_ADDRESS_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	REGION0_BASE	R/W	0h	This field stores the base address for the first region of the linking RAM. This may be anywhere in 32-bit address space but would be typically located in on-chip memory.
1-0	RESERVED	R/W	0h	Reserved

Table 11-2514. Register Call Summary for QM_LINKING_RAM_REGION_0_BASE_ADDRESS_REG

NAVSS Registers
- NSS_0_CFG_QMGR0_CFG Registers: [0]
- QM_LINKING_RAM_REGION_0_BASE_ADDRESS_REG Register (Offset = Ch) [reset = 0h]: [0]

11.13.5.2.4.1.4 QM_LINKING_RAM_REGION_0_SIZE_REG Register (Offset = 10h) [reset = 0h]

QM_LINKING_RAM_REGION_0_SIZE_REG is shown in Figure 11-1075 and described in Table 11-2516.

Table 11-2515. QM_LINKING_RAM_REGION_0_SIZE_REG Instances

Instance	Physical Address
NSS_0_CFG_QMGR0_CFG	0404 0010h

Figure 11-1075. QM_LINKING_RAM_REGION_0_SIZE_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													REGION0_SIZE																		
R-0h													R/W-0h																		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2516. QM_LINKING_RAM_REGION_0_SIZE_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18-0	REGION0_SIZE	R/W	0h	This field indicates the number of entries that are contained in the linking RAM region 0. A descriptor with index less than REGION0_SIZE value has its linking location in region 0. A descriptor with index greater than REGION0_SIZE has its linking location in region 1. The queue manager will add the index (left shifted by 3 bits) to the appropriate REGIONX_BASE_ADDR to get the absolute 32-bit address to the linking location for a descriptor.

Table 11-2517. Register Call Summary for QM_LINKING_RAM_REGION_0_SIZE_REG

NAVSS Registers
- NSS_0_CFG_QMGR0_CFG Registers: [0]
- QM_LINKING_RAM_REGION_0_SIZE_REG Register (Offset = 10h) [reset = 0h]: [0]

11.13.5.2.4.1.5 QM_LINKING_RAM_REGION_1_BASE_ADDRESS_REG Register (Offset = 14h) [reset = 0h]

QM_LINKING_RAM_REGION_1_BASE_ADDRESS_REG is shown in Figure 11-1076 and described in Table 11-2519.

Table 11-2518. QM_LINKING_RAM_REGION_1_BASE_ADDRESS_REG Instances

Instance	Physical Address
NSS_0_CFG_QMGR0_CFG	0404 0014h

Figure 11-1076. QM_LINKING_RAM_REGION_1_BASE_ADDRESS_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REGION1_BASE																														RESERVED	
R/W-0h																														R/W-0h	

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2519. QM_LINKING_RAM_REGION_1_BASE_ADDRESS_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	REGION1_BASE	R/W	0h	This field stores the base address for the first region of the linking RAM. This may be anywhere in 32-bit address space but would be typically located in on-chip memory.
1-0	RESERVED	R/W	0h	Reserved

Table 11-2520. Register Call Summary for QM_LINKING_RAM_REGION_1_BASE_ADDRESS_REG

NAVSS Registers
- NSS_0_CFG_QMGR0_CFG Registers: [0]
- QM_LINKING_RAM_REGION_1_BASE_ADDRESS_REG Register (Offset = 14h) [reset = 0h]: [0]

11.13.5.2.4.1.6 QM_FREE_DESCRIPTOR_STARVE_COUNT_REG_0 to QM_FREE_DESCRIPTOR_STARVE_COUNT_REG_15 Register (Offset = 20h to 5Ch) [reset = 0h]

QM_FREE_DESCRIPTOR_STARVE_COUNT_REG_0 to QM_FREE_DESCRIPTOR_STARVE_COUNT_REG_15 is shown in Figure 11-1077 and described in Table 11-2522.

Table 11-2521. QM_FREE_DESCRIPTOR_STARVE_COUNT_REG_0 to QM_FREE_DESCRIPTOR_STARVE_COUNT_REG_15 Instances

Instance	Physical Address
NSS_0_CFG_QMGR0_CFG	0404 0020h to 0404 005Ch

Figure 11-1077. QM_FREE_DESCRIPTOR_STARVE_COUNT_REG_0 to QM_FREE_DESCRIPTOR_STARVE_COUNT_REG_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FDBQ3_STARVE_CNT								FDBQ2_STARVE_CNT							
COR-0h								COR-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FDBQ1_STARVE_CNT								FDBQ0_STARVE_CNT							
COR-0h								COR-0h							

LEGEND: COR = Cleared on Read; -*n* = value after reset

Table 11-2522. QM_FREE_DESCRIPTOR_STARVE_COUNT_REG_0 to QM_FREE_DESCRIPTOR_STARVE_COUNT_REG_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	FDBQ3_STARVE_CNT	COR	0h	This field increments each time the Free Descriptor/Buffer Queue \[(N*4) + 3\]==3 is read while it is empty. This field is cleared when read.
23-16	FDBQ2_STARVE_CNT	COR	0h	This field increments each time the Free Descriptor/Buffer Queue \[(N*4) + 2\]==2 is read while it is empty. This field is cleared when read.
15-8	FDBQ1_STARVE_CNT	COR	0h	This field increments each time the Free Descriptor/Buffer Queue \[(N*4) +1\]==1 is read while it is empty. This field is cleared when read.
7-0	FDBQ0_STARVE_CNT	COR	0h	This field increments each time the Free Descriptor/Buffer Queue \[(N*4) \]==0 is read while it is empty. This field is cleared when read.

Table 11-2523. Register Call Summary for QM_FREE_DESCRIPTOR_STARVE_COUNT_REG_0

NAVSS Registers

- NSS_0_CFG_QMGR0_CFG Registers: [0]
- QM_FREE_DESCRIPTOR_STARVE_COUNT_REG_0 to QM_FREE_DESCRIPTOR_STARVE_COUNT_REG_15 Register (Offset = 20h to 5Ch) [reset = 0h]: [0]

11.13.5.2.4.2 NSS_0_CFG_QMGR0_DESCRIPTOR_REGIONS Registers

Table 11-2525 lists the memory-mapped registers for the NSS_0_CFG_QMGR0_DESCRIPTOR_REGIONS. All register offset addresses not listed in Table 11-2525 should be considered as reserved locations and the register contents should not be modified.

Table 11-2524. NSS_0_CFG_QMGR0_DESCRIPTOR_REGIONS Instances

Instance	Base Address
NSS_0_CFG_QMGR0_DESCRIPTOR_REGIONS	0408 0000h

Table 11-2525. NSS_0_CFG_QMGR0_DESCRIPTOR_REGIONS Registers

Offset	Acronym	Register Name	NSS_0_CFG_QMGR0_DESCRIPTOR_REGIONS Physical Address	Section
0h + (i * 10h) [1]	QM_MEMORY_REGION_BASE_ADDRESS_REG_0	Memory Region i Base Address Register	0408 0000h + (i * 10h)	Section 11.13.5.2.4.2.1
4h + (i * 10h)	QM_MEMORY_REGION_START_INDEX_REG_0	Memory Region i Start Index Register	0408 0004h + (i * 10h)	Section 11.13.5.2.4.2.2
8h (i * 10h)	QM_MEMORY_REGION_DESCRIPTOR_SETUP_REG_0	Memory Region i Descriptor Setup Register	0408 0008h + (i * 10h)	Section 11.13.5.2.4.2.3

(1) i = 0 to 15

11.13.5.2.4.2.1 QM_MEMORY_REGION_BASE_ADDRESS_REG_0 to QM_MEMORY_REGION_BASE_ADDRESS_REG_15 Register (Offset = 0h + [i * 10h], where i = 0 to 15) [reset = 0h]

QM_MEMORY_REGION_BASE_ADDRESS_REG_0 to QM_MEMORY_REGION_BASE_ADDRESS_REG_15 is shown in Figure 11-1078 and described in Table 11-2527.

Table 11-2526. QM_MEMORY_REGION_BASE_ADDRESS_REG_0 to QM_MEMORY_REGION_BASE_ADDRESS_REG_15 Instances

Instance	Physical Address
NSS_0_CFG_QMGR0_DESCRIPTOR_REGIONS	0408 0000h + [i * 10h], where i = 0 to 15

Figure 11-1078. QM_MEMORY_REGION_BASE_ADDRESS_REG_0 to QM_MEMORY_REGION_BASE_ADDRESS_REG_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REGR_BASE																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2527. QM_MEMORY_REGION_BASE_ADDRESS_REG_0 to QM_MEMORY_REGION_BASE_ADDRESS_REG_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REGR_BASE	R/W	0h	This field contains the base address of the memory region R.

Table 11-2528. Register Call Summary for QM_MEMORY_REGION_BASE_ADDRESS_REG_0

NAVSS Registers

- QM_MEMORY_REGION_BASE_ADDRESS_REG_0 to QM_MEMORY_REGION_BASE_ADDRESS_REG_15 Register (Offset = 0h + [i * 10h], where i = 0 to 15) [reset = 0h]: [0]
- NSS_0_CFG_QMGR0_DESCRIPTOR_REGIONS Registers: [0]

11.13.5.2.4.2.2 QM_MEMORY_REGION_START_INDEX_REG_0 to QM_MEMORY_REGION_START_INDEX_REG_15 Register (Offset = 4h + [i * 10h], where i = 0 to 15) [reset = 0h]

QM_MEMORY_REGION_START_INDEX_REG_0 to QM_MEMORY_REGION_START_INDEX_REG_15 is shown in Figure 11-1079 and described in Table 11-2530.

Table 11-2529. QM_MEMORY_REGION_START_INDEX_REG_0 to QM_MEMORY_REGION_START_INDEX_REG_15 Instances

Instance	Physical Address
NSS_0_CFG_QMGR0_DESCRIPTOR_REGIONS	0408 0004h + [i * 10h], where i = 0 to 15

Figure 11-1079. QM_MEMORY_REGION_START_INDEX_REG_0 to QM_MEMORY_REGION_START_INDEX_REG_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													START_INDEX																		
R-0h													R/W-0h																		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2530. QM_MEMORY_REGION_START_INDEX_REG_0 to QM_MEMORY_REGION_START_INDEX_REG_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18-0	START_INDEX	R/W	0h	This field indicates where in linking RAM does the descriptor linking information corresponding to memory region R starts.

Table 11-2531. Register Call Summary for QM_MEMORY_REGION_START_INDEX_REG_0

NAVSS Registers

- QM_MEMORY_REGION_START_INDEX_REG_0 to QM_MEMORY_REGION_START_INDEX_REG_15 Register (Offset = 4h + [i * 10h], where i = 0 to 15) [reset = 0h]: [0]
- NSS_0_CFG_QMGR0_DESCRIPTOR_REGIONS Registers: [0]

11.13.5.2.4.2.3 QM_MEMORY_REGION_DESCRIPTOR_SETUP_REG_0 to QM_MEMORY_REGION_DESCRIPTOR_SETUP_REG_15 Register (Offset = 8h + [i * 10h], where i = 0 to 15) [reset = 0h]

QM_MEMORY_REGION_DESCRIPTOR_SETUP_REG_0 to QM_MEMORY_REGION_DESCRIPTOR_SETUP_REG_15 is shown in Figure 11-1080 and described in Table 11-2533.

Table 11-2532. QM_MEMORY_REGION_DESCRIPTOR_SETUP_REG_0 to QM_MEMORY_REGION_DESCRIPTOR_SETUP_REG_15 Instances

Instance	Physical Address
NSS_0_CFG_QMGR0_DESCRIPTOR_REGIONS	0408 0008h + [i * 10h], where i = 0 to 15

Figure 11-1080. QM_MEMORY_REGION_DESCRIPTOR_SETUP_REG_0 to QM_MEMORY_REGION_DESCRIPTOR_SETUP_REG_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED			DESC_SIZE												
R-0h			R/W-0h												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												REG_SIZE			
R-0h												R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2533. QM_MEMORY_REGION_DESCRIPTOR_SETUP_REG_0 to QM_MEMORY_REGION_DESCRIPTOR_SETUP_REG_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved
28-16	DESC_SIZE	R/W	0h	This field indicates the size of each descriptor in this memory region. The value programmed is the multipler minus one that needs to be applied to 16 to get the actual descriptor size.So, for 16 and 64 byte descriptors, value programmed will be 0 and 3 respectively.
15-4	RESERVED	R	0h	Reserved
3-0	REG_SIZE	R/W	0h	This field indicates the size of the memory region (in terms of number of descriptors). It is an encoded value that specifies region size as 2^\[5+reg_size\] number of descriptors.

Table 11-2534. Register Call Summary for QM_MEMORY_REGION_DESCRIPTOR_SETUP_REG_0

NAVSS Registers

- QM_MEMORY_REGION_DESCRIPTOR_SETUP_REG_0 to QM_MEMORY_REGION_DESCRIPTOR_SETUP_REG_15 Register (Offset = 8h + [i * 10h], where i = 0 to 15) [reset = 0h]: [0]
- NSS_0_CFG_QMGR0_DESCRIPTOR_REGIONS Registers: [0]

11.13.5.2.4.3 NSS_0_CFG_QMGR0_QUEUE Registers

Table 11-2536 lists the memory-mapped registers for the NSS_0_CFG_QMGR0_QUEUE. All register offset addresses not listed in Table 11-2536 should be considered as reserved locations and the register contents should not be modified.

Table 11-2535. NSS_0_CFG_QMGR0_QUEUE Instances

Instance	Base Address
NSS_0_CFG_QMGR0_QUEUE	040C 0000h

Table 11-2536. NSS_0_CFG_QMGR0_QUEUE Registers

Offset	Acronym	Register Name	NSS_0_CFG_QMGR0_QUEUE Physical Address	Section
0h + (i * 10h)[1]	QM_QUEUE_REG_A_0	Queue i Register A	040C 0000h + (i * 10h)	Section 11.13.5.2.4.3.1
4h + (i * 10h)	QM_QUEUE_REG_B_0	Queue i Register B	040C 0004h + (i * 10h)	Section 11.13.5.2.4.3.2
8h + (i * 10h)	QM_QUEUE_REG_C_0	Queue i Register C	040C 0008h + (i * 10h)	Section 11.13.5.2.4.3.3
Ch + (i * 10h)	QM_QUEUE_REG_D_0	Queue i Register D	040C 000Ch + (i * 10h)	Section 11.13.5.2.4.3.4

(1) i = 0 to 63

11.13.5.2.4.3.1 QM_QUEUE_REG_A_0 to QM_QUEUE_REG_A_63 Register (Offset = 0h + [i * 10h], where i = 0 to 63) [reset = 0h]

QM_QUEUE_REG_A_0 to QM_QUEUE_REG_A_63 is shown in Figure 11-1081 and described in Table 11-2538.

Table 11-2537. QM_QUEUE_REG_A_0 to QM_QUEUE_REG_A_63 Instances

Instance	Physical Address
NSS_0_CFG_QMGR0_QUEUE	040C 0000h + [i * 0h], where i = 0 to 63

Figure 11-1081. QM_QUEUE_REG_A_0 to QM_QUEUE_REG_A_63 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													QUEUE_ENTRY_COUNT																		
R-0h													R-0h																		

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2538. QM_QUEUE_REG_A_0 to QM_QUEUE_REG_A_63 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18-0	QUEUE_ENTRY_COUNT	R	0h	This field indicates how many packets are currently queued on the queue. This count is incremented by 1 whenever a packet is added to the queue. This count is decremented by 1 whenever a packet is popped from the queue.

Table 11-2539. Register Call Summary for QM_QUEUE_REG_A_0

NAVSS Registers
- NSS_0_CFG_QMGR0_QUEUE Registers: [0]
- QM_QUEUE_REG_A_0 to QM_QUEUE_REG_A_63 Register (Offset = 0h + [i * 10h], where i = 0 to 63) [reset = 0h]: [0]

11.13.5.2.4.3.2 QM_QUEUE_REG_B_0 to QM_QUEUE_REG_B_63 Register (Offset = 4h + [i *10h], where i = 0 to 63) [reset = 0h]

QM_QUEUE_REG_B_0 to QM_QUEUE_REG_B_63 is shown in Figure 11-1082 and described in Table 11-2541.

Table 11-2540. QM_QUEUE_REG_B_0 to QM_QUEUE_REG_B_63 Instances

Instance	Physical Address
NSS_0_CFG_QMGR0_QUEUE	040C 0004h + [i * 10h], where i = 0 to 63

Figure 11-1082. QM_QUEUE_REG_B_0 to QM_QUEUE_REG_B_63 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUEUE_BYTE_COUNT																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2541. QM_QUEUE_REG_B_0 to QM_QUEUE_REG_B_63 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QUEUE_BYTE_COUNT	R	0h	This field indicates how many bytes total are contained in all of the packets which are currently queued on this queue.

Table 11-2542. Register Call Summary for QM_QUEUE_REG_B_0

NAVSS Registers
- NSS_0_CFG_QMGR0_QUEUE Registers: [0]
- QM_QUEUE_REG_B_0 to QM_QUEUE_REG_B_63 Register (Offset = 4h + [i *10h], where i = 0 to 63) [reset = 0h]: [0]

11.13.5.2.4.3.3 QM_QUEUE_REG_C_0 to QM_QUEUE_REG_C_63 Register (Offset = 8h + [i * 10h], where i = 0 to 63) [reset = 0h]

QM_QUEUE_REG_C_0 to QM_QUEUE_REG_C_63 is shown in Figure 11-1083 and described in Table 11-2544.

Table 11-2543. QM_QUEUE_REG_C_0 to QM_QUEUE_REG_C_63 Instances

Instance	Physical Address
NSS_0_CFG_QMGR0_QUEUE	040C 0008h + [i * 10h], where i = 0 to 63

Figure 11-1083. QM_QUEUE_REG_C_0 to QM_QUEUE_REG_C_63 Register

31	30	29	28	27	26	25	24
HEAD_TAIL	RESERVED						
W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							PACKET_SIZE
R-0h							R/W-0h
15	14	13	12	11	10	9	8
PACKET_SIZE							
R/W-0h							
7	6	5	4	3	2	1	0
PACKET_SIZE							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; W = Write Only; -*n* = value after reset

Table 11-2544. QM_QUEUE_REG_C_0 to QM_QUEUE_REG_C_63 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	HEAD_TAIL	W	0h	Head/Tail Push Control. Set to zero in order to push packet onto tail of queue and set to one in order to push packet onto head of queue.
30-17	RESERVED	R	0h	Reserved
16-0	PACKET_SIZE	R/W	0h	This field indicates packet size and is assumed to be zero on each packet add unless the value is explicitly overwritten. This field indicates packet size for packet pop operation.

Table 11-2545. Register Call Summary for QM_QUEUE_REG_C_0

Navigator Subsystem (NAVSS)
- Queuing Descriptors: [0][1]
- Packet Queuing: [0][1]

NAVSS Registers
- QM_QUEUE_REG_C_0 to QM_QUEUE_REG_C_63 Register (Offset = 8h + [i * 10h], where i = 0 to 63) [reset = 0h]: [0]
- NSS_0_CFG_QMGR0_QUEUE Registers: [0]

11.13.5.2.4.3.4 QM_QUEUE_REG_D_0 to QM_QUEUE_REG_D_63 Register (Offset = Ch + [i * 10h], where i = 0 to 63) [reset = 0h]

QM_QUEUE_REG_D_0 to QM_QUEUE_REG_D_63 is shown in Figure 11-1084 and described in Table 11-2547.

Table 11-2546. QM_QUEUE_REG_D_0 to QM_QUEUE_REG_D_63 Instances

Instance	Physical Address
NSS_0_CFG_QMGR0_QUEUE	040C 000Ch + [i * 0h], where i = 0 to 63

Figure 11-1084. QM_QUEUE_REG_D_0 to QM_QUEUE_REG_D_63 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DESC_PTR															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DESC_PTR												DESC_INFO			
R/W-0h												R/W-0h			

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2547. QM_QUEUE_REG_D_0 to QM_QUEUE_REG_D_63 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	DESC_PTR	R/W	0h	Descriptor pointer. It will be read as zero if the queue is empty. It will indicate a 32-bit aligned address that points to a descriptor when the queue is not empty.
3-0	DESC_INFO	R/W	0h	A generic value that is preserved during the push/pop cycle through the queue manager. This field will return a 0x0 when an empty queue is read.

Table 11-2548. Register Call Summary for QM_QUEUE_REG_D_0

Navigator Subsystem (NAVSS) • Queuing Descriptors: [0][1] • De-queuing Descriptors: [0][1][2] • Receive Free Descriptor/Buffer Queue Setup (Host Packets): [0]
NAVSS Registers • QM_QUEUE_REG_D_0 to QM_QUEUE_REG_D_63 Register (Offset = Ch + [i * 10h], where i = 0 to 63) [reset = 0h]: [0] • NSS_0_CFG_QMGR0_QUEUE Registers: [0]

11.13.5.2.4.4 NSS_0_CFG_QMGR0_QUEUE_STATUS Registers

Table 11-2550 lists the memory-mapped registers for the NSS_0_CFG_QMGR0_QUEUE_STATUS. All register offset addresses not listed in Table 11-2550 should be considered as reserved locations and the register contents should not be modified.

Table 11-2549. NSS_0_CFG_QMGR0_QUEUE_STATUS Instances

Instance	Base Address
NSS_0_CFG_QMGR0_QUEUE_STATUS	0410 0000h

Table 11-2550. NSS_0_CFG_QMGR0_QUEUE_STATUS Registers

Offset	Acronym	Register Name	NSS_0_CFG_QMGR0_QUEUE_STATUS Physical Address	Section
0h + (i * 10h)[1]	QM_REG_A_0	Queue i Status and Configuration Register A	0410 0000h + (i * 10h)	Section 11.13.5.2.4.4.1
4h + (i * 10h)	QM_REG_B_0	Queue i Status and Configuration Register B	0410 0004h + (i * 10h)	Section 11.13.5.2.4.4.2
8h + (i * 10h)	QM_REG_C_0	Queue i Status and Configuration Register C	0410 0008h + (i * 10h)	Section 11.13.5.2.4.4.3
Ch + (i * 10h)	QM_REG_D_0	Queue i Status and Configuration Register D	0410 000Ch + (i * 10h)	Section 11.13.5.2.4.4.4

(1) i = 0 to 63

11.13.5.2.4.4.1 QM_REG_A_0 to QM_REG_A_63 Register (Offset = 0h + [i * 10h], where i = 0 to 63) [reset = 0h]

QM_REG_A_0 to QM_REG_A_63 is shown in Figure 11-1085 and described in Table 11-2552.

Table 11-2551. QM_REG_A_0 to QM_REG_A_63 Instances

Instance	Physical Address
NSS_0_CFG__qmgr0_queue_status	0410 0000h + [i * 0h], where i = 0 to 63

Figure 11-1085. QM_REG_A_0 to QM_REG_A_63 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													QUEUE_ENTRY_COUNT																		
R-0h													R-0h																		

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2552. QM_REG_A_0 to QM_REG_A_63 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18-0	QUEUE_ENTRY_COUNT	R	0h	This field indicates how many packets are currently queued on the queue.

Table 11-2553. Register Call Summary for QM_REG_A_0

NAVSS Registers

- NSS_0_CFG_QMGR0_QUEUE_STATUS Registers: [0]
- QM_REG_A_0 to QM_REG_A_63 Register (Offset = 0h + [i * 10h], where i = 0 to 63) [reset = 0h]: [0]

11.13.5.2.4.4.2 QM_REG_B_0 to QM_REG_B_63 Register (Offset = 4h + [i * 10h], where i = 0 to 63) [reset = 0h]

QM_REG_B_0 to QM_REG_B_63 is shown in Figure 11-1086 and described in Table 11-2555.

Table 11-2554. QM_REG_B_0 to QM_REG_B_63 Instances

Instance	Physical Address
NSS_0_CFG__qmgr0_queue_status	0410 0004h + [i * 0h], where i = 0 to 63

Figure 11-1086. QM_REG_B_0 to QM_REG_B_63 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUEUE_BYTE_COUNT																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2555. QM_REG_B_0 to QM_REG_B_63 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QUEUE_BYTE_COUNT	R	0h	This field indicates how many bytes total are contained in all of the packets which are currently queued on this queue.

Table 11-2556. Register Call Summary for QM_REG_B_0

NAVSS Registers
- NSS_0_CFG_QMGR0_QUEUE_STATUS Registers: [0]
- QM_REG_B_0 to QM_REG_B_63 Register (Offset = 4h + [i * 10h], where i = 0 to 63) [reset = 0h]: [0]

11.13.5.2.4.4.3 QM_REG_C_0 to QM_REG_C_63 Register (Offset = 8h + [i * 10h], where i = 0 to 63) [reset = 0h]

QM_REG_C_0 to QM_REG_C_63 is shown in Figure 11-1087 and described in Table 11-2558.

Table 11-2557. QM_REG_C_0 to QM_REG_C_63 Instances

Instance	Physical Address
NSS_0_CFG__qmgr0_queue_status	0410 0008h + [i * 10h], where i = 0 to 63

Figure 11-1087. QM_REG_C_0 to QM_REG_C_63 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															PACKET_SIZE																
R-0h															R-0h																

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2558. QM_REG_C_0 to QM_REG_C_63 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16-0	PACKET_SIZE	R	0h	This field indicates the packet size of the head element of a queue.

Table 11-2559. Register Call Summary for QM_REG_C_0

NAVSS Registers
- NSS_0_CFG_QMGR0_QUEUE_STATUS Registers: [0]
- QM_REG_C_0 to QM_REG_C_63 Register (Offset = 8h + [i * 10h], where i = 0 to 63) [reset = 0h]: [0]

11.13.5.2.4.4.4 QM_REG_D_0 to QM_REG_D_63 Register (Offset = Ch + [i * 10h], where i = 0 to 63) [reset = 41h]

QM_REG_D_0 to QM_REG_D_63 is shown in Figure 11-1088 and described in Table 11-2561.

Table 11-2560. QM_REG_D_0 to QM_REG_D_63 Instances

Instance	Physical Address
NSS_0_CFG__qmgr0_queue_status	0410 000Ch + [i * 10h], where i = 0 to 63

Figure 11-1088. QM_REG_D_0 to QM_REG_D_63 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
THRESHOLD_HILO	RESERVED			THRESHOLD			
R/W-1h	R-0h			R/W-1h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2561. QM_REG_D_0 to QM_REG_D_63 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	THRESHOLD_HILO	R/W	1h	{This field indicates whether the number of items in a queue should be greater than equal to or less than the threshold before the queue_ecnt_status\[queue\] bit is asserted.}
6-4	RESERVED	R	0h	Reserved
3-0	THRESHOLD	R/W	1h	{This field indicates the threshold at which the queue threshold pin is asserted.}

Table 11-2562. Register Call Summary for QM_REG_D_0

NAVSS Registers
- NSS_0_CFG_QMGR0_QUEUE_STATUS Registers: [0]
- QM_REG_D_0 to QM_REG_D_63 Register (Offset = Ch + [i * 10h], where i = 0 to 63) [reset = 41h]: [0]

11.14 Peripheral Component Interconnect Express Subsystem (PCIe SS)

This section describes the features and functions of the device Peripheral Component Interconnect Express (PCIe) Controllers which provide a high-speed glueless serial interconnect to peripherals utilizing high bandwidth applications.

NOTE: The supported set of features and peripherals is device part number dependent. For more information, see the device Data Manual.

11.14.1 PCIe SS Overview

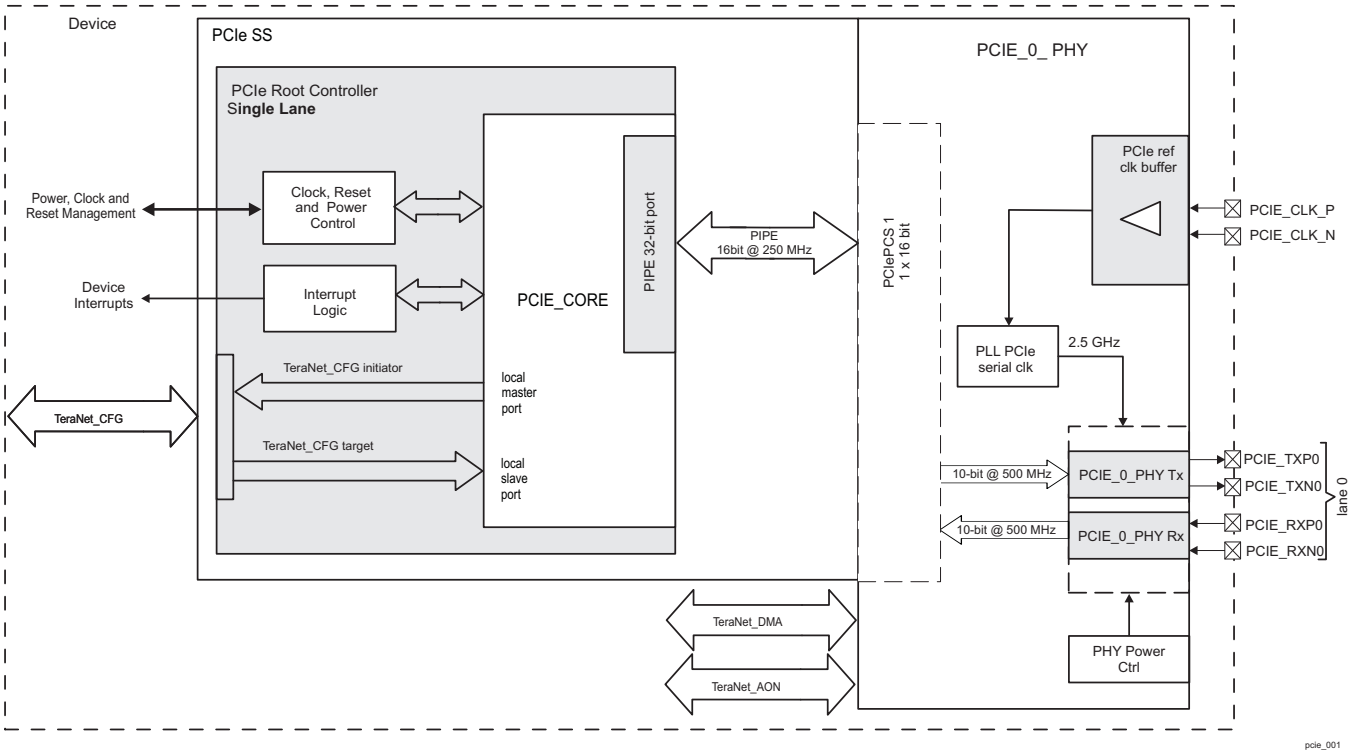
PCIe module is a multi-lane I/O interconnect that provides low pin-count, high reliability, and high-speed data transfer at rates of up to 5.0 Gbps per lane, per direction, for serial links on backplanes and printed curcuit boards. It is a 2nd generation I/O interconnect technology succeeding PCI and ISA bus designed to be used as a general-purpose serial I/O interconnect. It is also used as a bridge to other interconnects such as SATA, USB2/3.0, GbE MAC, and so forth.

The PCI Express standard's predecessor - PCI, is a parallel bus architecture that is increasingly difficult to scale-up in bandwidth, which is usually performed by increasing the number of data signal lines. The PCIe architecture was developed to help minimize I/O bus bottlenecks within systems and to provide the necessary bandwidth for high-speed, chip-to-chip, and board-to-board communications within a system. It is designed to replace the PCI-based shared, parallel bus signaling technology that is approaching its practical performance limits while simplifying the interface design.

NOTE: The device instantiates one PCIe subsystem supporting a single lane.

Figure 11-1089 shows the PCIe module overview.

Figure 11-1089. PCIe Controller Subsystem Overview



11.14.1.1 PCIe SS Features

PCIe module supports the following features:

- Dual operation mode: Root Complex (RC) or End Point (EP), see Section 11.14.4.6.1, *PCI Express Topology*
- Supports a single bidirectional link interface (a single input port and a single output port) with one lane
- Operated at a raw speed of 2.5 Gbps or 5.0 Gbps per lane per direction
- Maximum outbound payload size of 128 bytes
- Maximum inbound payload size of 256 bytes
- Maximum remote read request size of 256 bytes
- Ultra-low transmit and receive latency
- Support for dynamic-width conversion
- Automatic lane reversal
- Polarity inversion on receive
- Single virtual channel (VC)
- Single traffic class (TC)
- Single function in End Point (EP) mode
- Automatic credit management
- ECRC generation and checking
- PCI device power management with the exception of D3cold with Vaux
- PCI Express active state power management (ASPM) state L0s and L1
- PCI Express link power management states, except L2 state
- PCI Express advanced error reporting
- PCI Express messages for both transmit and receive
- Filtering for posted, non-posted, and completion traffic
- Configurable BAR filtering, I/O filtering, configuration filtering, and completion lookup/timeout
- Access to configuration space registers and external application memory-mapped registers through BAR0 and through configuration access
- Legacy interrupts reception (in RC) and generation (in EP)
- MSI generation and reception
- PHY loopback in RC mode

PCIe module does not support the following features:

- No support for multiple lanes
- No support for multiple VCs
- No support for multiple TCs
- No support for function-level reset
- No support for PCI Express beacon for in-band wake
- No built-in hardware support for hot-plug
- No support for vendor messaging
- No support for I/O access in inbound direction in RC or EP mode
- No support for addressing modes other than incremental for burst transactions. Thus, the PCIe addresses cannot be in cacheable memory space
- No auxiliary power to maintain controller context when rezuming from D3cold state
- No support for L2 link state

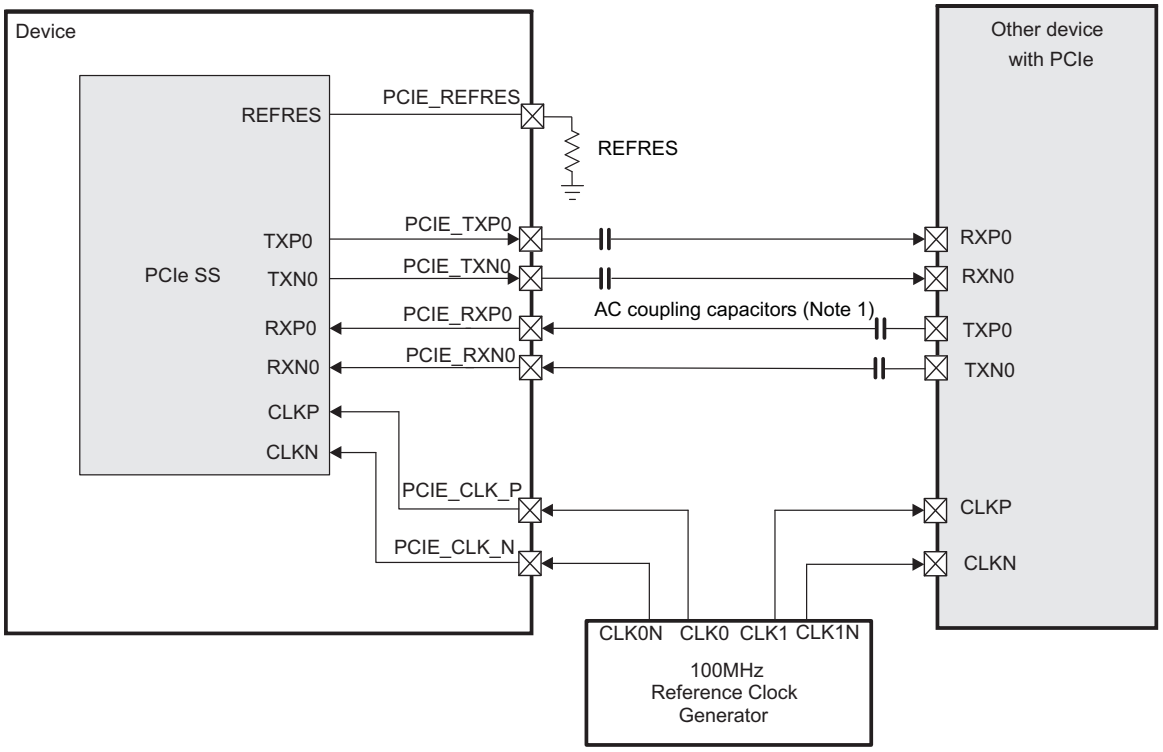
11.14.2 PCIe SS Environment

This section describes PCIe SS connection with an external device.

11.14.2.1 PCIe SS Interface

Figure 11-1090 shows the I/O interface signals of PCIe SS.

Figure 11-1090. PCIe SS Enviroment



Note 1 - For more information about the values of AC Coupling capacitor see PCIe specification

Table 11-2563 lists the PCIe SS interface input/output (I/O) signals.

Table 11-2563. PCIe Interface Signals

Device Signal	Module Signal	I/O[1]	Description
PCIE_RXN0	RXNO	I	RX input of the PCIe port 0 PHY differential transmission line (negative by default)
PCIE_RXP0	RXPO	I	RX input of the PCIe port 0 PHY differential transmission line (positive by default)
PCIE_TXN0	TXNO	O	TX output of the PCIe port 0 PHY differential transmission line (negative by default)
PCIE_TXP0	TXPO	O	TX output of the PCIe port 0 PHY differential transmission line (positive by default)
PCIE_CLK_P	CLKP	I	PCIe Clock Input (positive)
PCIE_CLK_N	CLKN	I	PCIe Clock Input (negative)
PCIE_REFRES	REFRES	A	PCIe SerDes Reference Resistor input (3kΩ ± 1%), see the device Data Manual for more information

(1) I = Input; O = Output; A = Analog

11.14.3 PCIe SS Integration

This section describes module integration in the device, including information about clocks, resets, and hardware requests.

Figure 11-1091 shows the device internal connections with related modules for PCIe functions.

Figure 11-1091. PCIe SS Integration

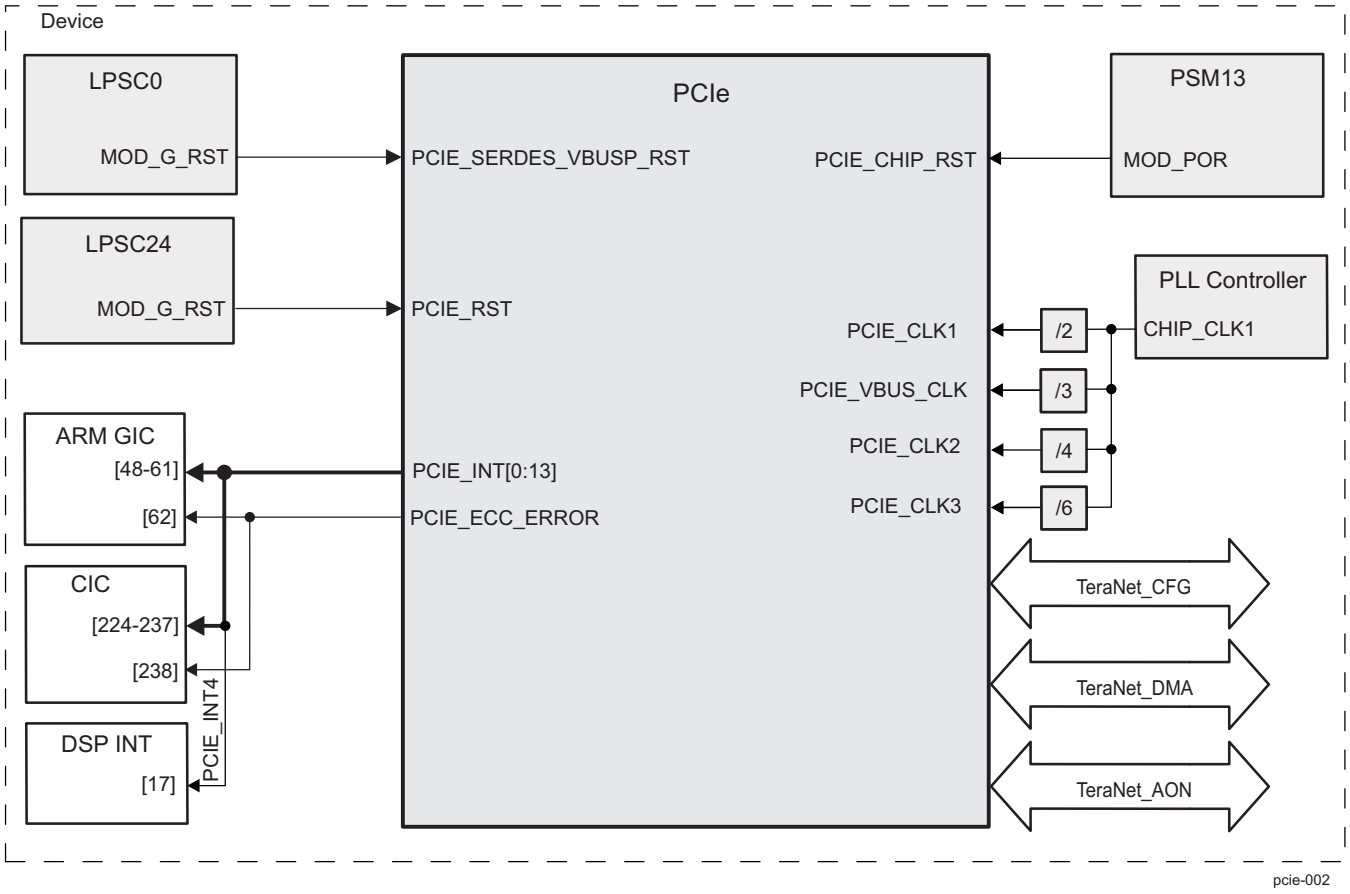


Table 11-2564 through Table 11-2566 summarize the integration of the module in the device.

Table 11-2564. PCIe SS Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
PCIE	PD13	LPSC24	TeraNet_CFG TeraNet_DMA TeraNet_AON
PCIE_0_PHY	PD0	LPSC0	TeraNet_CFG TeraNet_DMA TeraNet_AON

Table 11-2565. PCIe SS Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
PCIE	PCIE_CLK1	CHIP_CKL1/2	PLL Controller	Clock to PCIe module
	PCIE_VBUS_CLK	CHIP_CKL1/3	PLL Controller	Clock to PCIe module
	PCIE_CLK2	CHIP_CKL1/4	PLL Controller	Clock to PCIe module

Table 11-2565. PCIe SS Clocks and Resets (continued)

	PCIE_CLK3	CHIP_CKL1/6	PLL Controller	Clock to PCIe module
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
PCIE	PCIE_RST	MOD_G_RST	LPSC24	PCIe Module Reset signal
	PCIE_CHIP_RST	CHIP_RST	PSM13	Power on reset
PCIE_0_PHY	PCIE_SERDES_VBUSP_RST	MOD_G_RST	LPSC0	SERDES Reset

Table 11-2566. PCIe SS Hardware Requests

Interrupt Requests					
Module Instance	**Event Name**	**Mapped To Input Event [Number]**			**Description**
		ARM GIC	**CIC**	**DSP INTC**	
PCIE	PCIE_INT0	[48]	[224]	-	PCIe SS INT0 Interrupt Request
	PCIE_INT1	[49]	[225]	-	PCIe SS INT1 Interrupt Request
	PCIE_INT2	[50]	[226]	-	PCIe SS INT2 Interrupt Request
	PCIE_INT3	[51]	[227]	-	PCIe SS INT3 Interrupt Request
	PCIE_INT4	[52]	[228]	[17]	PCIe SS INT4 Interrupt Request
	PCIE_INT5	[53]	[229]	-	PCIe SS INT5 Interrupt Request
	PCIE_INT6	[54]	[230]	-	PCIe SS INT6 Interrupt Request
	PCIE_INT7	[55]	[231]	-	PCIe SS INT7 Interrupt Request
	PCIE_INT8	[56]	[232]	-	PCIe SS INT8 Interrupt Request
	PCIE_INT9	[57]	[233]	-	PCIe SS INT9 Interrupt Request
	PCIE_INT10	[58]	[234]	-	PCIe SS INT10 Interrupt Request
	PCIE_INT11	[59]	[235]	-	PCIe SS INT11 Interrupt Request
	PCIE_INT12	[60]	[236]	-	PCIe SS INT12 Interrupt Request
	PCIE_INT13	[61]	[237]	-	PCIe SS INT13 Interrupt Request
	PCIE_ECC_ERROR	[62]	[238]	-	PCIe SS ECC Interrupt Request

NOTE: For more information on device power, reset, and clock management, see Chapter 5, *Device Configuration*.

NOTE: PCIe interrupts are described in Section 11.14.4.4, *Interrupts*.

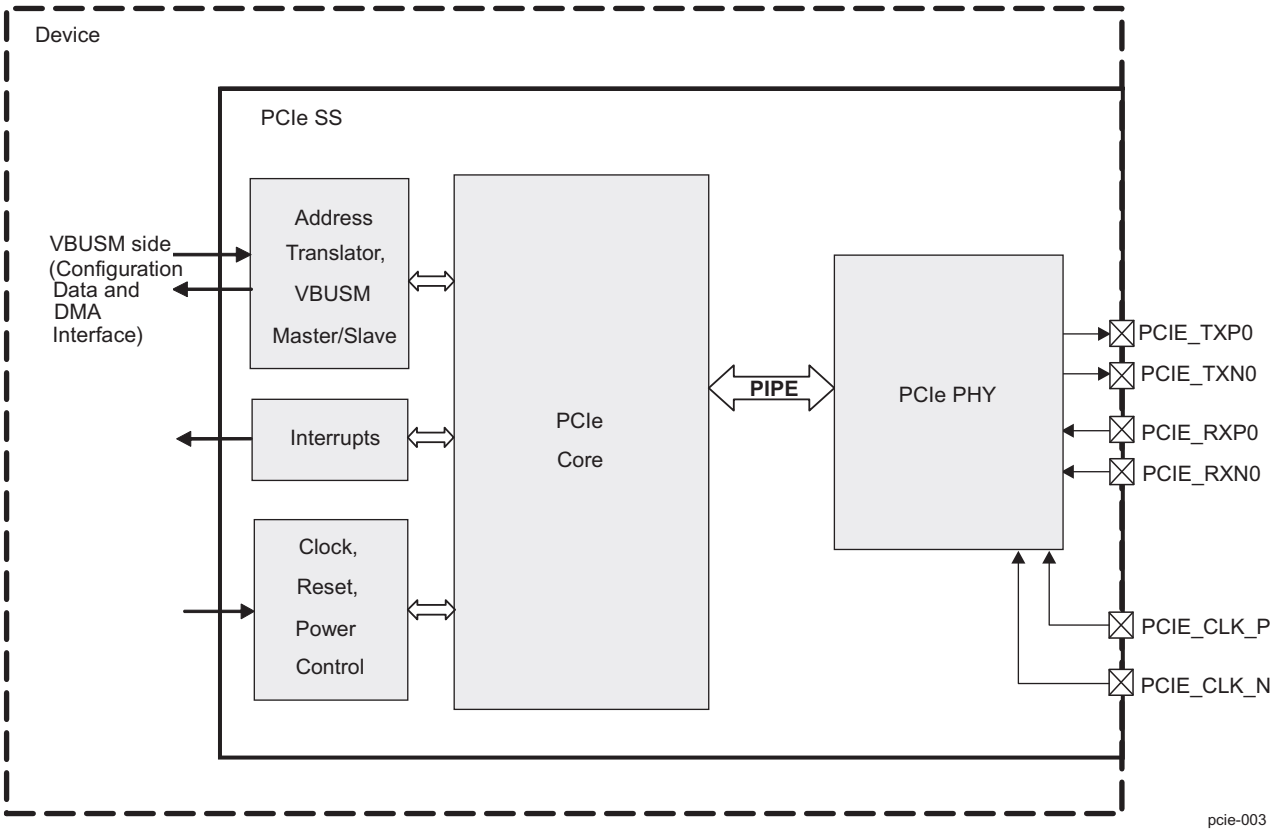
11.14.4 PCIe SS Functional Description

PCIe SS complies with the industry standards listed below:

- PCI Express Base Specification Revision 1.1
- PCI Express Base Specification Revision 2.0

Figure 11-1092 shows the block diagram of the PCIe SS.

Figure 11-1092. PCIe SS Block Diagram



11.14.4.1 PCIe PHY Interface

The PCIE_0_PHY (SERDES PHY) contains the analog portion of the PHY, which is the transmission line channel that is used to transmit and receive data. It contains a phase locked loop, CBA to SB Bridge, Local Registers, CFG/STS Overrides, Clock Gen, Rx Comma Align, and test logic.

11.14.4.2 PCIe Core

11.14.4.2.1 PCIe Core Module

The PCIe core contains the Transaction Layer, Data Link Layer, and MAC part of the PHY. The PCIe core is a Dual Mode core allowing it to operate as RC or EP. As an EP, it can operate as a legacy end point or native PCIe end point. Software can set the PCIe mode (EP, Legacy EP, RC) by writing to the BOOTCFG_DEVCFG[2-1] register, see Section 5.1, *Control Module (BOOT_CFG)* for details. The PCIe Core implements all PCIe specific protocol functionality.

11.14.4.3 Clock, Reset, Power Control Logic

Several clock domains exist within the PCIe SS. These clocks are functional clocks used by the PCIe controller and interface bridges as well as receive and transmit clocks used to clock data in and out. The clocks required for clocking data and PHY functional clocks are generated by the PHY through the supplied input differential clock.

The PCIe SS supports the conventional reset mechanism that is specified within the PCIe Specification. The reset shown in Figure 11-1092 pertains to a hardware reset (cold or warm reset).

In addition to automatic power down mode executed and entered by the hardware (Active State Power Management) when no activity is present, the PCIe SS supports higher level power down mode that is controlled by user software.

11.14.4.4 Interrupts

The PCIe SS is capable of generating up to 14 interrupts (INTA/B/C/D-legacy interrupts combined, MSI, error, power management, and reset) connected to the Interrupt Controller. The user software is required to acknowledge the serviced interrupt by writing to the corresponding vector onto the PCIE_IRQ_EOI register.

11.14.4.5 Differential Data Lines

A pair of differential data lines exists, for both transmit and receive paths lane.

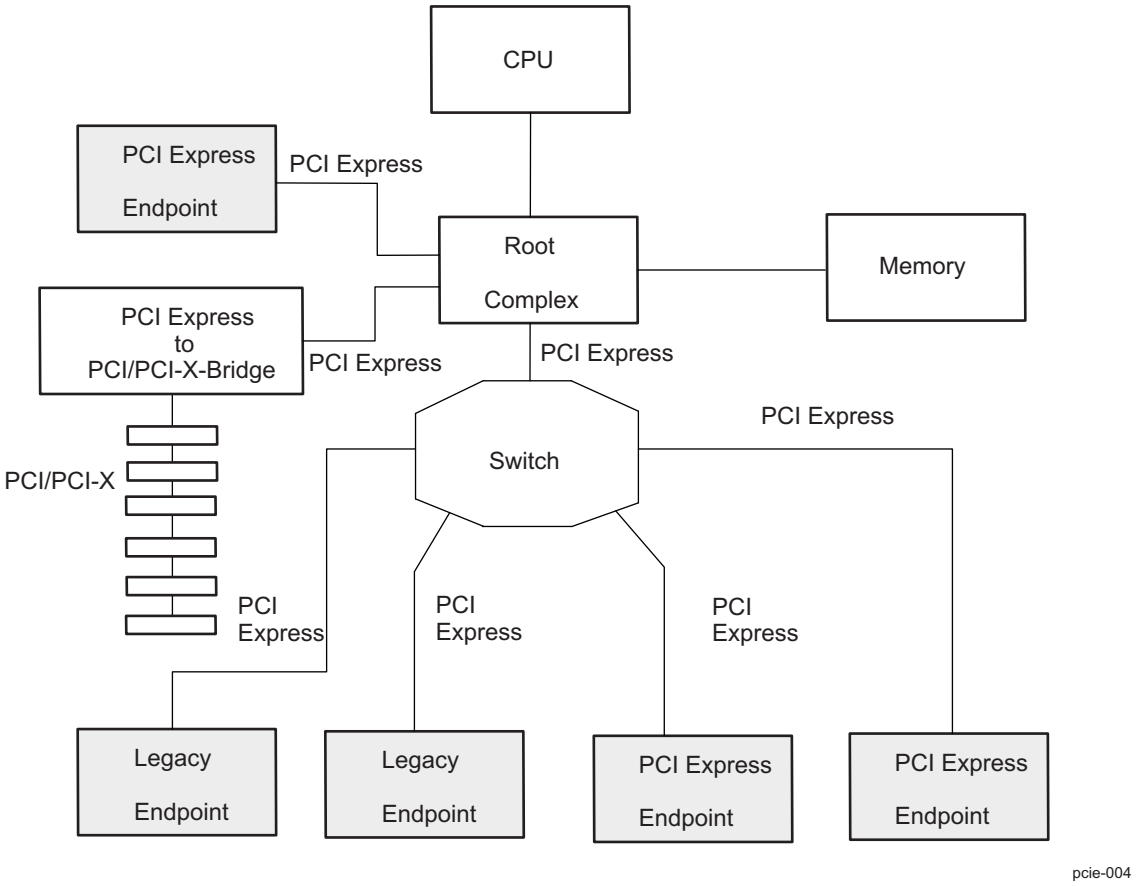
11.14.4.6 Protocol Description

11.14.4.6.1 PCIe Topology

The PCI Express fabric looks like a tree structure with nodes connected to each other via point-to-point links. The root node is called the *root complex* (RC), the leaf nodes are called *end points* (EP) and the nodes that connect multiple devices to each other are called *switches* (SW). The RC can have multiple downstream ports but that still requires multiple instances of the PCIe protocol stack for each RC port.

NOTE: Having one RC port in the PCIe SS does not imply that more than one EP can be connected to the PCIe SS without adding a PCIe switch.

Figure 11-1093. PCIe Example Topology



11.14.4.6.2 Serial Link

PCIe is a point-to-point serial signaling protocol. Each link consists of one TX and one RX differential pair. On each link, depending upon the protocol version, 2.5 Gbps or 5.0 Gbps can be transported in each direction. Accounting for the 8-bit or 10-bit encoding used over the serial link, the data rates translate to 2.0 Gbps and 4.0 Gbps of throughput at higher layers of PCIe protocol. Data is transferred in packets, which include an address and a variable size data payload.

11.14.4.6.3 Supported PCIe Transactions

All of the PCIe transactions defined, Posted and Non-posted, are supported except the Locked Memory Read request transaction and its subsequent completion Locked response transaction. Inbound I/O read and write transactions are also not supported.

Table 11-2567 summarizes the supported PCIe transactions.

Table 11-2567. Supported PCIe Transactions

Transaction Packet Types	Posted/Non-posted
Memory read	Non-posted
Memory write	Posted
I/O read	Non-posted
I/O write	Non-posted
Configuration read (type 0 and type 1)	Non-posted
Configuration write (type 0 and type 1)	Non-posted
Message request without data	Posted
Message request with data	Posted
Completion without data	-
Completion with data	-
Completion without data for locked access	-
Completion with data for locked access	-

The non-posted transactions comprise of a transaction layer packet (TLP) from the requester to completer. The completer, at a later time, sends a completion TLP to the requester. The completion TLP is used to inform the requester that the completer has received the request. In addition, the completion TLP also contains the data if the transaction was a read transaction. Non-posted write transactions contain the data in request TLP.

For posted transactions, the request TLP is sent but there is no response TLP sent from the completer to the requester.

11.14.4.7 Clock Control

The PCIe SS uses multiple clock domains. At the top level, there are only two functional clocks of relevance – a reference clock to the PCIE_0_PHY and a clock for VBUSM interfaces of the subsystem.

The VBUSM interface functional clock is generated from the PLL Controller, see Section 5.4, *Clock Management*, for details.

The other clock input to the PCIe SS is the differential clock. This differential clock is used by the PCIE_0_PHY PLL to generate the necessary functional and bit clock required by the PHY.

To comply with the PCIe Specifications, it is acceptable that a reference clock of 100 MHz that meets the requirements for REFCLK as described in the PCIe Card Electromechanical Specification be driven as a differential signal into PCIe reference clock pads of the device. Greater fluctuation tolerance can be achieved by using one of the higher frequency reference clock values specified in the device Data Manual.

It is recommended that the reference clock be synchronous between the two link partners. To achieve this, a common clock source should be used in the system to provide REFCLK to both ends of each PCIe link.

If common clock architecture is not used, then the software driver must setup appropriate bit (COMMON_CLK_CFG bit in PCIE_LINK_STAT_CTRL register) in the PCIe Configuration registers to indicate so to the system. The number of training sequences is significantly increased if reference clock is not shared between devices on a PCIe link.

The PCIe SS operation is completely dependent upon availability of clock from the PLL that is inside the PHY. All registers in the PCIe SS are located in the clock domain that is dependent upon PLL to be properly configured and in lock. Therefore, for transactions that have been initiated before ensuring that the PLL is locked, the subsystem does not operate. The IP registers are used to enable PLL and verify lock status, see Section 11.14.4.8, *SERDES Configuration*, for more details.

11.14.4.8 SERDES Configuration

The physical layer SERDES has a built-in PLL, which is used for the clock recovery circuitry. The PLL is responsible for clock multiplication of a slow speed reference clock (REFCLKP/N). And the reference clock is the input to PCIe clock pins PCIE_CLK_N and PCIE_CLK_P. This reference clock has no timing relationship to the serial data and is asynchronous to any VBUS clock. The multiplied high-speed clock is routed only within the SERDES block. It is not distributed to the remaining blocks of the peripheral, nor is it a boundary signal to the core of the device. It is extremely important to have a good quality reference clock, and to isolate it and the PLL from all noise sources.

For more information about SERDES Configuration Registers, see Section 11.14.5.1, *SERDES Configuration Registers*.

For more information about implementation instructions for the serializer/deserializer (SerDes)-based interfaces on this device, see SPRUHO3, *KeyStone II Architecture Serializer/Deserializer (SerDes)*.

11.14.4.8.1 Enabling the PLL

Setting of PLL_ENABLE_VAL and PLL_ENABLE_OVL bits of PCIE_PHY_PLL_CTRL enables the internal PLL. For more datails about the stabilization of provided REFCLKP/N, see the device Data Manual.

The PLL_OK bit of PCIE_PHY_PLL_CTRL register will be driven high by the digital lock detector. The software can sample this bit during the PLL initialization to indicate the PLL signal is stable.

11.14.4.8.2 Reference Clock Multiplication

Table 11-2568 summarizes the calculation of the line rate versus PLL output clock frequency.

Table 11-2568. Line Rate versus PLL Output Clock Frequency

Rate	Line Rate	PLL Output Frequency	RATESCALE
Full	x Gbps	0.5x GHz	0.5
Half	x Gbps	1x GHz	1
Quarter	x Gbps	2x GHz	2
REFCLKP/N = (LINERATE*RATESCALE)/MPY			

NOTE: The targeted PLL output frequency (line rate) is 2.5 GHz in both PCIe Gen1 and Gen2 modes. The PCIe PHY performs data bus width conversion from 8-bit to 16-bit when the module switches from Gen1 speed to Gen2 speed. The output frequency of the PLL stays constant irrespective of whether it is operating in Gen1 or Gen2 mode. Set the DIR_SPD bit to 1 in the PCIE_PL_GEN2 register during the initialization to switch the PCIe link speed mode from Gen1 (2.5 Gbps) to Gen2 (5.0 Gbps).

11.14.4.9 Address Translation

PCIe TLP transactions use PCIe addresses. There is a mapping requirement between a PCIe address and a local internal bus address and to accommodate this address mapping, built-in hardware address translation mechanisms exist. That is, if the *Type* field of an outgoing or received TLP indicates the use of address routing, then an outbound or inbound address translation is required and is performed accordingly to map internal bus address to PCIe address or vice-versa using hardware address translators. Address translations for outbound and inbound transactions are discussed below.

PCIe recognizes four address spaces – Memory, I/O, Configuration, and Message. Messages do not consume any Memory or I/O resource. I/O addressing is used by PCIe that supports legacy device operation capabilities; this is not a mandatory feature by PCIe and when supported the I/O spaces is memory mapped to system memory. This implies that only two types of device address spaces, Configuration and Memory spaces, need to map between PCIe address and internal bus address.

The device address space is divided into two spaces (Range 0 and Range 1). Address space (Range 0) that is used for PCIe configuration task is referred to as Configuration space, and a second address space (Range 1) that is used for accessing memory (non-configuration related) is referred to as Memory space.

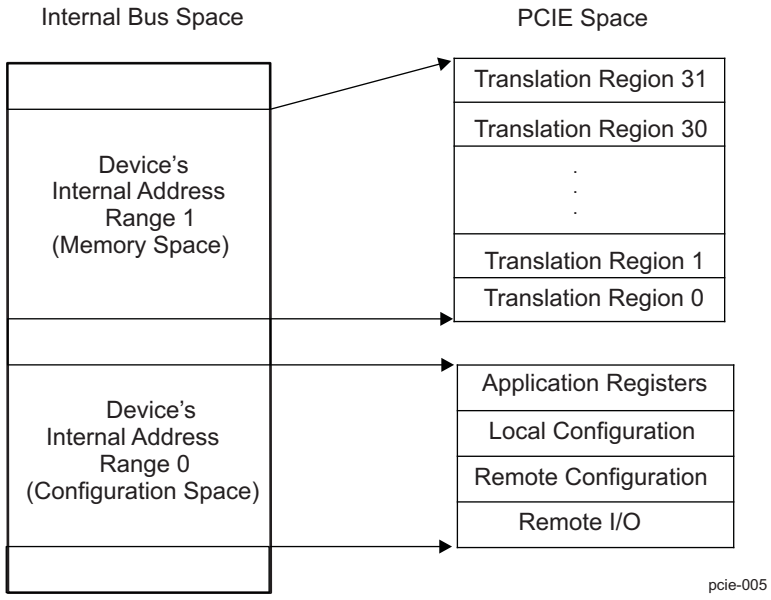
NOTE: All I/O accesses use address routing. There is no support for I/O access in inbound direction. That is, when it comes to a transaction that performs I/O access only TLPs with outbound transactions are supported.

11.14.4.9.1 Outbound Address Translation

The PCIe SS allows mapping of internal bus address to or from PCIe address on outbound TLPs. This is accomplished by using outbound address translation logic. For each outbound read or write request, the address translation module within the PCIe SS can convert an internal bus address to a PCIe address of memory read or write type.

The address translation logic uses information in address translation registers to perform the mapping. The registers, PCIE_OB_SIZE, PCIE_OB_OFFSET_INDEXn and PCIE_OB_OFFSETn_HI are used in conjunction with outbound address translator.

Figure 11-1094. Outbound Address Translation



The memory range that the PCIe SS occupies in the devices internal address range is divided into 32 equally-sized translation regions (Regions 0 to 31). These regions can be programmed to be of 1, 2, 4, or 8MB in size via the PCIE_OB_SIZE register. Each such region can be remapped to a PCIe address range of same size as the size of translation region itself. The address translation logic identifies and extracts the 5 bits (32 regions) of the device's internal address and determines which of the 32 regions it belongs to. The bit address positions of these 5 bits depend on the range size. Once the region is identified, the address translation logic then generates PCIe base address, from the values provided within the corresponding configuration registers for that region, which are the registers PCIE_OB_OFFSET_INDEXn (n = [1,31]) and PCIE_OB_OFFSETn_HI (n = [1,31]). If 32-bit addressing is used, PCIE_OB_OFFSETn_HI will always be programmed with zero.

Once the PCIe base address has been identified, the offset that is to be added to this base address is derived from the lower bit fields of the internal bus address and the bit fields that make up this offset correspond to the size of the regions.

Application software is required to identify the desired internal bus memory that is to be accessible by PCIe module and initialize the corresponding registers prior to enabling PCIe transactions.

To accomplish address translation, each region uses three registers content values:

1. **Outbound Size Register (PCIE_OB_SIZE)** – Application software initializes this register with the size value. This field is used to identify the size of all equally spaced regions and identify one of the 32 regions based on the size value. OB_SIZE field = 0, 1, 2 and 3 corresponds to region sizes of 1MB, 2MB, 4MB and 8MB and the corresponding indexed regions for these sizes are bits[24-20], bits [25-21], bits[26-22] and bits[27-23]. So, index identification is done based on the size of regions and the values of these 5-bit fields identify one of the 32 Regions that are used for mapping. This affects the meaningful bit fields used within PCIE_OB_OFFSET_INDEXn register. The index value will be extracted from the device internal bus address and will be matched to one of the 32 regions. The lower bit fields corresponding to the size of the region are masked and not used in the mapping.
2. **Outbound Translation Region n Offset Low and Index Register (PCIE_OB_OFFSET_INDEXn)** – Application software initializes this register with the 32-bit PCIe address. Not all bits in this field are used in all cases. The number of bits used changes depending upon the size of each address translation region. If each region is 1MB, 2MB, 4MB or 8MB, then this field will be used to create bits[31-20], bits[31-21], bits[31-22] or bits[31-23] of the translated address.
3. **Outbound Address Offset High Register (PCIE_OB_OFFSETn_HI)** – Application software initializes this register and this is a 32-bit field that represents bits [63-32] of translated PCIe space address if using 64-bit addressing. This register is required to be programmed with a Zero value if using 32-bit addressing.

Example 11-5 demonstrates the mapping of a given internal bus address to a PCIe address.

Example 11-5. Outbound Address Translation

For a given device internal bus address of 0x9D3A1234, what would be the corresponding PCIe address that would be used on an outgoing TLP header (assume a 2MB region partition)?

For this example, further assume the following:

- 64-bit addressing is to be used
- Translation Region 9 high offset PCIE_OB_OFFSET9_HI = 0x33445566
- Translation Region 9 low offset PCIE_OB_OFFSET9_INDEX = 0x56E00001 (Region 9 is enabled)

For an internal address 0x9D3A1234, the translated address is calculated based as outlined below:

1. Extract the 5-bit index and offset from given internal bus address of 0x9D3A1234. Because the region partition is 2MB (PCIE_OB_SIZE = 1), bits [25-21] of address 0x9D3A1234 is extracted as 01001b, which is 9.
2. Match it to one of the region index fields. In this example, it matches translation region 9 as internal address bits [25-21] are equal to value of index 9. So the translation registers contents for region 9 will be used.
3. Use programmed offsets 0x33445566 (from PCIE_OB_OFFSET9_HI) for 64-bit addressing bits[63-32] and 0x56E (from PCIE_OB_OFFSET9_INDEX) for bits[31-21]. The translated PCIe base address =

Example 11-5. Outbound Address Translation (continued)

0x33445566 56E00000.

4. Map the offset (bits[20-0]) of the internal bus address directly to PCIe address. Bits[20-0] of 0x9D3A1234 is 0x1A1234.
5. Compute translated address = translated base address + offset = 0x33445566 56E00000+0x1A1234 = 0x33445566 56FA1234.

In Example 11-5, 2MB size region will allow a total of 2MB*32 = 64MB unique internal bus address location to be mapped onto a corresponding PCIe address, values 0x0000 0000 to 0x03FF FFFF. Any internal bus address outside this 64MB space will not be uniquely mapped and the internal bus address will be truncated to fall in within the 64MB region, which happens to be for this example. In other words, bits [31-26] of the given example internal bus address is masked and does not contribute to the PCIe translated address generation (for example, internal bus address 0x9D3A 1234 will have the same PCIe translated address as 0x913A 1234 or 0x953A 1234, and so forth).

NOTE: The outbound translation is used only for memory read and write transactions and not for PCIe configuration or I/O read and write transactions. Outbound translation may not be necessary in all applications.

11.14.4.9.1.1 Transactions Violating Address Translation Boundaries

If a transaction is large enough that it goes past the address translation region, unspecified behavior may occur. The address translation works only at the time a command is issued. So, a memory write for example, does not automatically go to the next translation region if the write has started in the previous one and is larger than the remaining size in the starting translation region.

11.14.4.9.2 Inbound Address Translation

Inbound address translation is used to remap accepted incoming accesses from other PCIe devices to locations within the device's memory map.

The PCIe SS is aware of two internal bus address spaces. The first address space (Address Space 0) is dedicated for local application registers, local configuration accesses, remote configuration accesses and remote I/O accesses (RC only). The second one (Address Space 1) is dedicated for data transfer. Details on this are covered in Section 11.14.4.10, *PCIe Address Spaces*.

Address Space 0 occupies a contiguous 16KB of location, where 4KB of 16KB is the configuration space. Address Space 1 is used for data transfer and is large in size and not necessarily contiguous. To perform a mapping of a PCIe addresses that would land within Address Space 1, 4 dedicated regions (Region[0-3] are to be used by the inbound address translator to achieve the desired address mapping. Otbound translation has 32 regions while inbound translation has 4 regions.

Any accepted incoming address should match one of the BARs (Base Address Registers). If the PCIe SS is configured as EP, PCIE_BAR0÷PCIE_BAR5 will be mapped to one of the two address spaces (Address Space 0/1). If the PCIe SS is configured as RC, only PCIE_BAR0 and PCIE_BAR1 will be used to be mapped to the address spaces. See Section 11.14.4.9.3, *Use of Base Address Registers (BARs)*, for details of BAR usage.

In 32-bit addressing, PCIE_BAR0 is dedicated to Address Space 0 and PCIE_BAR1÷PCIE_BAR5 are dedicated to Address Space 1. Region0, Region1, Region2, and Region3 should be associated only with PCIE_BAR1÷PCIE_BAR5.

In 64-bit addressing, a pair of adjacent BARs concatenated is required to hold the 64-bit address. This means PCIE_BAR0 and PCIE_BAR1 hold 64-bit address with PCIE_BAR0 holding the lower 32-bit address to match while PCIE_BAR1 holding the higher 32-bit address to match. And PCIE_BAR0 and PCIE_BAR1 are dedicated to Address Space 0. The same holds for Address Space 1 association. The two pairs of registers - PCIE_BAR2 and PCIE_BAR3, and PCIE_BAR4 and PCIE_BAR5, hold the accepted 64-bit address that is associated to Address Space 1 where mapping takes place via the use of Region0, Region1, Region2, and Region3.

Address translation for Address Space 0 does not exist because the internal location is unique and is contiguous. The PCIe address should be within the received TLP address matches PCIE_BAR0 for 32-bit addressing, and PCIE_BAR0, PCIE_BAR1 for 64-bit addressing.

In 64-bit addressing, PCIE_BAR0 and PCIE_BAR1 are concatenated and dedicated to Address Space 0. There is no other BAR available in RC mode to map packets with 64-bit addressing to any other internal memory regions with inbound address translation.

However, address translation for Address Space 1 requires the use of one of the four regions (Region0, Region1, Region2, and Region3 to map accepted TLPs to internal memory. Four memory mapped registers that are region specific are used by the inbound address translator.

1. **Inbound Translation Bar Match Register (PCIE_IB_BARn (n = [1,3]))** – This field indicates which BAR the inbound transaction must be targeted to for the translation rule specified to activate.
2. **Inbound Translation Start Address High Register (PCIE_IB_START_HIn (n = [1,3]))** – This field indicates the starting address bits [63-32] as seen in PCIe address. Typically, this field will match the BAR value and is used as the reference for this address translation region in 64-bit addressing. This register is required to be programmed with a Zero value if using 32-bit addressing.
3. **Inbound Translation Start Address Low Register (PCIE_IB_START_LOn (n = [1,3]))** – This field indicates the starting address bits [31-8] as seen in PCIe address. Typically, this field will match the BAR value and is used as the reference for this address translation region in both of 32-bit and 64-bit addressing.
4. **Inbound Translation Address Offset Register (PCIE_IB_OFFSETn (n = [1,3]))** – This is the VBUSM address that will be the starting point of the mapped or translated PCIe address region.

The procedure used by the inbound address translator to perform the mapping between the PCIe address and internal bus address as follows:

1. Extract the offset: PCIe address – (PCIE_IB_STARTn_HI : PCIE_IB_STARTn_LO).
2. Compute internal address: add PCIE_IB_OFFSETn to the offset extracted on step 1.

Example 11-6 demonstrates the mapping of a given PCIe address to an internal bus address.

Example 11-6. Inbound Address Translation

For a given 64-bit PCIe address of 0x12345678 ABC50000, which qualifies for acceptance, what would be the corresponding internal bus address (assume that Region1 registers are programmed to match PCIE_BAR2, PCIE_BAR3 programmed addresses)?

For this example, further assume that application software has programmed the set of registers corresponding to Region1 as follows:

- PCIE_IB_BAR1 = 2. This assignment associates Region1 with PCIE_BAR2, PCIE_BAR3 for 64-bit addressing. The programmed value 2 here associate the match between Region1 (PCIE_IB_BAR1) and configuration register PCIE_BAR2.
- PCIE_IB_START1_HI = 0x12345678
- PCIE_IB_START1_LO = 0xABC00000
- PCIE_IB_OFFSET1 = 0x33400000

The internal bus address is computed by extracting the offset from the PCIe address (address within the TLP header) and add the resultant to the start address of the internal address.

- Extract the offset from the PCIe address:
  - Extracted offset = PCIe address – (PCIE_IB_STARTn_HI : PCIE_IB_STARTn_LO) =
  - = 0x12345678 ABC50000 - 0x12345678 ABC00000 =
  - = 0x00050000
- Compute internal device address:
  - Internal address = PCIE_IB_OFFSET + extracted offset =
  - = 0x33400000 + 0x00050000 =
  - = 0x33450000

In this example, PCIe address 0x12345678 ABC50000 is translated or mapped to internal bus address 0x33450000.

11.14.4.9.2.1 Mapping Multiple Non-Contiguous Memory Ranges To One Region

A single inbound address translation region can be used to map accesses to multiple non-contiguous internal address ranges, within Address Space 1, by ensuring that the start addresses of these regions are programmed in ascending order in the inbound start address registers.

Example 11-7. Mapping Multiple Non-Contiguous Memory Ranges to One Region

In this example, two BARs (PCIE_BAR1 and PCIE_BAR2) are remapped to four separate locations that are non-contiguous using 32-bit addressing. The first two regions (Region0 and Region1) remap accesses landing in PCIE_BAR1 space and the following two regions (Region2 and Region3) remap accesses landing in PCIE_BAR2 space. Each BAR is treated as a 32-bit BAR and therefore, the higher 32-bits of start address are zero.

Programmed value of Base Address Registers in configuration space:

- PCIE_IB_BAR1: 0x11110000
- PCIE_IB_BAR2: 0x22220000

Table 11-2569. Mapping Multiple Non-Contiguous Memory Ranges to One Region

IB Region	PCIE_IB_BAR	PCIE_IB_START_ADDR	PCIE_IB_OFFSET	TLP Address	Translated Address
0	1	0x11110000	0x33330000	0x11110080	[0x11110080 - 0x11110000] + 0x33330000 = 0x33330080
1	1	0x11118000	0x44440000	0x11119000	[0x11119000 - 0x11118000] + 0x44440000 = 0x44441000
2	2	0x22220000	0x55550000	0x22220400	[0x22220400 - 0x22220000] + 0x55550000 = 0x55550400
3	2	0x22220800	0x66660000	0x22231000	[0x22231000 - 0x22220800] + 0x66660000 = 0x66670800

In case there are more than one matching BAR, then the highest start address that is less than the PCIe TLP address is the one that is considered for translation.

11.14.4.9.2.2 PCIE_IB_BAR0 Exception for In-Bound Address Translation

The memory space covered by PCIE_BAR0 in inbound direction is completely dedicated to accessing the application registers, Address Space 0, in both RC and EP modes. It implies that the PCIE_BAR0 cannot be remapped to any other location but to application registers. Any remote inbound access matching the PCIE_BAR0 region will automatically be routed to these registers. It allows RC devices to control EP devices in absence of dedicated software running on EP. For RC and EP, this mapping of PCIE_BAR0 to registers allows the message signaled interrupts to work. There is no support to disable PCIE_BAR0 accesses from reaching the application registers.

11.14.4.9.2.3 Using PCIE_IB_BAR1 Value As Start Address

The inbound address translation for PCIE_BAR1 has additional capability of using the value of PCIE_BAR1 register (from PCIe configuration space) as the start address for inbound address translation. This feature can be activated by leaving the start address of the corresponding inbound translation region (one of the four regions associated to PCIE_BAR1) programmed with zero. When an incoming read/write access matches PCIE_BAR1 and the inbound region's BAR match (PCIE_IB_BARn) is set to 1 with start address programmed to zero, the PCIE_BAR1 value is used to compute the translated address.

NOTE: If this feature is used, only one inbound translation window is available and it is relative to PCIE_BAR1 programmed value. No other BARs or inbound translation windows can be used if PCIE_BAR1 is designated as the reference for inbound address translation.

11.14.4.9.3 Use of Base Address Registers (BARs)

Base address registers (BARs) are treated differently depending upon the type, Type 0 or Type 1, of the configuration space supported by the device. Typically, a RC module uses Type 1 and EP configuration uses Type 0 configuration space.

The main difference between Type 0 and Type 1 configuration space is in regarding TLP filtering. When a TLP is to be routed by its address, the address range in the BAR will decide whether the TLP is rejected or accepted. In the case of an EP, if the address is in the range configured in the BAR, the EP will accept the TLP and pass it to the internal bus side. In the case of RC, if the address is in the range configured in the BAR or is outside of the range defined by the three Base/Limit register sets (non-prefetchable memory, prefetchable memory, and I/O), and then that TLP is accepted.

These three Base/Limit registers contain the address ranges setup for all of the downstream devices:

- Memory Limit and Base Register (PCIE_MEMSPACE)
- Prefetchable Memory Limit and Base Register (PCIE_PREFETCH_MEM). Prefetchable Memory Base Upper 32 bits Register (PCIE_PREFETCH_BASE) and Prefetchable Limit Upper 32 bits Register (PCIE_PREFETCH_LIMIT) are used as well for 64-bit addressing.
- I/O Base and Limit Upper 16 bits Register (PCIE_IOSPACE)

Therefore, any TLP coming into the RC with the downstream address within the range means that it is intended for a device somewhere downstream of RC, and therefore it has been misrouted and will be rejected by the RC filter. TLPs with an address outside the range of the Base/Limit registers are given to the RC application.

For an RC port, the use of BAR register is applicable only when the RC port itself has a memory that can be targeted from the PCIe link. The BAR range in the RC is normally must be outside of the three Base/Limit regions. In a normal system setup, the system memory would be connected on the host/CPU bus. This host/CPU is the initiator/completer for TLPs. To access a downstream device, the host/CPU would initiate a request that is forwarded by the RC port to either a switch or an EP device. Completions (if any), for this request are given to the host/CPU application. If a downstream EP wants to access the system memory, it would initiate a request with an address range outside of the RC port Base/Limit registers. That TLP will be forwarded to the host/CPU interface. If the BAR range in RC is inside of the three Base/Limit regions, the TLP with the address targeted to the RC BAR range will still be accepted by RC.

11.14.4.9.3.1 BAR Mask Registers

The BARs are accessible in both EP and RC modes by the device itself via internal bus. The BARs in EP devices are usually programmed by RC during the PCIe configuration process. Because the PCIe SS is a Dual Mode (DM) core, prior to the time when PCIe configuration starts, the PCIe SS EP device has the opportunity to modify the behavior of BAR registers prior to RC starts the enumeration process and assignment of PCIe Base Address. Using the BAR Mask registers, which are overlaid on the BAR registers, the software on the EP device can configure the amount of address space that the EP will request from the RC during configuration for each of its BARs. The software can also modify the BAR types and can enable/disable the BARs. That is, the read-only bit fields of BARs are user-programmable and is required to be programmed prior to the enumeration process begins.

NOTE: The BAR Mask registers are accessible, for configuration purposes of the BARs, in both EP and RC mode and only when DBI_CS2 bit in PCIE_CMD_STATUS is enabled.

The software needs to clear PCIE_CMD_STATUS [DBI_CS2] after initial configuration prior to RC starts enumeration. The software needs to always read back the written value to DBI_CS2 after modification to ensure that the write has completed because the read will not happen until the write completion. And the read back should be done in both cases of enabling and disabling the DBI_CS2 bit.

The BAR Mask registers are writable but not readable. The read back value will be the BAR registers values instead of BAR Mask registers.

In order to verify if the BAR Mask registers have been set correctly, software writes test pattern into BAR registers and then sees if the bits have been masked correctly.

For example, if PCIE_BAR0 is 32-bit BAR, enable DBI_CS2 bit, write PCIE_BAR0_MASK = 0x00003FFF, disable DBI_CS2 bit, write PCIE_BAR0 = 0xFFFFFFF0, then PCIE_BAR0 should be read as 0xFFFFC000, as the lower bits of PCIE_BAR0 have been masked.

It is important to not attempt modification of BAR Mask registers from serial link side as unpredictable behavior may occur and the system would become unusable.

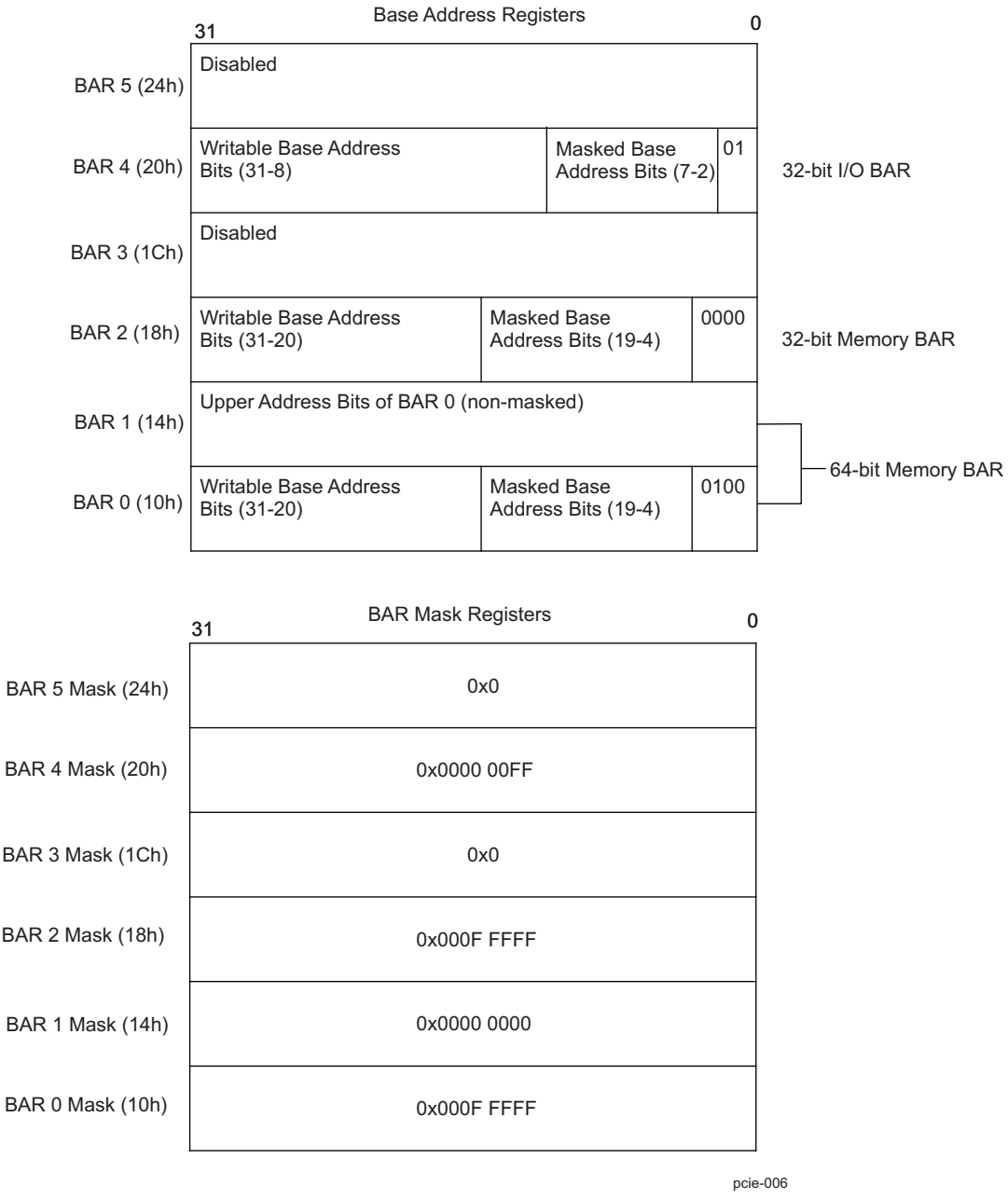
The PCIE_BAR0 Mask register in both RC and EP modes is fixed because PCIE_BAR0 is completely dedicated to accessing the application registers. Only bits [31-20] in PCIE_BAR0 are configurable. Refer to section Section 11.14.4.9.2.2 for more details.

11.14.4.9.3.2 Example BAR Setup

Figure 11-1095 shows an example configuration of the six BARs and their corresponding BAR Mask Registers (EP mode). The example configuration includes:

- One 64-bit memory BAR (non-prefetchable)
- One 32-bit memory BAR (non-prefetchable)
- One 32-bit I/O BAR

Figure 11-1095. Example Base Address Register Configuration



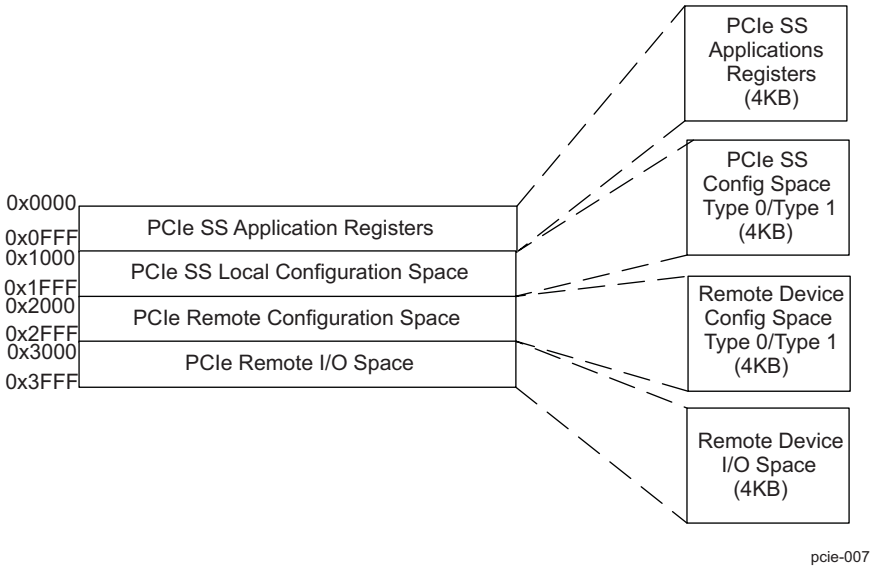
11.14.4.10 PCIe Address Spaces

PCIe SS has two VBUSM regions.The first (Address Space 0) is dedicated for local application registers, local configuration accesses, remote configuration accesses and remote I/O accesses (RC only). The second (Address Space 1) is dedicated for data transfer.

11.14.4.10.1 Address Space 0

Address Space 0 occupies a contiguous 16KB of memory partitioned into 4 regions with equal sizes (each 4KB long). Figure 11-1096 illustrates the relationship of the various address regions within the Address Space 0.

Figure 11-1096. Mapping of the PCIe SS Address Space 0



Each of the four regions is meant for accessing a set of registers.

1. **PCIe SS Application Registers** – These registers are used to configure and monitor various settings within the PCIe SS. These registers are specific to the application needs and are not related to the PCIe configuration registers. All PCIe SS application registers should be accessed in 32-bit mode even if the PCIe SS data bus is wider than 32 bits.

   The application registers are accessible in multiple ways depending on the point of origin of such accesses. When accessed from the device internal bus slave, these registers are accessed via the 4KB space in Address Space 0. When accessed from the PCIe serial link side, the application registers are mapped to PCIE_BAR0 of an EP as well as RC. In addition, a RC can access these registers in a PCIe EP via the upper 1KB space of the PCIe configuration space. The offsets of the registers do not change irrespective of what the mode of accessing these registers is.
2. **PCIe Local Configuration Registers** – These registers are used to read the settings of configuration registers of the local PCIe device. Prior to PCIe configuration is complete, these registers can also be written to. Depending upon whether the PCIe SS is configured as RC or EP, the layout of these registers is either Config Space Type 0 or Type 1. All accesses on PCIe local configuration registers must be made in 32-bit mode even if the PCIe SS data bus is wider than 32 bits.
3. **PCIe Remote Configuration Registers** – PCIe remote configuration registers are accessed by first setting up the bus number, device number and function number of the remote PCIe device in one of the application register (PCIE_CFG_SETUP) and then accessing the PCI remote configuration registers as if it were the PCIe configuration space of a single PCIe function. The layout of the remote configuration registers varies based upon whether the device is an end point or a PCIe switch.
4. **PCIe I/O Access Window** – A 4KB region is dedicated for remote I/O accesses when the PCIe SS is in RC mode. Any access made on this space becomes an I/O access. The actual address in the TLP gets its base address from the PCIE_IOBASE Register value and an offset that is directly derived from the address of the internal bus access in this 4KB space.

NOTE: None of the four address ranges described support burst transactions. Only single 32-bit transactions should be issued to these addresses. These addresses should also be configured as non-cacheable address space.

CAUTION

The remote configuration and I/O transaction windows are directly mapped to internal bus space, which is why software must not access these spaces when there is no operational PCIe link. No response may be generated for such transactions. It is recommended that checks be built into software to avoid remote accesses in the absence of an operational link.

11.14.4.10.1.1 Organization of Configuration Registers

As showed in Table 11-2570, the registers for various PCIe capabilities are linked to each other via address offsets specified in the registers themselves.

Table 11-2570. Layout of PCIe Configuration Registers

Offset From Start of Configuration Space	Register Block
0x000	PCI-Compatible Header (Type 0/1)
0x040	Power Management Capabilities Registers
0x050	MSI Capabilities Registers
0x070	PCIe Capabilities Registers
0x100	PCIe Extended Capabilities Registers
0x700	Port Logic Registers

11.14.4.10.2 Address Space 1

The second address space is used for data transfer. The BAR values setup in PCIe local configuration registers define where within the memory map of the CPU on root complex side the EPs are located. All locations other than what is setup in BAR registers of the EPs are on the RC side.

The Address Space 1 is mapped to multiple devices in RC mode. Each remote device could be allocated a portion of this memory space and any transaction that is targeted to this address space gets converted to a PCIe transaction targeted to the appropriate remote PCIe device.

11.14.4.11 DMA Support

The PCIe SS does not have DMA capabilities built into it. It has internal bus interface slave and master ports connected to the chip-level interconnect. If the PCIe SS is accessing the external PCIe device, a DMA engine can make burst-data reads and writes on the slave port of the PCIe SS. And the master port on the PCIe SS can initiate reads and writes to memory on behalf of a remote PCIe device without need of DMA. Both of the master and slave ports can be used, no matter the PCIe SS is in RC or EP mode.

11.14.4.11.1 DMA Support in RC Mode

When operating in root complex mode, a DMA controller internal to the device (outside of the PCIe SS) can perform DMA to and from any remote device located on the PCIe fabric. The memory address of such devices is available to the software via the PCIe bus enumeration procedure. In addition, PCIe SS has a provision to perform memory address translation on outbound requests. Thus, the software is able to map different memory regions in its memory map to correspond to different addresses (and different access types) on the PCIe side.

There are bandwidth implications of using an external DMA. If the PCIe core has been programmed to establish a link in the PCIe 2.5 Gbps rate, then the DMA controller that drives the PCIe SS slave port must be able to write/read data at about 85% of 2 Gbps bandwidth per PCIe link. For a PCIe link speed of 5.0 Gbps, the DMA controller must be able to provide bandwidth of about 85% of 4 Gbps per PCIe link.

In addition, the master port on PCIe port can issue read/write accesses that have been initiated by a remote PCIe device. The interconnect fabric should provide sufficient capacity to serve 85% of 2 Gbps (4 Gbps in Gen2 mode) per PCIe link in each direction.

11.14.4.11.2 DMA Support in EP Mode

When operating as a PCIe EP, the device will be located in PCIe memory map at location programmed in the Base Address Registers by the PCIe RC device. Any PCIe transactions destined for the device from the upstream ports get transferred to the master port on PCIe SS. Similarly, any transactions originating from the software is sent over to PCIe link.

In end point mode, the PCIe SS provides address translation functionality. It is possible to map I/O, config and memory accesses originating on PCIe side to memory accesses with different address on the internal bus side. These address ranges are configurable through application registers.

The approximate data rate for each of these transactions will be about 85% of 2 Gbps (4 Gbps in Gen2 mode) in each direction on each PCIe lane.

11.14.4.11.3 EDMA Transfer Examples

11.14.4.11.3.1 Memory Write Transfer

Example 11-8 details how to perform an EDMA transfer from the device source (for example, L2 or EMIF) to PCIe memory. For the EDMA configuration, see Chapter 10, *Enhanced Direct Memory Access (EDMA) Controller*.

Example 11-8. Memory Write

```
/* Configure PCIe SS module and prepare transfer parameters*/
/* Initialize the PCIe SS module */
/* Define payload size and buffer size */
PCIE_max_payload = 128; /* Assume PCIe maximum payload size is 128 bytes */
buff_size = 2048; /* Assume the transfer buffer size 2048 bytes */
/* Define PCIe data starting address for EDMA transfer */
PCIEAddr = <PCIe data address defined in the device Data Manual>
...
```

Example 11-8. Memory Write (continued)

```
/* Setup EDMA module and enable the DMA region */
/* setup EDMA PARAM */
/* The OPT register contains following bit fields:
ITCCHEN = 0x0; TCCHEN = 0x0; ITCINTEN = 0x0; TCINTEN = 0x1;
TCC = 0x0; TCCMODE = 0x0; FWID = 0x0; STATIC = 0x0; SYNCDIM = 0x1; DAM = 0x0; SAM = 0x0 */
paramSetup_PCIE.option = CSL_EDMA3_OPT_MAKE(0,0,0,1,0,0,0,0,1,0,0);
paramSetup_PCIE.aCntbCnt = CSL_EDMA3_CNT_MAKE(PCIE_max_payload,
buff_size/PCIe_max_payload);
paramSetup_PCIE.srcDstBidx = CSL_EDMA3_BIDX_MAKE(PCIE_max_payload,PCIe_max_payload);
paramSetup_PCIE.srcDstCidx = CSL_EDMA3_CIDX_MAKE(0,0); paramSetup_PCIE.cCnt = 1;
paramSetup_PCIE.linkBcntrld = CSL_EDMA3_LINKBCNTRLD_MAKE(CSL_EDMA3_LINK_NULL,0);
paramSetup_PCIE.srcAddr = (Uint32)srcAddr;
paramSetup_PCIE.dstAddr = (Uint32)PCIEAddr;
/* Setup EDMA Channel */
/* Enable the EDMA Channel */
/* Wait for a transmit completion */
```

11.14.4.11.3.2 Memory Read Transfer

Example 11-9 details how to perform an EDMA transfer from PCIe memory to the device destination (for example, L2 or EMIF). For the EDMA configuration, see Chapter 10, *Enhanced Direct Memory Access (EDMA) Controller*.

Example 11-9. Memory Read

```
/* Configure PCIe SS module and prepare transfer parameters*/
/* Initialize the PCIe SS module */
/* Define payload size and buffer size */
PCIE_max_payload = 128; /* Assume PCIe maximum payload size is 128 bytes */
buff_size = 2048; /* Assume the transfer buffer size 2048 bytes */
/* Define PCIe data starting address for EDMA transfer */
PCIeAddr = <PCIe data starting address defined in the device Data Manual>
...
/* Setup EDMA module and enable the DMA region */
/* Setup EDMA PARAM */
/* The OPT register contains following bit fields:
ITCCHEN = 0x0; TCCHEN = 0x0; ITCINTEN = 0x0; TCINTEN = 0x1;
TCC = 0x0; TCCMODE = 0x0; FWID = 0x0; STATIC = 0x0; SYNCDIM = 0x1; DAM = 0x0; SAM = 0x0 */
paramSetup_PCIE.option = CSL_EDMA3_OPT_MAKE(0,0,0,1,0,0,0,0,1,0,0);
paramSetup_PCIE.aCntbCnt = CSL_EDMA3_CNT_MAKE(PCIE_max_payload,
buff_size/PCIe_max_payload);
```

Example 11-9. Memory Read (continued)

```
paramSetup_PCIE.srcDstBidx = CSL_EDMA3_BIDX_MAKE(PCIE_max_payload,PCIe_max_payload);
paramSetup_PCIE.srcDstCidx = CSL_EDMA3_CIDX_MAKE(0,0); paramSetup_PCIe.cCnt = 1;
paramSetup_PCIE.linkBcntrld = CSL_EDMA3_LINKBCNTRLD_MAKE(CSL_EDMA3_LINK_NULL,0);
paramSetup_PCIE.srcAddr = (Uint32)PCIeAddr;
paramSetup_PCIE.dstAddr = (Uint32)dstAddr;
/* Setup EDMA Channel */
/* Enable the EDMA Channel */
/* Wait for a transmit completion */
```

11.14.4.12 PCIe Transactions

11.14.4.12.1 Transaction Requirements

There are some requirements that must be adhered to when issuing transactions to the PCIe SS internal bus interface.

11.14.4.12.1.1 Bus Mastering

An end point that is capable of becoming a bus master and initiate transactions in upstream direction must have its Bus Master Enable bit set in configuration space registers (PCIE_STATUS_COMMAND[BUS_MASTER]). If transactions are initiated (while assuming the role of a RC or EP) before enabling bus mastering capability, then transactions will not start until the bus master enable is set. There is no timeout or error response generated when transactions does not go out because of bus master bit not being set.

11.14.4.12.1.2 Address Alignment Requirements

The limit on transaction size on the internal bus interface is 128 bytes. Therefore, the transactions must be 128 bytes or smaller on the internal bus interface. If a transaction received in the inbound direction crosses a 128-byte-aligned address, then the PCIe SS master interface can split such transaction into two transactions at the 128-byte boundary.

If the starting address is not aligned to an 8-byte boundary, then the maximum transaction size is reduced to 120 bytes. Unspecified behavior will occur if misaligned transactions in outbound direction are not limited to 120 bytes.

11.14.4.12.1.3 Burst Type Requirements

The PCIe SS does not support *fixed* or *wrap* burst types. Transactions that required any burst type other than incremental burst type will result in unspecified behavior, possibly bus lock up.

11.14.4.12.1.4 Read Interleaving

Read interleaving refers to the process of returning split read responses from multiple transactions. This implies that read data is not guaranteed to be sent in sequential order (data for one transaction to be sent completely before the next). PCIe core is guaranteed to not interleave read responses if the outbound read command/transaction size does not exceed the max transaction size configured in the PCIe core. Currently this is configured as 128 bytes.

11.14.4.12.1.5 Byte Strobe Requirements

For any type of write transactions, the byte enables can have only a single unbroken string of 1s. In other words, in a transaction, if a byte's write strobe is set, then all following bytes must have write strobe set until the last byte with write enabled. *Holes* or *0s* in between the byte enables are not allowed.

Because the internal bus width is greater that 32-bit, the TLP size will not be 1 (PCIe counts in 32-bit units) and therefore, it is through the FBE/LBE (First/Last Byte Enable) that the actual data transfer size is controlled.

11.14.4.12.2 Support for Endian Modes

The PCIe core supports only little-endian mode (at the pins). The PCIe SS of the device also supports only little-endian mode.

11.14.4.13 Operations

11.14.4.13.1 PCIe as Root Complex

11.14.4.13.1.1 Initialization Sequence

The initialization sequence is as follows:

1. Make sure PLL reference clock is running.
2. Turn on the power domain and clock domain of PCIe module. See Section 5.2, *Power Management*, for more details.
3. Set PCIE_DEV_TYPE to 0b10 in the device level register BOOTCFG_DEVCFG to operate the PCIe SS in RC mode, see Section 5.1, *Control Module (BOOT_CFG)*.
4. Enable PLL using PCIE_PHY_PLL_CTRL register, see Section 11.14.4.8.1, *Enabling the PLL*, for details.
5. Wait until PLL is locked by sampling PLL_OK bit in the PCIE_PHY_PLL_CTRL register. See Section 11.14.4.8.1, *Enabling the PLL*, for details.
6. Disable link training by de-asserting the LTSSM_EN bit in the PCIe SS Command Status Register (PCIE_CMD_STATUS[LTSSM_EN] = 0). Upon reset, the LTSSM_EN is de-asserted automatically by hardware.
7. Program the configuration registers in the PCIe SS to desired values.
8. Initiate link training can be initiated by asserting LTSSM_EN bit in the PCIE_CMD_STATUS register (PCIE_CMD_STATUS[LTSSM_EN] = 1).
9. Insure link training completion and success by observing LTSSM_STATE field in PCIE_DEBUG0 register change to 0x11.
10. In conjunction with the system software, start bus enumeration and setup configuration space on downstream ports.
11. Continue software handshake and initialization on the remote devices. This includes setting up DMA protocols, interrupt procedures, and so forth.
12. With the completion of software initialization, DMA accesses can be started on various end points.

11.14.4.13.1.2 Configuration Accesses

Configuration accesses are made by RC port to individual function in each downstream EP device to program the PCIe specific operating parameters. In particular, the configuration accesses are used to allocate memory ranges for each downstream device, configure those memory ranges as I/O or Memory type, enable bus master capability on the device if necessary and also build a software database of PCIe attributes and capabilities of each downstream device. Each downstream device can be configured through the configuration access region of the PCIe SS. The PCIe SS converts memory reads and writes on the configuration region of internal bus interface into configuration access on the serial link.

11.14.4.13.1.3 Memory Accesses

There are two types of memory accesses – inbound and outbound memory accesses. The outbound memory accesses that are initiated by the DMA on the PCIe SS slave port and the inbound memory accesses that are initiated by the PCIe SS master port targeted to internal memory regions within the internal bus interconnect.

The outbound PCIe memory read and write accesses are made through the PCIe SS slave port through the device memory range that is dedicated to data transfers. The read and write transactions on this region are directly mapped to PCIe space by the PCIe SS in conjunction with the outbound address translation mechanism.

The completions to the bus transactions are generated by the PCIe SS when it receives completions from remote devices. In case of errors or timeouts, an error response is provided. For reads, the error responses span as many phases as there would be data phases if the error had not occurred.

The inbound memory accesses received by the PCIe SS on the serial link are initiated by remote PCIe end points that are capable of bus mastership. Such accesses are converted to bus transactions on the PCIe SS master port and once a response/completion is received, the PCIe SS sends data/response back to the PCIe bus master over the serial links. Typically, the inbound accesses will be targeted to memory space resident on the bus side of the RC port. The locations to which inbound memory accesses map are determined by software. The software must inform the remote devices about what protocol is to be followed so that the remote device will access the relevant memory regions to read or write data/control information. Not all end points have the capability to initiate inbound accesses.

An inbound access cannot cross a 4-KB boundary as per PCIe Specifications. In addition, any access that spans a 128-byte boundary in the device memory map can get split into two transactions at the 128-byte boundary.

11.14.4.13.1.4 I/O Accesses

The PCIe SS supports outbound I/O accesses in RC mode. These accesses are initiated through a 4-KB memory space and a programmable register. All accesses made to the 4-KB space become I/O accesses and the I/O base register determines the target address for such accesses. The I/O accesses cannot be for more than 32-bits of data aligned at a 4-byte boundary. See Chapter 2, *Memory Map*, for location of the I/O access address range. Inbound I/O accesses are not supported in the PCIe SS RC mode.

11.14.4.13.2 PCIe as End Point

11.14.4.13.2.1 Initialization Sequence

Upon de-assertion of reset, the PCIe SS is configured as end point by chip-level setting in DEVCFG register, see Section 5.1, *Control Module (BOOT_CFG)*. Before a root complex is allowed to access the configuration space of the end point, the following initialization sequence should be followed:

1. Make sure PLL reference clock is running.
2. Turn on the power domain and clock domain of PCIe module. See Section 5.2, *Power Management*, for mare details.
3. Set PCIE_DEV_TYPE to 0b0 in the device level register BOOTCFG_DEVCFG to operate the PCIe SS in RC mode, see Section 5.1, *Control Module (BOOT_CFG)*.
4. Enable PLL using PCIE_PHY_PLL_CTRL register, see Section 11.14.4.8.1, *Enabling the PLL*, for details.
5. Wait until PLL is locked by sampling PLL_OK bit in the PCIE_PHY_PLL_CTRL register, see Section 11.14.4.8.1, *Enabling the PLL*, for details.
6. Disable link training by de-asserting the LTSSM_EN bit in the PCIe SS Command Status Register (PCIE_CMD_STATUS[LTSSM_EN] = 0). Upon reset, the LTSSM_EN is de-asserted automatically by hardware.
7. Program the configuration registers in the PCIe SS to desired values.
8. Initiate link training can be initiated by asserting LTSSM_EN bit in the PCIE_CMD_STATUS register (PCIE_CMD_STATUS[LTSSM_EN] = 1).
9. Insure link training completion and success by observing LTSSM_STATE field in PCIE_DEBUG0 register change to 0x11.
10. If further configuration register initialization is required, the application request retry bit (APP_RETRY_EN) should be set in PCIE_CMD_STATUS register. This will lead to incoming accesses to be responded with the retry response. This feature allows slow devices extra time before the root port assumes the devices to be inactive. Once programming is complete, de-assert the APP_RETRY_EN bit to allow transactions from the root complex
11. Once configuration setup is complete, DMA transactions can begin.
12. Inbound PCIe transactions will arrive at the master port of the PCIe SS end point. These will be responded to by the target slave devices and the PCIe SS will relay the response back to the PCIe device that initiated the transaction.
13. Outbound PCIe transactions will be targeted to the slave port of the PCIe SS end point. These transactions will be serviced only if the PCIe SS end point has been given bus master capability by the

root complex.

11.14.4.13.2.2 Configuration Accesses

As an end point, the PCIe SS can be only a target of configuration accesses from upstream. Until the link is established and APP_RETRY_EN is disabled, the PCIe SS will not respond to configuration accesses. When enabled, the PCIe SS will respond to configuration accesses automatically and these accesses do not get relayed to the master interface on the device interconnect side.

End point is not capable of accessing configuration space of devices other than its own. The system software can initialize the read-only fields in the PCIe SS configuration space via the slave port before the link training has been initiated. Once the configuration is complete by the PCIe root complex, the system software should not modify the PCIe SS configuration parameters. There is no explicit prevention mechanism in hardware to disallow such accesses though.

11.14.4.13.2.3 Memory Accesses

There are two types of memory accesses – inbound and outbound memory accesses.

An inbound access is typically initiated by a RC port or by another EP that is reaching the PCIe SS via a PCIe switch that supports peer-to-peer access. In either case, the incoming PCIe transaction results in an access on the PCIe SS master port. The response to such accesses is relayed back to the originating PCIe device.

In an outbound access, a DMA module or a host CPU initiates a read/write access on the PCIe SS slave port. This request is converted into a PCIe memory read/write transaction over the PCIe link. Once the PCIe SS receives completion from the remote device, it generates a completion on the internal bus slave port. The software/hardware that initiates the request on the slave port must perform it in a memory region that has been determined previously through software protocols. For example, the software may get information about applicable memory regions from the software that is running on the root complex device.

11.14.4.13.2.4 I/O Accesses

The PCIe SS in EP mode does not support inbound or outbound I/O accesses.

11.14.4.13.3 Accessing Read-Only Registers in Configuration Space

Some of the register fields provided in the configuration space can be written to prior to bus enumeration via the slave interface of the PCIe SS.

NOTE: The hardware does not prevent modification of read-only fields after bus enumeration but it is strongly advised that read-only fields not be written once the bus enumeration is complete.

In addition, the BAR Mask registers can also be programmed by first enabling the DBI_CS2 bit in PCIE_CMD_STATUS register and then performing writes on the BAR registers. The BAR Mask registers are overlaid on BAR registers. For the BAR Mask registers to be writable, the respective BAR must first be enabled.

11.14.4.13.4 Accessing EP Application Registers from PCIe RC

The application registers are also mapped at 2K and above address in the configuration space. The RC software can access these registers over the PCIe link provided the registers are programmed to some default values by the boot code in the PCIe SS host device that enables link training and TLP exchange.

11.14.4.14 PCIe Loopback

The PCIe Specifications provides loopback support in two ways:

- Through PIPE interface (Link Layer)
- Through PHY loopback capability (PHY Layer)

The PIPE interface loopback requires for two PCIe devices/components to attached to each other in a loopback master and a loopback slave configuration. A loopback master is the component requesting loopback. A loopback slave is the component looping back the data. Regardless with the PCIe SS role it assumes (RC or EP), it can assume a role of a loopback master or a loopback slave.

11.14.4.14.1 PIPE Loopback

The procedure depends upon whether the device is operating in RC or EP mode. In either case, the PCIe SS can be loopback master or loopback slave as outlined in the PCIe Specifications. These modes are to be used with PCIe test equipment only for symbol level loopback.

NOTE: PIPE loopback mode cannot be used for looping back transactions.

11.14.4.14.1.1 Loopback Master

The loopback path when PCIe SS is loopback master is:

1. PCIe SS
2. PIPE (TX)
3. PCIe Link
4. Loopback
5. PCIe Link
6. PIPE (RX)
7. PCIe SS

The loopback entry procedure when the PCIe SS is a loopback master:

RC Mode

1. Set Loopback Enable bit (LPBK_EN) in the Port Link Control Register (PCIE_PL_LINK_CTRL).
2. The link retraining sequence must be initiated by setting Retrain Link bit (RETRAIN_LINK) in the Link Status and Control Register (PCIE_LINK_STAT_CTRL).

EP Mode

1. Set Loopback Enable bit (LPBK_EN) in the PCIE_PL_LINK_CTRL Register.
2. Force the LTSSM to be in recovery state via the Link State field (LINK_STATE) in Port Force Link Register (PCIE_PL_FORCE_LINK).
3. Set Force Link bit (FORCE_LINK) in PCIE_PL_FORCE_LINK Register.

Once this is done, devices at the ends of a PCIe link enter the PCIe LTSSM loopback state. The initiator of loopback state is the loopback master and the other device is the loopback slave.

NOTE: Sending TLPs in Loopback Master EP mode and returning them via the loopback state of the other device are not possible.

11.14.4.14.1.2 Loopback Slave

The loopback path when the PCIe SS is loopback slave is:

1. Remote device
2. PCIe Link
3. PIPE (RX)

4. Loopback
5. PIPE (TX)
6. PCIe Link
7. Remote device

If the PCIe SS is a loopback slave, then the incoming serial data is routed back to the originating device from the PIPE interface as per PCIe loopback requirements. Typically, PCIe test equipment will be used as loopback master and it will transition the PCIe SS into the loopback slave state following which the inbound transactions will be looped back to the test equipment. There is no programming required on the PCIe SS to enter loopback in slave mode. PHY support is not required to use this loopback mode.

11.14.4.14.2 PHY Loopback

The PHY loopback is accomplished by switching the PHY to loopback where the transmitted data is looped back to the receive path at the PHY level. This mode can be used to perform TLP loopback even if there is no link partner. It is, however, possible to set up only when the PCIe SS is used in RC mode. This limitation is because of the fact that link training cannot occur between two upstream ports; at least one port must be a downstream port.

The loopback path in PHY loopback is:

1. Slave interface (TX)
2. PCIe SS
3. PHY (TX)
4. PHY (RX)
5. PCIe SS
6. Master Interface (RX)

The procedure is similar to the one for PIPE (Link Layer) interface but the PHY programming is used instead. This mode is entered by configuring the SERDES configuration registers. Both the Transmit and Receive paths must be set in loopback mode to enable PHY loopback mode.

PHY loopback configuration for the device:

- Follow the steps (up to Step 7) described in Section 11.14.4.13.1.1, *Initialization Sequence*, for RC mode.
- Before executing Step 8, the software should configure the SERDES configuration registers to set Transmit and Receive paths to be in loopback mode, see Section 11.14.5.1, *SERDES Configuration Registers*.
- Then initialize link training (Step 8 in Section 11.14.4.13.1.1, *Initialization Sequence*).
- Before checking link training status (Step 9 in Section 11.14.4.13.1.1, *Initialization Sequence*), it is required to skip Detect state in LTSSM and force link to begin with POLL_ACTIVE state.
  - Set 0x2 (POLL_ACTIVE state) in LNK_STATE field of PCIE_PL_FORCE_LINK Register PCIE_PL_FORCE_LINK[LINK_STATE] = 0x2).
  - Set 0x1 in FORCE_LINK field of PCIE_PL_FORCE_LINK register to force the link to the state specified by LINK_STATE field (PCIE_PL_FORCE_LINK[FORCE_LINK] = 0x1).
- Then insure link training completion (Step 9 in Section 11.14.4.13.1.1, *Initialization Sequence*).
- With the initialization above, the PCIe SS should be configured in PHY loopback mode.

For proper operation the following requirements should also be observed:

1. The PHY should be configured to operate in loopback mode before any transaction is sent out. Recommended approach is to set loopback before link training. Otherwise, any transactions that were not looped back will cause sequence numbers to increment on transmitter but not on receiver and all following transactions will be dropped because of sequence number mismatch.
2. The PCIE_BAR0 and PCIE_BAR1 values (if programmed) should not match the address for the transaction that is issued on slave interface. Otherwise, it will not get to the master port but to internal registers.
3. The memory Base/Limit registers should not be configured such that the address of the transaction lies

in the range specified by the memory Base/Limit registers. Otherwise, the transaction will be discarded as a misrouted packet.

4. It is not possible to use configuration type transactions in this mode as RC cannot be a target of configuration transactions. Such transactions will be invalid and loopback will not work.

11.14.4.15 Reset Considerations

The PCIe SS supports the conventional reset mechanism that is specified within the PCIe Specification. Both hardware and software resets are supported.

No support for the functional level reset is needed because the PCIe SS supports a single function.

11.14.4.15.1 Hardware Reset Considerations

11.14.4.15.1.1 Power-On Reset

The power-on reset is used as cold reset of the PCIe SS. The entire device is reset when power-on reset is asserted. The device level power-on reset acts as the power-on reset for the PCIe SS.

NOTE: The PCIe SS requires power-on reset in case the device software transitions the subsystem to certain power-down states that mandate power-on reset to exit.

11.14.4.15.1.2 System Reset

The PCIe system reset is a warm reset, which can be hard reset or soft reset. The hard reset resets the entire PCIe SS. Soft reset resets the entire PCIe SS except the sticky bits in registers and power management logic inside the PCIe. See Section 5.3, *Reset Management*, for details.

11.14.4.15.2 Software Reset Considerations

The PCIe SS module contains a software reset (INIT_RST) bit in the Reset Command Register (PCIE_RSTCMD) that is used to issue a hot reset. In general, this reset is software controlled procedure and can be issued only by a root complex in a PCIe network as the propagation is downstream only. As a result of a reset, the PCIe SS module register values go to their reset state except the sticky bits. The default states of the registers are shown in Section 11.14.5, *PCIe Registers*.

NOTE: After a reset, the software must wait a determinate period of time before attempting any PCIe transaction on the device that has been reset. See the device Data Manual for the concrete value.

11.14.4.15.3 Power Domain and Module State Transitions Considerations

Turning off the PCIe power domain and module state and turning them back on reset PCIe module to initial state. However, the user must ensure that all of the outbound and inbound traffic on PCIe link is stopped from both the local device and the remote device. If a system hang occurs from improperly turning off PCIe module, the user may need to apply device power-on reset to recover the system.

To turn on and off the power domain and module state, the user must follow the transition sequence and check the transition status bit field (GOSTAT), see Section 5.2.2, *Power Sleep Controller (PSC)*. There is no additional delay requirements from when the PCIe power domain is turned on until it is ready for use.

After turning PCIe module back on, the link training must be restarted following any internal configuration that software may need to perform.

11.14.4.16 Interrupt Support

The PCIe SS provides a total of fourteen interrupts to the device interrupt controller.

Both message signaled interrupt (MSI) and legacy interrupt are handled by the PCIe SS. When operating as an EP, the PCIe SS is capable of generating MSI or legacy interrupt, depending on the role it is assuming.

NOTE: One PCIe component can not generate both types of interrupts.

It is either one or the other. The interrupt type an EP generates is configured during configuration time. When operating as RC, the PCIe SS is capable of handling both MSI and legacy interrupts. This is because when operating as RC it should be able to service both PCIe end points as well as legacy end points.

11.14.4.16.1 Interrupt Allocation

Multiple events from the PCIe SS are mapped to CPU subsystem in the device. The PCIe SS provides a total of fourteen interrupts to the device interrupt controller. Some of the events are meaningful based on the role the PCIe SS assumes (RC or EP). The interrupts and their underlying events are listed below.

Table 11-2571. PCIe SS Interrupt Events

Interrupt Event Number	Interrupt Description
0	PCIe legacy interrupt mode – INTA (RC mode only)
1	PCIe legacy interrupt mode – INTB (RC mode only)
2	PCIe legacy interrupt mode – INTC (RC mode only)
3	PCIe legacy interrupt mode – INTD (RC mode only)
4	MSI interrupts 0, 8, 16, 24 (EP/RC modes)
5	MSI interrupts 1, 9, 17, 25 (EP/RC modes)
6	MSI interrupts 2, 10, 18, 26 (EP/RC modes)
7	MSI interrupts 3, 11, 19, 27 (EP/RC modes)
8	MSI Interrupts 4, 12, 20, 28 (EP/RC modes)
9	MSI Interrupts 5, 13, 21, 29 (EP/RC modes)
10	MSI Interrupts 6, 14, 22, 30 (EP/RC modes)
11	MSI Interrupts 7, 15, 23, 31 (EP/RC modes)
12	Error Interrupts • [0] System error (OR of fatal, nonfatal, correctable errors) (RC mode only) • [1] PCIe fatal error (RC mode only) • [2] PCIe non-fatal error (RC mode only) • [3] PCIe correctable error (RC mode only) • [4] AXI Error due to fatal condition in AXI bridge (EP/RC modes) • [5] PCIe advanced error (RC mode only)
13	Power management and reset event interrupts • [0] Power management turn-off message interrupt (EP mode only) • [1] Power management ack message interrupt (RC mode only) • [2] Power management event interrupt (RC mode only) • [3] Link request reset interrupt (hot reset or link down) (RC mode only)

11.14.4.16.2 Interrupt Generation in EP Mode

When the PCIe SS is operating as an EP, either the legacy interrupts or the MSI interrupts can be triggered to the upstream ports (eventually leading to an interrupt in RC device). As per PCIe Specifications, each PCIe function may generate only one of the legacy or MSI interrupt types as decided during configuration period.

11.14.4.16.2.1 Legacy Interrupt Generation in EP Mode

The end point can trigger generation of a PCI legacy interrupt at the root complex via an in-band Assert_INTx/Deassert_INTx message. The actual interrupt that is generated on RC port is based on the configuration of the EP that generates the interrupt and it could be one of INTA, INTB, INTC, or INTD. See the interrupt-related registers in configuration space registers for details Section 11.14.5.3, *Application Registers*.

To generate an interrupt, following steps are required:

1. Legacy interrupt generation should be enabled via PCIE_LEGACY_A_IRQ_ENABLE_SET, PCIE_LEGACY_B_IRQ_ENABLE_SET, PCIE_LEGACY_C_IRQ_ENABLE_SET or PCIE_LEGACY_D_IRQ_ENABLE_SET.
2. Write 0x1 to PCIE_EP_IRQ_SET register to enable the legacy interrupt
3. An assert INTA/B/C/D message is automatically sent.
4. Write 0x1 to PCIE_EP_IRQ_CLR register to disable the legacy interrupt by sending a deassert INT A/B/C/D message.

Once an assert message has been generated, it cannot be generated again until a deassert message is generated. Thus, only one interrupt can be pending at a time. The pending status can be checked in PCIE_EP_IRQ_STATUS register.

There is no hardware input port provided that allows generation of legacy interrupts on the EP port.

NOTE: The interrupt messaging mechanism makes it unfeasible to guarantee a time of delivery of the interrupt unlike in conventional designs where the interrupt line is often electrically connected to the final destination.

11.14.4.16.2.2 MSI Interrupt Generation in EP Mode

MSI interrupts are generated by a PCIe 32-bit memory write transaction to a pre-determined address with pre-determined data. The PCIe system software configures the address and the data that is to be used in the memory write transaction at the time of initialization of the EP device. The MSI scheme supports multiple interrupts and each device can request up to 32 interrupt vectors even though the allotted interrupts may be less than the requested number.

To generate MSI interrupts, the following steps are required:

1. Ensure that the MSI support has been enabled in the device (Set MSI_EN bit in MSI Capabilities Register (PCIE_MSI_CAP). Legacy interrupt must be disabled).
2. Read the value of the MSI Address Register in the local PCIe configuration space (Read the value of MSI Lower 32 Bits Register (PCIE_MSI_LOW32) for 32-bit addressing or MSI Upper 32 Bits Register (PCIE_MSI_UP32) and MSI_LOW32 together for 64-bit addressing (64BIT_EN bit is enabled in PCIE_MSI_CAP register)).
3. Read the value of the MSI Data Register in the local PCIe configuration space (Read the value of MSI Data Register (PCIE_MSI_DATA)).
4. Determine the number of MSI vectors allocated (and the number requested) to the device.
5. Depending upon the number of MSI interrupts allocated, issue a memory write transaction with the address the same as PCIE_MSI_LOW32 and PCIE_MSI_UP32 and the data the same as the PCIE_MSI_DATA. In the data, the LSBs can be modified to reflect the appropriate MSI event that needs to be notified to root complex.
6. The memory write transaction can also be optionally routed through the outbound address translation interface if the destination PCIe address is not directly accessible.

The memory write transactions to generate MSI interrupts in RC are actually targeted at PCIE_MSI_IRQ register. The MSI interrupt is generated as a result of the one of 32 events that is triggered by a write of MSI vector value to PCIE_MSI_IRQ register in RC. Before the end point devices can issue MSI interrupts, the MSI address and data registers must be configured by system software to make sure the PCIE_MSI_IRQ registers could be accessed correctly with proper MSI vector value.

If there is no system software supported, the user application needs to make sure EP could issue memory write transaction to PCIE_MSI_IRQ register in RC with proper MSI vector value to generate MSI interrupt in RC.

For more details about how MSI interrupts are expected to behave, see the PCIe standard specifications.

As per the PCIe Specification, an EP device can generate MSI interrupts only to RC. However, the PCIe SS has a provision to allow generation of MSI interrupts from an EP to another EP. To generate an interrupt to another EP, instead of doing the memory write to a register in RC memory space, an EP can target this memory write to an analogous register in another EP device that integrates a PCIe SS. This memory write would be to the PCIE_BAR0 memory space where all registers are located.

NOTE: There is no hardware input port to allow generation of MSI interrupts on the EP port.

11.14.4.16.3 Interrupt Generation in RC Mode

In accordance with PCIe base specifications, root complex ports only receive interrupts. There is no mechanism to generate interrupts from RC port to EP mode as per PCIe Specification. However, the PCIe SS does support generation of interrupts from RC to EP. The behavior is similar to generation and reception of MSI interrupts in RC mode except for the fact that this functionality is enabled in EP mode as well.

The RC device can perform a memory write into the PCIE_MSI_IRQ register over the PCIe link to generate one of 32 EP interrupts.

NOTE: The PCIe SS follows PCIe MSI rules and does not necessarily accumulate multiple writes to the same MSI vector. Only one of such writes is guaranteed to be processed and subsequent writes on the same vector arriving before the interrupt status is cleared may be lost.

11.14.4.16.4 Interrupt Reception in EP Mode

The PCIe Specification does not have provision for end points to receive legacy interrupts. As a result, only events other than these are used to map to interrupts. The MSI interrupts are not supported on EP devices as per PCIe Specification but the PCIe SS does support these interrupts. The MSI interrupt is generated as a result of the one of 32 events that is triggered by a write of MSI vector value to PCIE_MSI_IRQ register in EP.

These interrupts, delivered via register writes over the serial link, could also be coming from another end point that performs a write to the appropriate interrupt registers in the EP's PCIE_BAR0 space. It is up to software designers to implement a way to determine the actual source of the interrupt.

Among the 32 MSI interrupt events, every four MSI interrupts share the same interrupt event number in the PCIe SS, which could generate interrupt event to CPU. See Table 11-2571 for details. The MSI interrupts can be enabled by setting corresponding bits in PCIE_MSI0_IRQ_ENABLE_SET to PCIe_MSI7_IRQ_SET registers. They can be disabled by setting corresponding bits in PCIE_MSI0_IRQ_ENABLE_CLR to PCIE_MSI7_IRQ_ENABLE_CLR registers.

When EP receives the MSI interrupt, the corresponding status bit is set in PCIE_MSI0_IRQ_STATUS to PCIE_MSI7_IRQ_STATUS registers. Before the same MSI interrupt event to be generated again, the MSI interrupt status should be cleared by writing one to the corresponding bit in PCIE_MSI0_IRQ_STATUS to PCIE_MSI7_IRQ_STATUS registers and writing the interrupt event number in PCIE_IRQ_EOI register to indicate the end of the interrupt event. See Table 11-2571 for the interrupt event number of each MSI interrupt.

11.14.4.16.4.1 Hot Reset Request Interrupt

When the link is down, the upstream port may request reset of the end point. This request is terminated as an interrupt to the end point host software. The PCIe SS automatically disables LTSSM by de-asserting the LTSSM_EN bit in the PCIE_CMD_STATUS register and suspends LTSSM in the *detect quiet* state. All outstanding transactions are errored out on the slave port and further transactions are not generated on the master port. Once the transactions are completely stopped through the internal bus disconnect protocol, the software should issue a local reset to the PCIe SS. The re-initialization process may then be started.

11.14.4.16.5 Interrupt Reception in RC Mode

11.14.4.16.5.1 Legacy Interrupts Reception in RC Mode

Any of the four legacy interrupts may be generated by the PCIe SS. Each interrupt can originate from multiple end point devices. The software should service these by probing the interrupt registers in each downstream device's configuration space. When all devices have been serviced, the last device serviced will send an interrupt deassert message, which will clear the interrupt.

The interrupt request signal at the PCIe SS boundary is a pulse signal that is triggered each time an assert interrupt message is received. The interrupt pending signal is a level signal that is high as long as the interrupt has not be serviced and the interrupt status not cleared through a register write.

The traditional EOI procedure, although implemented, may not operate as expected if the deassert message arrives after the EOI has been issued. This cannot be corrected but only worked around by doing a read on the interrupt register downstream before issuing an EOI. If the EOI register is written to before the deassert message has reached the interrupt logic, the interrupt pending status will not have cleared and the interrupt will retrigger.

In addition, the software drivers for downstream devices must ensure that the data transaction that is expected to complete before interrupt is triggered in RC device's CPU has completed. Because PCIe write transactions are posted, it is not necessary that a write from EP to system memory in RC has completed before a write to MSI interrupt generation register has completed. This could create a potential race condition.

The reason is that the PCIe RC or switch could reorder the memory write transactions just posted ahead of previously posted memory write transactions or message transactions. Similarly, message transactions just posted may be ordered ahead of previously posted memory write or message transactions due to the relax ordering feature in PCIe. The relaxed ordering could improve the performance of the post transactions while it could potentially cause issues if the reordering is not permitted in some circumstances.

In the RC mode, the user could disable the relaxed ordering feature by clearing the RELAXED field to 0 in DEV_STAT_CTRL register. The RC could also clear RELAXED field to 0 in TLPCFG register to disable the reordering request for all outgoing TLPs.

11.14.4.16.5.2 MSI Interrupts Reception in RC Mode

A total of 32 MSI interrupts can be generated from one or more downstream devices. These MSI interrupts represent the MSI interrupts that have been allocated to various downstream devices during PCIe configuration/enumeration procedure. The MSI interrupt is generated as a result of the one of 32 events that is triggered by a write of MSI vector value to MSI_IRQ register in RC.

Among the 32 MSI interrupt events, every four MSI interrupts share the same interrupt event number in the PCIe SS, which could generate interrupt event to CPU. See Table 11-2571 for details. The MSI interrupts can be enabled by setting corresponding bits in PCIE_MSI0_IRQ_ENABLE_SET to PCIE_MSI7_IRQ_ENABLE_SET registers. They can be disabled by setting corresponding bits in PCIE_MSI0_IRQ_ENABLE_CLR to PCIE_MSI7_IRQ_ENABLE_CLR registers.

When RC receives the MSI interrupt, the corresponding status bit will be set in PCIE_MSI0_IRQ_STATUS to PCIE_MSI7_IRQ_STATUS registers. Before the same MSI interrupt event to be generated again, the MSI interrupt status needs to be cleared by writing one to the corresponding bit in PCIE_MSI0_IRQ_STATUS to PCIE_MSI7_IRQ_STATUS registers and writing the interrupt event number in PCIE_IRQ_EOI register to indicate the end of the interrupt event. See Table 11-2571 for the interrupt event number of each MSI interrupt.

Each end point can use either MSI interrupts or legacy interrupts, but not both at the same time. MSI interrupts have the same race condition hazard as the legacy interrupts. Hence, software drivers should take precaution.

11.14.4.16.5.3 Advanced Error Reporting Interrupt

If enabled by software at the time of enumeration, the PCIe SS will generate this interrupt to indicate occurrence of errors of various levels of severity. The software can choose not to enable advanced error reporting. But if it is enabled, it must process the interrupt as per PCIe Specifications. The error message reporting enable bit must be set in the Root Error Command Register (PCIE_RC_ERR_CMD) to generate this interrupt. This interrupt can originate either via an MSI message or via an INTx internally in the root complex.

11.14.4.16.6 Interrupt Reception in RC and EP Mode

These interrupts are common to both RC and EP modes of operation.

11.14.4.16.6.1 Link down Interrupt

If the PHY link is disconnected, this interrupt will be generated. The expected course of action is to reset the entire PCIe SS subsystem and restart. All application states must also be initialized so that the operations can resume following the reset and renegotiation of the link.

11.14.4.16.6.2 Transaction Error Interrupts

If there is a timeout on PCIe or an abort, this interrupt will be issued. These errors should be handled based upon the severity of the error. For more details, see the PCIe standard specifications.

11.14.4.16.6.3 Power Management Event Interrupt

This interrupt is generated to let the software know of power management events.

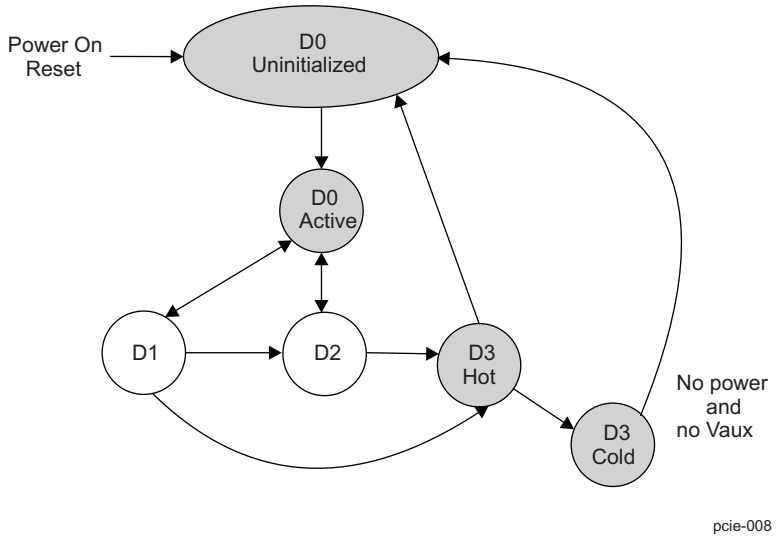
11.14.4.17 Power Management

PCIe has multiple power management protocols. Some of them are invoked by the hardware, such as Active State Power Management (ASPM), while others are activated at higher levels via software.

11.14.4.17.1 Device Power Management

The PCIe protocol is compatible with all PCI power management functionality. The power states specified are D0, D1, D2, D3hot, and D3cold. All functions must support D0 and D3 states.

Figure 11-1097. PCIe Power Management State Transitions



11.14.4.17.1.1 D0 State

Upon completion of a reset such as a power-on or hot reset, the function is considered to be in the D0 uninitialized state. Once the function is enumerated, configured, and one or more of the memory spaces are enabled, and I/O space enable or bus master enable bits are set, it is considered to be in D0 active state. The D0 active state is the full-operation state of a PCIe function.

11.14.4.17.1.2 D1 State

Support for the D1 state is optional and is primarily driven by software. It is considered to be a light sleep state that provides some power savings compared to D0 state while still allowing transition to D0 state. While in the D1 state, a function must not initiate any request TLPs on the link except for a power management event (PME) message. Configuration and message requests are the only TLPs accepted by a function in the D1 state. All other received requests must be handled as unsupported requests, and all received completions may optionally be handled as unexpected completions. A function's software driver participates in the process of transitioning the function from D0 to D1. It contributes to the process by saving any functional state (if necessary), and otherwise preparing the function for the transition to D1. As part of this quiescence process the function's software driver must ensure that any mid-transaction TLPs (that is requests with outstanding completions), are terminated prior to handing control to the system configuration software that would then complete the transition to D1.

11.14.4.17.1.3 D2 State

Support for the D2 state is optional in PCIe and is primarily driven by software. It provides significant power savings while still retaining capability to transition back to a previous condition. In this state, the PCI function can only initiate power management events and it can only respond to configuration accesses.

11.14.4.17.1.4 D3hot and D3cold State

All PCIe functions are required to support D3 state. In the D3hot state, power is still present while in D3cold there is no power. Devices in the D3hot state may be transitioned back to the D0 state while the devices in the D3cold state need re-initialization to be brought back to the D0 state.

NOTE: All PCIe devices support D3cold state. This state is a power off state.

Functions in D3hot respond to configuration space accesses as long as power and clock are supplied so that they can be returned to D0 by software. When programmed to D0, the function may return to the D0 Initialized or D0 Uninitialized state without PCIe reset being asserted. There is an option of either performing an internal reset or not performing an internal reset. If not performing an internal reset, upon completion of the D3hot to D0 Initialized state, no additional operating system intervention is required beyond writing the power state bits. If the internal reset is performed, devices return to the D0 Uninitialized and a full re-initialization is performed on the device. The full re-initialization sequence returns the device to D0 Initialized, where normal operation may resume.

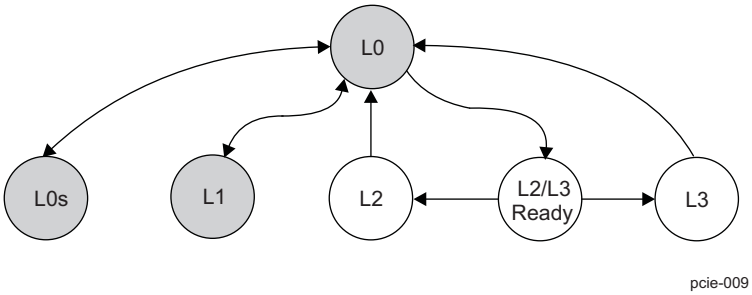
11.14.4.17.2 Link State Power Management

There are multiple states for the physical layer – L0, L0s, L1, L2, and L3 states, that provide incrementally higher power savings as the states transition from L0 toward L3. The PCIe SS supports the L0, L0s, and L1 states. The L3 state is a power–off state and is supported by default.

NOTE: The PCIe SS does not support L2 power down states.

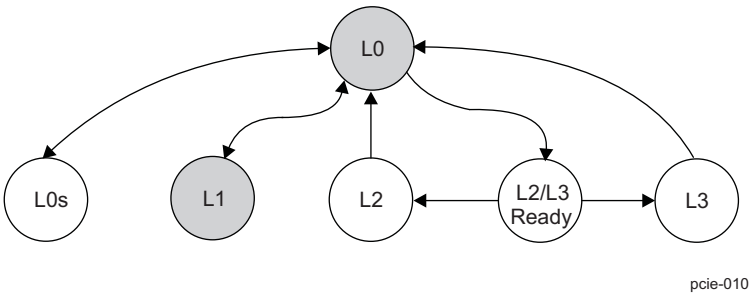
Some of the link power states are autonomously managed by hardware – ASPM. Figure 11-1098 illustrates the transitions between states L0, L0s, and L1 that are managed by ASPM. The transitions between gray states are allowed in ASPM. These transitions can be enabled through registers in the PCIe configuration space to allow either just L0s transition or both L0s and L1 transitions.

Figure 11-1098. ASPM Link State Transitions



The L1 power state can also be entered via software control. When the device power state is D1, D2, or D3hot, then the link state must transition to L1 state. Figure 11-1099 shows such state transitions.

Figure 11-1099. Software–driven Link Power State Transition



11.14.4.17.2.1 L0s State

L0s is a lower power state enabled by ASPM. Each device can control its L0s transition on its transmitter. Receiver side is controlled by the remote device.

11.14.4.17.2.2 L1 State

The L1 state is reached either through ASPM timer mechanism or via the software initiated transition to a D state other than the D0 state. In the L1 state, the link is in an idle state and both receiver and transmitter in devices on both ends of a link are able to conserve energy.

11.14.4.17.2.3 L2/L3 Ready State

This state is a transitional state reached from L1, from which the link must either transition to the L2 or to the L3 state. The L3 state is a full power off state. The L2 state is also a power off state except that the wake signal can be used by end point devices to request power and clock from the system.

11.14.4.17.2.4 L2 State

The L2 state is appropriate when a device needs to monitor an external event while in deep power down mode. In the L2 state, auxiliary power is supplied and a minimal amount of current is drawn from the power source. Almost all of the logic is without power. To recover, the wake signal is used.

11.14.4.17.2.5 L3 State

In this state, device does not supply any power to the PCIe fabric. There is no mechanism to communicate in the L3 state. To recover, the system must re–establish power and reference clock followed by a fundamental reset.

11.14.4.17.3 Relationship Between Device and Link Power States

The device D–states are correlated to the link power states. For each D state, there are specific states that the PCIe link or interconnect can transition to. Table 11-2572 shows the permissible state combinations.

Table 11-2572. Device and Link Power States Combinations

Downstream Device State	Permissible Upstream Device State	Permissible Link State
D0	D0	L0 (required), L0s (required), L1 ASPM (optional)
D1	D0 – D1	L1
D2	D0 – D2	L1
D3hot	D0 – D3hot	L1, L2/L3 Ready
D3cold	D0 – D3cold	L2aux, L3

11.14.4.17.4 CBA Power Management

The power management for PCIe SS WITH CBA interface is primarily accomplished through the clock stop protocol. Any time the clock stop protocol is initiated, the PCIe SS will acknowledge after making sure that there are no outstanding transactions. If there are any pending transactions in the subsystem, then the clock stop acknowledgement procedure cannot be completed. Therefore, it is necessary that the system software first suspend activity on the serial link as well as on the CBA interface. To do so, it is expected that the devices communicating with the device on which clock stop procedure is to be performed agree to stop transactions targeted to the device in question. Similarly, it is required that the outgoing transactions also be stopped to successfully complete the clock stop sequence. The PCIe SS will guarantee that in the event there are pending transactions that are still in process of draining will prevent the clock-stop acknowledgement to be issued to CBA fabric power management logic.

NOTE: The PCIe SS does not have ability to terminate the clock stop state. Therefore, any wakeup sequence can only be initiated through other means such as software driven timers or software detected events occurring outside of PCIe SS.

11.14.4.17.5 PCIe SS Power Management

PCIe SS supports several device and link power management states. The sections below outline the sequence involved in entering and exiting these states that is specific to PCIe SS architecture. For each state, entry and exit conditions are described along with their correlation with CBA power management procedures and protocols.

11.14.4.17.5.1 L0 State

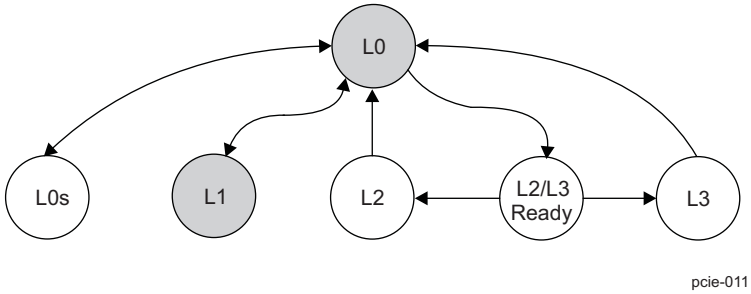
The state of normal operation (no power saving) is the L0 state. In this state, the transmitter and the received in the physical layer are both switched on and actively operating.

11.14.4.17.5.2 L0s State

The L0s transition is fully managed by PCIe SS as documented by the ASPM specifications. Based on a timer monitoring the activity on transmitter, the ASPM L0s state is reached whenever there has been no pending transmission for a specified period of time. The timeout interval for entering L0s can be controlled via L0S_ENTRY_LATENCY field in PCIE_ACK_FREQ register in Port Logic registers. The entry into L0s can be enabled or disabled by the AS_LINK_PM field in PCIE_LINK_CAP register PCIe Configuration space. The power saving in this state is due to the transmitters in SERDES being switch off. In the L0s state, there is no logic power saving except what is accomplished by lack of switching activity in the sequential logic.

Figure 11-1100 presents the ASPM L0s transaction.

Figure 11-1100. ASPM L0s Transition



11.14.4.17.5.2.1 Entry Conditions for ASPM L0s Stat

The following conditions must be met before L0s state is entered:

- ASPM L0s is enabled by the AS_LINK_PM field in PCIE_LINK_CAP register.
- The specified duration of time since last transmission has elapsed and no other higher state of power down is requested.

Whether or not a device enters L0s is configurable by Root Complex software.

11.14.4.17.5.2.2 Exit Conditions for ASPM L0s State

Any of the following conditions will bring the PCIe SS out of L0s state:

- The conditions to enter L1 state are met.
- A DLLP or TLP is pending to be transmitted. Since the device is in D0 state, there is no need for a PME message to be transmitted.
- The PCIe link partner requests link recovery.

11.14.4.17.5.2.3 CBA Power States

There is no correlation between CBA power management and L0s power state. If clock stop protocol has been initiated, the PCIe SS may acknowledge the clock stop request. If there have been any forthcoming transactions, PCIe SS is not able to service those if clock has been stopped and transactions have been received after that from the remote link partner.

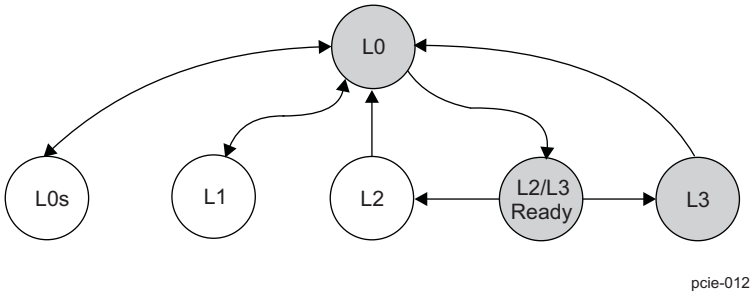
NOTE: It is not recommended to stop clocks in this state.

11.14.4.17.5.3 ASPM L1 State

The L1 state can also be reached by software controlled D1, D2 or D3 states. This section describes the ASPM L1 power down state. Both link partners are involved in this transition and negotiate to transition the link to L1 state. A link will transition back to L0 when there is a transaction level packet or Data link layer packet to be sent, application/software requests transition or if the link partner makes a request. The entry into L1 can be enabled or disabled by the ASPM Link Power Management Support field in the Link Capabilities Register and Link Control Register in PCIe Configuration space. The timeout value is controlled by the L1 Entry Latency field in Port Logic registers.

Figure 11-1101 presents the Link Power States L0 and L1.

Figure 11-1101. Link Power States L0 and L1



11.14.4.17.5.3.1 Entry Conditions for ASPM L1 State

There are two possible transition sequences for L1 state. All conditions listed must be met for entry to L1 to occur.

The first sequence is L1 Idle Timeout from L0s:

1. Both L1 and L0s modes of ASPM are enabled by ACTIVE_LINK_PM field in PCIE_LINK_STAT_CTRL register.
2. Link state is in L0s for both transmitter and receiver of the link. Alternatively, by setting ASPM_L1 bit in PCIE_ACK_FREQ, it is possible to override this and require that only the transmitter is in L0s before L1 transition is initiated.
3. There is no higher stage of power down requested and L1 enter conditions defined by PCIe spec regarding no pending TLPs or completions and availability of flow control credits are met.
4. A pre-specified period of time has elapsed in L0s state.

The second sequence is L1 Idle Timeout from L0:

1. ASPM L1 is enabled and L0s is not enabled in the PCIE_LINK_STAT_CTRL register.
2. Link state is in L0.
3. There is no higher stage of power down requested and L1 enter conditions defined by PCIe spec regarding no pending TLPs or completions and availability of flow control credits are met.
4. A pre-specified period of time has elapsed in L0 state.

11.14.4.17.5.3.2 Exit Conditions from ASPM L1 State

Any of the following conditions can occur to cause transition out of L1 state.

- Software initiates a higher stage of power down.
- Any pending DLLP or TLP.
- Link partner requesting exit from L1.

11.14.4.17.5.3.3 CBA Power States

In ASPM L1 state, the PCIe SS will acknowledge clock stop request if there are no pending transactions on CBA interface. The link is guaranteed to not have pending transactions in L1 state. Like in L0s state, clock stop procedure should be performed only if it is guaranteed that the CBA masters have already completed all transactions and are not going to schedule any new transactions. The same condition applies to remote device as well.

It is recommended that the clock stop sequence not be used without ensuring higher level protocols have been suspended. The ASPM states can be seen as a way to reduce system level power consumption in situations of transient inactive periods that are in order of microseconds at a time.

11.14.4.17.5.4 Software Initiated L1 State

The software initiated power management states in PCIe are managed via the Device states – D0, D1, D2 and D3. In L1 power down state, both link partners are involved and negotiate to transition to L1 state. A link will transition back to L0 when there is a transaction level packet or Data link layer packet to be sent, application/software requests transition or if the link partner makes a request.

11.14.4.17.5.4.1 Entry Conditions for L1 State on EP and RC

The L1 state is entered when the RC software writes the Power Management Control and Status Register (PMCSR) to change state of EP to D1, D2 or D3. The EP must be capable of transitioning to these states based on the values programmed in Power Management Capabilities Register (PMCAP).

The sequence followed to move PCIe SS EP to L1 state is:

- RC changes EP's state to D1, D2 or D3 through a Configuration Write to EP's PMCSR
- EP ensures the following conditions are true before it sends PM_ENTER_L1 DLLPs to RC
  - No new TLPs to be sent and any TLPs received will be discarded
  - Last TLP Ack received (retry buffer empty)
  - Flow Control Credits are available
- RC receives PM_ENTER_L1 DLLP
- RC ensures no new TLPs to be sent downstream and that last TLP ack received and FC credits available.
- RC sends PM_REQUEST_ACK continuously.
- Upon receiving PM_REQUEST_ACK, the EP transmits Electrical Idle ordered set. This indicates successful transition to L1 state.
- RC software may decide to either stay in D0 or switch to higher state such as D1 or D2.
- As an EP, any outbound transactions on PCIe SS slave port arriving after the PM_ENTER_L1 DLLP transmission has begun are blocked.

NOTE: If the software fails to ensure that configuration write to transition to a power saving D state is done only after all completions and read/write requests are completed, the EP may fail to process such transactions and may issue Error messages to the RC. This behavior is not a malfunction but it is compliance to PCIe protocol.

11.14.4.17.5.4.2 Exit Conditions for L1 State on EP

The following sequence outlines exit from L1 when in EP mode:

1. When the EP is in L1 mode, there are two possible ways to exit – if there is pending traffic on the slave port or if the link partner initiated transition back to L0 mode.

- If there is pending traffic on slave port, then the PCIe SS generates a PME message automatically. It is possible for a transaction to be generated on the link in response to a register write on the local PCIe memory map. In such cases as well, a PME message will be generated by PCIe SS automatically. The reception of PME message is used by RC software to change the D-state of the EP back to D0. Unless the D0 state is written by RC, the EP cannot send the TLP out on the serial link.
- Alternatively, if there is pending traffic in the inbound direction, the RC will first renegotiate link state to L0 on itself and on EP. Then, it will do a configuration write to switch the D-state back to D0. In the meantime, detecting transition out of electrical idle on the PHY, the PCIe SS will request the bus fabric to supply clocks if they were gated.

2. Once the EP and RC are back in D0 device state and L0 link state, the normal operation resumes.

11.14.4.17.5.4.3 Exit Conditions for L1 State on RC

The RC can exit L1 and D1/D2 state in the following sequence:

1. The execution of this step depends on the current state of RC:
   a. If the RC needs to transmit packets, it needs to get out of D1 state via a configuration write to its own config space.
   b. This step is skipped if RC stayed in D0 state.
2. The link is transitioned to L0.
3. The EP mode is changed by RC to D0 state.
4. The normal transfers can resume.

11.14.4.17.5.4.4 CBA Power States

After the link has transitioned to L1 as a result of software power state transition to D1 or a higher state, the clock stop sequence can be initiated. If there are no outstanding transactions, the PCIe SS will acknowledge and be prepared for removal of CBA clocks.

If there is activity on the wire and the link is no longer in electrical idle state (because of activity initiated by link partner), the PCIe SS will not be able to respond. The software architecture must comprehend this limitation and transition to this state only when a pre-determined wake sequence is established between the two link partners. Such wake sequence, implemented outside the PCIe SS, would ensure availability of clocks on both devices for resumption of normal operation.

To resume operation, the Root Complex can modify D-state on the End Point by initiating a configuration write. Alternatively, the software on End Point can request generation of PME message by writing to application registers. In response to this PME Message, the Root Complex will change the D-state of End Point through a configuration write. In either method, link would transition to L0 and be ready for normal operation.

NOTE: PCIe SS with CBA interface does not generate PME message automatically.

It is important to recognize that the software can successfully transition the PCIe SS to a software initiated power down mode followed by a clock stop request/acknowledge sequence only if prior steps have been taken to ensure that the activity on both ends of a PCIe link has been suspended. While the PCIe SS will guarantee that the clock stop request is not acknowledged if there are pending transfers, it cannot, on its own, force exit from D1 or higher states. For example, if the RC forces D1 state on an EP that has outgoing commands pending, the EP will not acknowledge clock stop request. Secondly, it will not automatically send a PME to the RC to let it know of pending transmission. So, it is important that the EP software driver running on RC implement safeguards against such situations.

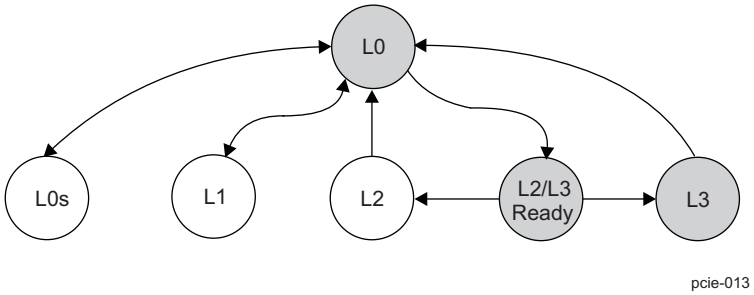
11.14.4.17.5.5 Software Initiated L2/L3 Ready state

The L2 and L3 states are reached via the L2/L3 ready state. Depending on whether the device has auxiliary power source, it will enter L2 state. Otherwise, it enters L3 state which is a full off power state.

NOTE: PCIe SS does not support L2 state with auxiliary power.

Figure 11-1102 presents the Link states L2/L3 Ready, L2 and L3 states.

Figure 11-1102. Link states L2/L3 Ready, L2 and L3 states



The L2/L3 Ready is a pseudo state. It is a transitional state for a PCIe device that leads to L2 or L3 state.

L2 state is reached when auxiliary power is available and the system controller switches the PCIe device from main power to auxiliary power on the PCIe connector.

L3 state is reached when auxiliary power is not available and the system controller switches a PCIe device off with no power supply at all.

11.14.4.17.5.5.1 Entry Conditions for L2/L3 Ready State on EP/RC

The entry conditions and sequence is as follows:

1. RC sets D3 state on all EP devices. This causes L0 –L1 transition on EP devices.
2. RC then sends PME_TURNOFF to EP device. This causes L1–L0 transition on EP devices to enable L2/L3 Ready handshake.
3. The message transmission causes the EP to wake from L1 and switch to L0
4. When the message is received, the PM_TURNOFF interrupt is generated. When the EP device is ready to get its PCIe SS shutdown, it must acknowledge the turn off message by asserting the ENTR_L23 bit in PCIE_PMCFG at some point in time. This time cannot be indefinite.
5. The PCIe SS EP automatically transmits PME_TO_ACK message while still in D3 state. The message is turned into PME_TO_ACK interrupt if PCIe SS is in RC mode and has enabled the interrupt.
6. Once the EP software has written the bit ENTR_L23, it is presumed that the PCIe SS is ready to get its power and reference clock removed. Then, the EP PCIe SS will automatically transmit a PM_ENTER_L23 DLLP. It is continued until the RC sends PM_REQUEST_ACK to EP.
7. Reception of PM_REQUEST_ACK causes the transition to L2/L3 Ready state.
8. The L3 state is reached if Aux power is not requested by the device.

11.14.4.17.5.5.2 Exit Conditions from L2/L3 on EP/RC

Since Wake from D3cold is not supported, the PCIe SS goes to L3 and the recovery to L0 is accomplished through supply of clock and power followed by a reset. System designers may be able to keep PCIe SS on a separate power domain and actually shut of power supply to PCIe SS. During recovery from L3, the power on reset must be provided to PCIe SS or some bits required to be reset for the recovery will not be reset.

11.14.4.17.5.5.3 CBA Power States

The L2/L3 Ready state is a preparation to turn off (completely power off) a PCIe device. It is expected that the PCIe SS will not be used for a long period of time (at least several minutes).

The clock stop request protocol can be executed after the L2/L3 state is reached. After the clock stop protocol is completed, power can be removed from PCIe SS.

To resume, the power should be restored, reset should be issued and link training restarted following any internal configuration that software may want to perform.

NOTE: The PCIe link activity must be stopped before preparation to turn the device off via L3/L3 Ready transition is performed. Without that, the clock stop request may not be acknowledged successfully.

11.14.4.17.5.6 Limitations and Exceptions

While the hardware makes best effort decisions for power state transitions, it is still susceptible to certain actions that are not in its control. Following is a list of events where unexpected behavior can occur.

11.14.4.17.5.6.1 L1 State

For PCIe SS variants that use GS70 technology, the SERDES does not support asynchronous loss of signal detect circuit. As a result, specifically in Gen2 mode, there is a dependency on clock for a squelch circuit in loss-of-signal-detect circuit. This dependency prevents the PCIe SS from correctly issuing interface wake signal. The implication of this limitation is that PCIe SS subsystems that operate in Gen2 mode should not use L1 power down state if it is expected that the PCIe SS would make a request to wake up via Interface Wake when there is activity on the link.

11.14.4.17.5.6.2 L2 and L3 States

NOTE: L2 state is not supported by PCIe SS.

Upon entry into L2/L3 Ready, the next state will be L3 which is a complete power off state. It is strongly recommended to transition to L3 only if required. A power cycle with a power on reset will be required for PCIe SS to recover to L0 state on resumption of power. The system software should use L1 state for power savings while the device is powered on.

PCIe SS does not have the ability to transmit in-band beacon because the SERDES does not support beacon.

11.14.4.17.5.6.3 PME Transmission

PCIe SS will transmit more than one PME message even though only one PME Message is expected to be transmitted. This is a known issue with the behavior of the third party PCIe IP and is not necessarily a malfunction.

11.14.4.18 Error Handling

11.14.4.18.1 Error Reporting

PCIe provides three mechanisms for establishing the error reporting policy. These mechanisms are controlled and reported through configuration registers mapped into three distinct regions of configuration space.

The various error reporting features are enabled as follows:

- PCI-compatible Registers — this error reporting mechanism provides backward compatibility with existing PCI compatible software and is enabled via the PCIE_CMD_STATUS register.
- PCIe Capability Registers — this mechanism is available only to software that has knowledge of PCIe. This required error reporting is enabled via the PCIE_DEV_STAT_CTRL register mapped within PCI-compatible configuration space.
- PCIe Advanced Error Reporting Registers — this mechanism involves registers mapped into the extended configuration address space. PCIe compatible software enables error reporting for individual errors via the Error Mask Registers.

PCIe includes two methods of reporting errors:

- Error message transactions — used to report errors to the host
  The conventional PCI reports errors via the PERR# and SERR# signals. But PCIe eliminates these error-related signals and error messages have been defined to replace these signals by acting as virtual wires. Furthermore, these messages provide additional information that could not be conveyed directly via the PERR# and SERR# signals. This includes identification of the device that detected the error and an indication of the severity of each error.
- Completion status — used by the completer to report errors to the requester
  PCIe defines a completion status field within the completion header that enables the transaction completer to report errors back to the requester.

11.14.4.18.2 Error Detection and Handling

This section defines the required support for detecting and handling PCIe errors.

11.14.4.18.2.1 PCI-Compatible Error Handling

Each PCIe must map required PCIe error support to the PCI-related error registers. This involves enabling error reporting and setting status bits that can be read by PCI-compliant software. The PCIe errors tracked by the PCI compatible registers are:

- Transaction Poisoning
- Completer Abort (CA) detected by a completer
- Unrecognizable Request (UR) detected by a completer

The PCI mechanism for reporting errors is the assertion of PERR# (data parity errors) and SERR# (unrecoverable errors). The PCIe mechanisms for reporting these events are via the split transaction mechanism (transaction completions) and virtual SERR# signaling via error messages.

The Status and Command Register (PCIE_STATUS_COMMAND) has error-related bits to understand the features available from the PCI-compatible point of view. While the Status and Command Register bits have the PCI name, some of the field definitions have been modified to reflect the related PCIe error conditions and reporting mechanisms. See Section 11.14.5.4.3, *PCIE_STATUS_COMMAND Register*, for the details of this register.

The following bits are set to enable baseline error reporting under control of PCI-compatible software.

- SERR Enable (bit 8)
- Parity Error Response (bit 6)

The following bits are set under certain error circumstances:

- Detected Parity Error (bit 31)
- Signalled System Error (bit 30)

- Received Master Abort (bit 29)
- Received Target Abort (bit 28)
- Signalled Target Abort (bit 27)
- Master Data Parity Error (bit 24)

11.14.4.18.2.2 PCIe Baseline Error Handling

The PCIE_DEV_STAT_CTRL register permits software to enable generation of error messages for four error-related events and to check status information to determine which type of error has been detected:

- Correctable Errors
- Non-Fatal Errors
- Fatal Errors
- Unsupported Request Errors

11.14.4.18.2.2.1 Enabling Error Reporting

Setting the corresponding bit in the PCIE_DEV_STAT_CTRL register enables the generation of the corresponding error message, which reports errors associated with each classification. Unsupported Request errors are specified as Non-Fatal errors and are reported via a Non-Fatal error message, but only when the Unsupported Request Enable bit is set.

- Unsupported Request Reporting Enable (bit 3)
- Fatal Error Reporting Enable (bit 2)
- Non-Fatal Error Reporting Enable (bit 1)
- Correctable Error Reporting Enable (bit 0)

11.14.4.18.2.2.2 Error Status

An error status bit is set any time an error associated with its classification is detected. These bits are set irrespective of the setting of the Error Reporting Enable bits within the PCIE_DEV_STAT_CTRL register. Because Unsupported Request errors are by default considered Non-Fatal errors, when these errors occur both the Non-Fatal error status bit and the Unsupported Request status bit will be set.

NOTE: Both the Non-Fatal error status bit and the Unsupported Request status bit are cleared by software when writing a one (1) to the bit field.

- Unsupported Request Detected (bit 19)
- Fatal Error Detected (bit 18)
- Non-Fatal Error Detected (bit 17)
- Correctable Error Detected (bit 16)

11.14.4.18.2.2.3 Link Errors

The physical link connecting two devices may fail causing a variety of errors. Because the link has incurred errors, the error cannot be reported to the host via the failed link. Therefore, link errors must be reported via the upstream port of switches or by the Root Port itself. Also the related fields in the PCIE_LINK_STAT_CTRL are valid only in RC (not applicable within EP device). This permits system software to access link-related error registers on the port that is closest to the host.

The PCIE_LINK_STAT_CTRL register includes a RETRAIN_LINK bit (bit 5) that when set forces the RC to retrain the link. If transaction (upon completion of the retraining) can once again traverse the link without errors, the problem will have been solved.

Software can monitor the LINK_TRAINING bit (bit 27) in the PCIE_LINK_STAT_CTRL register to determine when retraining has completed.

11.14.4.18.2.2.4 Root's Response to Error Message

When a message is received by the RC the action that it takes when reporting the error message to the host system is determined in part by the PCIE_ROOT_CTRL_CAP register settings. There are three bit fields that specify whether an error should be reported as a fatal System Error (SERR set enables generation of a system error):

- SERR on Fatal Error Enable (bit 2)
- SERR on Non-Fatal Error Enable (bit 1)
- SERR on Correctable Error Enable (bit 0)

The PME Interrupt Enable bit (bit 3) allows software to enable and disable interrupt generation upon the RC detecting a PME Message transaction.

11.14.4.18.2.3 PCIe Advanced Error Reporting

Advanced Error Reporting requires implementation of the Advanced Error Reporting registers. These registers provide several additional error reporting features.

11.14.4.18.2.3.1 ECRC Generation and Checking

End-to-End CRC (ECRC) generation and checking can be enabled by PCIE_ACCR

- ECRC Check Enable (bit 8)
- ECRC Check Capable (bit 7)
- ECRC Generation Enable (bit 6)
- ECRC Generation Capable (bit 5)

In some cases, multiple uncorrectable errors may be detected prior to software reading and clearing the register. The First Error Pointer field (bit[4-0]) identifies the bit position within the PCIe Uncorrectable Error Status Register (PCIE_UNCERR) corresponding to the error that occurred first.

NOTE: When connecting KeyStone PCIe device to another external PCIe device, it is worth checking if the external PCIe device supports the ECRC feature. The ECRC should be disabled by clearing bits ECRC_CHK_EN and ECRC_GEN_EN in the PCIE_ACCR register if the external PCIe device does not support it, especially if the KeyStone PCIe device is configured as RC.

11.14.4.18.2.3.2 Advanced Correctable Error Handling

Advanced error reporting provides the ability to pinpoint specific correctable errors. These errors can selectively cause the generation of a Correctable Error Message being sent to the host system:

- Advanced Correctable Error Status
- When a correctable error occurs the corresponding bit within the PCIe Correctable Error Status Register (PCIE_CERR) is set. These bits are automatically set by hardware and are cleared by software when writing a 1 to the bit position. These bits are set whether or not the error is reported via an error message.
  - Replay Timer Timeout Status (RPLY_TMR_ST) (bit 12)
  - REPLAY_NUM Rollover Status (RPLT_RO_ST) (bit 8)
  - Bad DLLP Status (BAD_DLLP_ST) (bit 7)
  - Bad TLP Status (BAD_TLP_ST) (bit 6)
  - Receiver Error Status (RCVR_ERR_ST) (bit 0)
- Advanced Correctable Error Reporting
- Whether a particular correctable error is reported to the host is specified by the PCIe Correctable Error Mask Register (PCIE_CERR_MASK). The default state of the mask bits are cleared (0), thereby causing a Correctable Error message to be delivered when any of the correctable errors are detected. Software may choose to set one or more bits to prevent a Correctable Error Message from being sent

when the selected error is detected.

- Replay Timer Timeout Mask (RPLY_TMR_MSK) (bit 12)
- REPLAY_NUM Rollover Mask (RPLT_RO_MSK) (bit 8)
- Bad DLLP Mask (BAD_DLLP_MSK) (bit 7)
- Bad TLP Mask (BAD_TLP_MSK) (bit 6)
- Receiver Error Mask (RCVR_ERR_MSK) (bit 0)

11.14.4.18.2.3.3 Advanced Uncorrectable Error Handling

Advanced error reporting provides the ability to pinpoint which uncorrectable error has occurred. Furthermore software can specify the severity of each error and select which errors will result in an error message being sent to the host system (Root Complex).

- **Advanced Uncorrectable Error Status:** When an uncorrectable error occurs the corresponding bit within the PCIe Uncorrectable Error Status Register (PCIE_UNCERR) is set. These bits are automatically set by hardware and are cleared by software when writing a 1 to the bit position. Bits [4,12-20] are set whether or not the error is reported via an error message.
- **Advanced Uncorrectable Error Reporting:** Software can mask out specific errors so that they never cause an error message to be generated. The default condition in the PCIe Uncorrectable Error Mask Register (PCIE_UNCERR_MASK) is to generate error messages for each type of error (all bits are cleared). The same numbers of bits [4,12-20] , as in PCIE_UNCERR are used.
- **Selecting the Severity of Each Uncorrectable Error:** Advanced error handling permits software to select the severity of each error within the PCIe Uncorrectable Error Severity Register (PCIE_UNCERR_SVRTY). This gives software the opportunity to treat errors according to the severity associated with a given application. For example, Poisoned TLP data associated with audio data being sent to a speaker, while not correctable has no serious side effects relative to the integrity of the system. However, if real-time status information is being retrieved that will help make critical decisions, any errors in this data can be very serious. The values in the individual bit fields represent the default severity levels for each type of error (0 = Non-Fatal,1 = Fatal).

11.14.4.18.2.3.4 Error Logging

A four double-word portion (PCIE_HDR_LOG0÷PCIE_HDR_LOG3) of the PCIe Extended Capabilities Registers block is reserved for storing the header of the transaction that has incurred a failure. Only a select group of transaction layer errors result in the transaction header being logged. Table 11-2573 lists the transactions that are logged.

Table 11-2573. Transaction Layer Error That are Logged

Name of Error	Default Classification
Poisoned TLP Received	Uncorrectable - Non-Fatal
ECRC Check Failed	Uncorrectable – Non-Fatal
Unsupported Request	Uncorrectable – Non-Fatal
Completion Abort	Uncorrectable – Non-Fatal
Unexpected Completion	Uncorrectable – Non-Fatal
Malformed TLP	Uncorrectable – Fatal

11.14.4.18.2.3.5 Root Complex Error Tracking and Reporting

The Root Complex is the target of all error messages issued by devices within the PCIe fabric. Errors received by the RC result in status registers being updated and the error being conditionally reported to the appropriate software handler or handlers.

- **Root Error Status Registers:** When the Root Complex receives an error message, it sets status bits within the Root Error Status Register (PCIE_RC_ERR_ST). This register indicates the types of errors received and also indicates when multiple errors of the same type have been received.
  - Fatal Error Messages Received (bit 6)

  - Non-Fatal Error Messages Received (bit 5)
  - First Uncorrectable Fatal Received (bit 4)
  - Multiple Uncorrectable Error Received (bit 3)
  - Uncorrectable Error Received (bit 2)
  - Multiple Correctable Error Received (bit 1)
  - Correctable Error Received (bit 0)
- **Root Error Command Register:** The Root Error Status Register sets status bits that determines whether a Correctable, Fatal, or Non-Fatal error has occurred. In conjunction with these status bits the Root Complex can also generate separate interrupts that call handlers for each of the error categories. The Root Error Command register (PCIE_RC_ERR_CMD) enables interrupt generation for all three categories as follows:
  - Fatal Error Reporting Enable (bit 2)
  - Non-Fatal Error Reporting Enable (bit 1)
  - Correctable Error Reporting Enable (bit 0)
- **Error Source ID Register:** Software error handlers may need to read and clear error status registers within the device that detected and reported the error. The error messages contain the ID of the device reporting the error. The Error Source Identification Register (PCIE_ERR_SRC_ID) captures the error message ID associated with the first Fatal/Non-Fatal and correctable Error message received by the Root Complex.

11.14.4.18.2.4 Read Completion with an Error Completion Status

When a Read Completion is received with a Completion Status other than Successful Completion:

- No data is included with the Completion.
- Cpl (Completion without Data) encoding is used instead of CplD (Completion with Data). PCIe SS will return all the data as 0's to the host with an error status indicated on the bus.
- This Completion is the final Completion for the Request.
- The Requester must consider the Request terminated, and not expect additional Completions.

NOTE: Some system configuration software depends on reading a data value of all 1’s when a Configuration Read Request is terminated as an Unsupported Request, particularly when probing to determine the existence of a device in the system. A Root Complex intended for use with software that depends on a read-data value of all 1’s must synthesize this value when UR Completion Status is returned for a Configuration Read Request.

11.14.4.19 Memory Error Detection and Correction

11.14.4.19.1 ECC Wrapper and ECC Aggregator

The ECC wrapper implements SECDED code for single bit error detection and correction and double bit error detection. ECC Aggregator module gathers level pending status from the ECC RAMs into a single interrupt to the host. Software interface is provided for status and control.

There are three types of ECC registers:

1. *Global registers* that are common to all ECC RAMs. This includes the PCIE_ECC_REVISION, PCIE_ECC_VECTOR, and PCIE_ECC_MISC_STATUS registers. Every ECC RAM serviced by the aggregator module is assigned a unique ID by the module. Writing this ID to the PCIE_ECC_VECTOR register selects the ECC RAM to be controlled or read the ECC status from.
2. *ECC control and status registers.* These are specific to each ECC RAM and selected by the RAM_ID written to the PCIE_ECC_VECTOR register. The accesses to these registers are done via the serial VBUSP protocol. Note that due to the long latencies on the serial VBUS, the reads to the ECC control and status registers are triggered by a write to the PCIE_ECC_VECTOR register. This is described in detail in Section 11.14.4.19.1.1.
3. *Interrupt registers.* These include the standard interrupt status, interrupt enable, interrupt clear and EOI registers that are part of a standard interrupt module.

   ECC registers are described in Section 11.14.5.2, *PCIE_ECC Registers*.

11.14.4.19.1.1 Reads to ECC Control and Status Registers

Due to the long latencies on the serial VBUS, the reads to the ECC control and status registers for each ECC RAM are triggered by writing a *read* message to the PCIE_ECC_VECTOR register as described below:

1. Software writes the RAM_ID in PCIE_ECC_VECTOR[10-0] , sets bit [15] TRIGGER_READ, and writes [23-16] READ_ADDRESS. The READ_ADDRESS corresponds to any of the ECC control or status registers (at offset 0x10, that is, PCIE_ECC_WRAPPER_REVISION through 0x24, that is, PCIE_ECC_ERROR_STATUS2) which require serial VBUS access to the ECC RAM(s).
2. Software then polls the PCIE_ECC_VECTOR[24] READ_DONE bit waiting for setting to 1. This indicates that the read operation has completed on the serial VBUS.
3. Software reads the data from the ECC control or status register. Read data is returned on the following clock cycle.

11.14.4.19.1.2 ECC Interrupts

The ECC aggregator module aggregates all the level pending status from the ECC RAMs to a single EOI-handshake based interrupt to the host. Software is expected to follow the sequence described below:

1. Software enables the interrupts for the ECC RAM(s) by writing to the Interrupt Enable register (PCIE_ECC_INT_ENABLE_0 to PCIE_ECC_INT_ENABLE_15).
2. On receiving an interrupt, software checks the Interrupt Status Register (PCIE_ECC_INT_STATUS_0 to PCIE_ECC_INT_STATUS_15) to see which ECC RAM(s) caused the error. Note that this status read is not a serial VBUS operation.
3. Software writes the RAM_ID in the PCIE_ECC_VECTOR register along with the read message that includes setting bit [15] TRIGGER_READ, writing the PCIE_ECC_ERROR_STATUS1 register's address (that is, 0x20) to bits [23-16] READ_ADDRESS. Software needs to load the read message in the PCIE_ECC_VECTOR register again if it needs to read also the PCIE_ECC_ERROR_STATUS2.
4. Software polls the PCIE_ECC_VECTOR[24] READ_DONE bit. When this is set to 1, it can read the PCIE_ECC_ERROR_STATUS1 and PCIE_ECC_ERROR_STATUS2 registers.
5. After the interrupt has been serviced, software must clear the interrupt status by writing to bits 8, 9, or 10 depending on the type of ECC error in the PCIE_ECC_ERROR_STATUS1 register.
6. Software must poll the PCIE_ECC_ERROR_STATUS1 register to guarantee that the status bit has been cleared. Otherwise there is no indication that the write has actually completed over the serial VBUS

7. Software must write to the PCIE_ECC_EOI register to clear the aggregated interrupt output.

11.14.4.20 Emulation Considerations

The emulation suspend interface is not supported on PCIe SS.

11.14.4.21 Encoding of LTSSM State in DEBUG Registers

This appendix contains the following supplementary information:

11.14.4.21.1 LTSSM States

The LTSSM state value that is read from DEBUG registers is an encoded value. The literal names of the LTSSM states corresponding to the encoded values are tabulated below.

Table 11-2574. Encoding of LTSSM State in DEBUG registers

Code	LTSSM State
0x00	DETECT_QUIET
0x01	DETECT_ACT
0x02	POLL_ACTIVE
0x03	POLL_COMPLIANCE
0x04	POLL_CONFIG
0x05	PRE_DETECT_QUIET
0x06	DETECT_WAIT
0x07	CFG_LINKWD_START
0x08	CFG_LINKWD_ACEPT
0x09	CFG_LANENUM_WAIT
0x0A	CFG_LANENUM_ACEPT
0x0B	CFG_COMPLETE
0x0C	CFG_IDLE
0x0D	RCVRY_LOCK
0x0E	RCVRY_SPEED
0x0F	RCVRY_RCVRCFG
0x10	RCVRY_IDLE
0x11	L0
0x12	L0s
0x13	L123_SEND_EIDLE
0x14	L1_IDLE
0x15	L2_IDLE
0x16	L2_WAKE
0x17	DISABLED_ENTRY
0x18	DISABLED_IDLE
0x19	DISABLED
0x1A	LPBK_ENTRY
0x1B	LPBK_ACTIVE
0x1C	LPBK_EXIT
0x1D	LPBK_EXIT_TIMEOUT
0x1E	HOT_RESET_ENTRY
0x1F	HOT_RESET

11.14.5 PCIe SS Registers

This section describes the PCIe registers. They are divided into 10 subsection based on their specific use.

11.14.5.1 SERDES Configuration Registers

11.14.5.1.1 Register Summary

Table 11-2576 shows the configuration registers of the SERDES.

Table 11-2575. PCIe Instances

Instance	Base Address
PCIE_0_PHY_CFG	0232 1FC0h

Table 11-2576. SERDES Configuration Registers

Offset	Acronym	Register Name	PCIE_0_PHY_CFG Physical Address	Section
0h	PCIE_PHY_MOD_VER	Identifies revision of SERDES	0232 1FC0h	Section 11.14.5.1.2
14h	PCIE_PHY_LANE_PLL_STS	Current configuration of REFCLK and LANE Mux selection	0232 1FD4h	Section 11.14.5.1.3
20h	PCIE_PHY_LANExCTL_STS	Override SERDES configuration settings.	0232 1FE0h	Section 11.14.5.1.4
34h	PCIE_PHY_PLL_CTRL	PLL Control register	0232 1FF4h	Section 11.14.5.1.5
38h	PCIE_PHY_COMMA_LINK_DELAY	Comma Link delay register	0232 1FF8h	Section 11.14.5.1.6
3Ch	PCIE_PHY_CMU_WAIT	CMU Wait Register	0232 1FFCh	Section 11.14.5.1.7

11.14.5.1.2 PCIE_PHY_MOD_VER Register (Offset = 0h) [reset = 4EBE2101h]

The Module and Version Register PCIE_PHY_MOD_VER identifies the module identifier and revision of the WIZ8B2M4SB module.

Table 11-2577. PCIE_PHY_MOD_VER Register

Instance	Physical Address
PCIE	0232 1FC0h

Figure 11-1103. PCIE_PHY_MOD_VER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 4EBE2101h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2578. PCIE_PHY_MOD_VER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4EBE2101h	TI internal data. Identifies revision of peripheral.

Table 11-2579. Register Call Summary for PCIE_PHY_MOD_VER

SERDES Configuration Registers
- Register Summary: [0]
- PCIE_PHY_MOD_VER Register (Offset = 0h) [reset = 4EBE2101h]: [0]

11.14.5.1.3 PCIE_PHY_LANE_PLL_STS Register (Offset = 14h) [reset = 800h]

The Lane and PLL Status register is used to indicate the current configuration of the REFCLK distribution and Lane Mux selection.

Table 11-2580. PCIE_PHY_LANE_PLL_STS Register

Instance	Physical Address
PCIE	0232 1FD4h

Figure 11-1104. PCIE_PHY_LANE_PLL_STS Register

31	30	29	28	27	26	25	24
RESERVED				PLL0_PWR_ON	PLL0_LEFT_SEL	PLL0_CHAIN_EN	PLL0_RIGHT_SEL
R-0h				R-1h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				LANE0_PLL_LOCK	LANE0_RST_ISO	LANE0_IP_SEL	
R-0h				R-0h	R-0h	R-0h	

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2581. PCIE_PHY_LANE_PLL_STS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	Reserved	R	0h	Reserved - writes are ignored, always reads zeros.
27	PLL0_PWR_ON	R	1h	Indicates the state of the PLL0 REFCLK supplies: • 0 - REFCLK power is off • 1 - REFCLK power is on
26	PLL0_LEFT_SEL	R	1h	Indicates the REFCLK source used for PLL0. This bit indicates that the left refclk input is used for the PLL0. When both PLL0_LEFT_SEL and PLL0_RIGHT_SEL are zero, the REFCLK pads are used for the PLL0. • 0 - disabled left clock • 1 - enable left clock
25	PLL0_CHAIN_EN	R	0h	Indicates the refclk chaining is enabled: • 0 - REFCLK right and left are driven from the REFCLK pads of this macro • 1 - REFCLK right or left is driven from left or right respectively.
24	PLL0_RIGHT_SEL	R	0h	Indicates the refclk source used for PLL0. This bit indicates that the right REFCLK Input is used for the PLL0. When bothPLL0_LEFT_SEL and PLL0_RIGHT_SEL are zero, the REFCLK pads are used for the PLL0. • 0 - disabled left clock • 1 - enable left clock
23-4	Reserved	R	0h	Reserved - writes are ignored, always reads zeros.
3	LANE0_PLL_LOCK	R	0h	Indicates that the lane 0 TX clock is valid
2	LANE0_RST_ISO	R	0h	Indicates that the selected IP has the Lane 0 reset isolation signal set.

Table 11-2581. PCIE_PHY_LANE_PLL_STS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	LANE0_IP_SEL	R	0h	Indicates which IP interface is used for lane 0. • 00 - IP1 is selected for lane 0. • 01 - IP2 is selected for lane 0. • 10 - IP3 is selected for lane 0. • 11 - IP4 is selected for lane 0.

Table 11-2582. Register Call Summary for PCIE_PHY_LANE_PLL_STS

SERDES Configuration Registers • Register Summary: [0]

11.14.5.1.4 PCIE_PHY_LANExCTL_STS Register (Offset = 14h) [reset = 0h]

The lane x control and status provides the ability to override SERDES configuration settings. It is mainly provided for test, but sometime used for mission setup as well.

Table 11-2583. PCIE_PHY_LANExCTL_STS Instances

Instance	Physical Address
PCIE	0232 1FE0h

Figure 11-1105. PCIE_PHY_LANExCTL_STS Register

31	30	29	28	27	26	25	24
TX0_ENABLE_OVL	TX0_ENABLE_VAL		TX0_RATE_OVL	TX0_RATE_VAL		TX0_IDLE_OVL	TX0_IDLE_VAL
R/W-0h	R/W-0h		R/W-0h	R/W-0h		R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
TX0_WIDTH_OVL	TX0_WIDTH_VAL		RESERVED				
R/W-0h	R/W-0h		R-0h				
15	14	13	12	11	10	9	8
RX0_ENABLE_OVL	RX0_ENABLE_VAL		RX0_RATE_OVL	RX0_RATE_VAL		RX0_POLARITY_OVL	RX0_POLARITY_VAL
R/W-0h	R/W-0h		R/W-0h	R/W-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RX0_ALIGN_OVL	RX0_ALIGN_VAL	RX0_WIDTH_OVL	RX0_WIDTH_VAL		RESERVED	RX0_OK	RX0_LOSS
R/W-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2584. PCIE_PHY_LANExCTL_STS Register Field Descriptions

Bit	Field	Type	Reset	Description
31	TX0_ENABLE_OVL	R/W	0h	The Tx Enable Overlay bit when set allows the Tx Enable Value to override the CFGTX Enable input.
30-29	TX0_ENABLE_VAL	R/W	0h	The Tx Enable Value when used allows the Tx lane to be placed in • 00 - Disable state • 01 - Sleep state • 10 - Snooze state • 11 - Enabled state
28	TX0_RATE_OVL	R/W	0h	The Tx Rate Overlay bit when set allows the Tx Rate Value to override the CFGTX Rate input.
27-26	TX0_RATE_VAL	R/W	0h	The Tx Rate Value when used allows the Tx lane to be placed into: • 00 - Full Rate mode • 01 - Half Rate mode • 10 - Quarter Rate mode • 11 - Reserved
25	TX0_IDLE_OVL	R/W	0h	The Tx Idle Overlay bit when set allows the Tx Idle Value to override the CFGTX.Idle input.
24	TX0_IDLE_VAL	R/W	0h	The Tx Idle Value when used allows the Tx lane to be placed electrical Idle state in non PCIE mode.
23	TX0_WIDTH_OVL	R/W	0h	The Tx Width Overlay bit when set allows the Tx Width Value to override the CFGTX.Width input.

Table 11-2584. PCIE_PHY_LANExCTL_STS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22-21	TX0_WIDTH_VAL	R/W	0h	The Tx Width Value when used allows the Tx lane to be placed into: • 00 - 10 bit mode • 01 - Reserved • 10 - 20 bit mode • 11 - 16 bit mode
20-16	Reserved	R	0h	Reserved - writes are ignored, always reads zeros.
15	RX0_ENABLE_OVL	R/W	0h	The Rx Enable Overlay bit when set allows the Rx Enable Value to override the CFGRX Enable input.
14-13	RX0_ENABLE_VAL	R/W	0h	The Rx Enable Value when used allows the Rx lane to be placed in: • 00 - Disable state • 01 - Sleep state • 10 - Snooze state • 11 - Enabled state
12	RX0_RATE_OVL	R/W	0h	The Rx Rate Overlay bit when set allows the Rx Rate Value to override the CFGRX Rate input.
11-10	RX0_RATE_VAL	R/W	0h	The Rx Rate Value when used allows the Rx lane to be placed into: • 00 - Full Rate mode • 01 - Half Rate mode • 10 - Quarter Rate mode • 11 - Reserved
9	RX0_POLARITY_OVL	R/W	0h	The Rx Polarity Overlay bit when set allows the Rx Polarity Value to override the CFGRX.Polarity input.
8	RX0_POLARITY_VAL	R/W	0h	The Rx Polarity Value when used allows the lane Rx Polarity to be inverted.
7	RX0_ALIGN_OVL	R/W	0h	The Rx Align Overlay bit when set allows the Rx Align Value to override the CFGRX.Align input.
6	RX0_ALIGN_VAL	R/W	0h	The Rx Align Value when used allows the Rx lane to align to K28.1, K28.5 and K28.7 characters otherwise known as Comma Characters.
5	RX0_WIDTH_OVL	R/W	0h	The Rx Width Overlay bit when set allows the Rx Width Value to override the CFGRX Width input.
4-3	RX0_WIDTH_VAL	R/W	0h	The Rx Width Value when used allows the Rx lane to be placed into • 00 - 10 bit mode • 01 - Reserved • 10 - 20 bit mode • 11 - 16 bit mode
2	Reserved	R	0h	Reserved - writes are ignored, always reads zeros.
1	RX0_OK	R	0h	The Rx OK indicate that the lane is in a functional state.
0	RX0_LOSS	R		The Rx Signal Loss Indicates that the data has not been detected or the CDR is not locked. • 0 - data detected or CDR locked • 1 - data not detected or CDR not locked

Table 11-2585. Register Call Summary for PCIE_PHY_LANExCTL_STS

SERDES Configuration Registers
• Register Summary: [0]

11.14.5.1.5 PCIE_PHY_PLL_CTRL Register (Offset = 34h) [reset = 0h]

The PLL Control register provides the ability to override the PLL state and control events for debug purposes.

Table 11-2586. PCIE_PHY_PLL_CTRL Instances

Instance	Physical Address
PCIE	0232 1FF4h

Figure 11-1106. PCIE_PHY_PLL_CTRL Register

31	30	29	28	27	26	25	24
PLL_ENABLE_OVL	PLL_ENABLE_VAL		PLL_OK	RESERVED			
R/W-0h	R/W-0h		R/W-0h	R-0h			
23	22	21	20	19	18	17	16
RESERVED						LN_WAFTER_OK	LN_WAFTER_SD
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED			LN0_CONT_OK	RESERVED			LN0_OK_STATE
R-0h			R/W-0h	R-0h			R/W-0h
7	6	5	4	3	2	1	0
RESERVED			LN0_CONT_SD	RESERVED			LN0_SD_STATE
R-0h			R/W-0h	R-0h			R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2587. PCIE_PHY_PLL_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	PLL_ENABLE_OVL	R/W	0h	The PLL Enable Overlay bit when set allows the PLL Enable Value to override the CFGPLL.Enable input.
30-29	PLL_ENABLE_VAL	R/W	0h	The PLL Enable Value when used allows the PLL to be placed in • 00 - Disable state • 01 - Sleep state • 10 - Snooze state • 11 - Enabled state
28	PLL_OK	R	0h	The PLL Ok indicate that the transmit clocks are valid and the PLL circuits are ready for use.
27-18	Reserved	R	0h	Reserved - writes are ignored, always reads zeros.
17	LN_WAFTER_OK	R/W	0h	The LN_WAFTER_OK field will cause the lane OK indication to be Blocked until the LNX_CONT_OK is set. This allows code to run after the lane OK but before the hardware can use the lane.
16	LN_WAFTER_SD	R/W	0h	The LN_WAFTER_SD field will cause the Rx signal indication to be Blocked until the LNX_CONT_SD is set. This allows code to run after signal detect but before the hardware can use the data.
15-13	Reserved	R	0h	Reserved - writes are ignored, always reads zeros.
12	LN0_CONT_OK	R/W	0h	The LN0_CONT_OK allows the Blocked lane OK for lane 0 to now pass to the controlling PCS IP. This bit is auto cleared when LN0_OK_STATE is driven inactive.
11-9	Reserved	R	0h	Reserved - writes are ignored, always reads zeros.
8	LN0_OK_STATE	R	0h	The LN0_OK_STATE indicate the current state of the lane OK signal for lane 0.
7-5	Reserved	R	0h	Reserved - writes are ignored, always reads zeros.

Table 11-2587. PCIE_PHY_PLL_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	LN0_CONT_SD	R/W	0h	The LN0_CONT_SD allows the Blocked signal detect for lane 0 to now pass to the controlling PCS IP. This bit is auto cleared when LN0_SD_STATE is driven inactive.
3-1	Reserved	R	0h	Reserved - writes are ignored, always reads zeros.
0	LN0_SD_STATE	R	0h	The LN0_SD_STATE indicate the current state of the signal detect signal for lane 0.

Table 11-2588. Register Call Summary for PCIE_PHY_PLL_CTRL

SERDES Configuration Registers • Register Summary: [0]
PCIe SS Functional Description • Enabling the PLL: [0][1] • Initialization Sequence: [0][1] • Initialization Sequence: [0][1]

11.14.5.1.6 PCIE_PHY_COMMA_LINK_DELAY Register (Offset = 38h) [reset = 0h]

The Comma Link Delay register defines the added delay due to comma alignment for the SERDES lanes. This value should be added to the receive time stamp to accurately time stamp the packet arrival.

Table 11-2589. PCIE_PHY_COMMA_LINK_DELAY Instances

Instance	Physical Address
PCIE	0232 1FF8h

Figure 11-1107. PCIE_PHY_COMMA_LINK_DELAY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								LANE0_CDELAY							
R-0h																								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2590. PCIE_PHY_COMMA_LINK_DELAY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	Reserved	R	0h	Reserved - writes are ignored, always reads zeros.
7-0	LANE0_CDELAY	R	0h	Defines the number of network bits the comma aligner has added to the fixed delay of the PCS/PMA layer for lane 0.

Table 11-2591. Register Call Summary for PCIE_PHY_COMMA_LINK_DELAY

SERDES Configuration Registers
• Register Summary: [0]

11.14.5.1.7 PCIE_PHY_CMU_WAIT Register (Offset = 3Ch) [reset = 0h]

The CMU Wait is used to create a fixed second delay between the completion of the SERDES configuration writes and activating the SERDES CMU logic allowing for supplies to settle.

Table 11-2592. PCIE_PHY_CMU_WAIT Instances

Instance	Physical Address
PCIE	0232 1FFCh

Figure 11-1108. PCIE_PHY_CMU_WAIT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															WAIT_VAL																
R-0h															R-0h																

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2593. PCIE_PHY_CMU_WAIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	Reserved	R	0h	Reserved - writes are ignored, always reads zeros.
16-0	WAIT_VAL	R	0h	Defines the number of clock cycles between writing to the CMU Reset register and CMU_RESET_I activation.

Table 11-2594. Register Call Summary for PCIE_PHY_CMU_WAIT

SERDES Configuration Registers
• Register Summary: [0]

11.14.5.2 PCIE_ECC Registers

Table 11-2596 lists the memory-mapped registers for the PCIE_ECC. All register offset addresses not listed in Table 11-2596 should be considered as reserved locations and the register contents should not be modified.

Table 11-2595. PCIE_ECC Instances

Instance	Base Address
PCIE_0_ECC_CFG	0233 0400h

Table 11-2596. PCIE_ECC Registers

Offset	Acronym	Register Name	PCIE_0_ECC_CFG Physical Address	Section
0h	PCIE_ECC_REVISION	The Revision Register contains the ID and revision information.	0233 0400h	Section 11.14.5.2.1
8h	PCIE_ECC_VECTOR	ECC RAM ID to select the ECC RAM to control or read status.	0233 0408h	Section 11.14.5.2.2
Ch	PCIE_ECC_MISC_STATUS	Miscellaneous status register.	0233 040Ch	Section 11.14.5.2.3
10h	PCIE_ECC_WRAPPER_REVISION	The Revision Register contains the ID and revision information for the ECC wrapper.	0233 0410h	Section 11.14.5.2.4
14h	PCIE_ECC_CONTROL	The Control Register controls the ECC control bits for the selected ECC RAM.	0233 0414h	Section 11.14.5.2.5
18h	PCIE_ECC_ERROR_CONTROL1	This register contains ECC error control bits for the selected ECC RAM.	0233 0418h	Section 11.14.5.2.6
1Ch	PCIE_ECC_ERROR_CONTROL2	This register contains ECC error control bits for the selected ECC RAM.	0233 041Ch	Section 11.14.5.2.7
20h	PCIE_ECC_ERROR_STATUS1	This register contains ECC status bits for the selected ECC RAM.	0233 0420h	Section 11.14.5.2.8
24h	PCIE_ECC_ERROR_STATUS2	This register contains ECC status bits for the selected ECC RAM.	0233 0424h	Section 11.14.5.2.9
3Ch	PCIE_ECC_EOI	This is the PCIE_ECC_EOI register for the interrupt to the host.	0233 043Ch	Section 11.14.5.2.10
40h to 7Ch	PCIE_ECC_INT_STATUS_0 to PCIE_ECC_INT_STATUS_15	These are the raw level interrupt status bits where each bit corresponds to the pending status from an ECC RAM.	0233 0440h to 0233 047Ch	Section 11.14.5.2.11
80h to BCh	PCIE_ECC_INT_ENABLE_0 to PCIE_ECC_INT_ENABLE_15	These are interrupt enables associated with the interrupt from each of the ECC RAMs. Writing a 1 to a bit position in the register enables the interrupt from the associated ECC RAM.	0233 0480h to 0233 04BCh	Section 11.14.5.2.12
C0h to FCh	PCIE_ECC_INT_CLEAR_0 to PCIE_ECC_INT_CLEAR_15	These are interrupt enable clear bits associated with the interrupt from each of the ECC RAMs. Writing a 1 to a bit position in the register disables the interrupt from the associated ECC RAM.	0233 04C0h to 0233 04FCh	Section 11.14.5.2.13

11.14.5.2.1 PCIE_ECC_REVISION Register (Offset = 0h) [reset = 4E100001h]

PCIE_ECC_REVISION is shown in and described in Table 11-2598.

The Revision Register contains the ID and revision information.

Table 11-2597. PCIE_ECC_REVISION Instances

Instance	Physical Address
PCIE_0_ECC_CFG	0233 0400h

Figure 11-1109. PCIE_ECC_REVISION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 4E100001h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2598. PCIE_ECC_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4E100001	TI internal data. Identifies revision of peripheral.

Table 11-2599. Register Call Summary for PCIE_ECC_REVISION

PCIE_ECC Registers • PCIE_ECC_REVISION Register (Offset = 0h) [reset = 4E100001h]: [0]
PCIe SS Registers • PCIE_ECC Registers: [0]
PCIe SS Functional Description • ECC Wrapper and ECC Aggregator: [0]

11.14.5.2.2 PCIE_ECC_VECTOR Register (Offset = 8h) [reset = 0h]

PCIE_ECC_VECTOR is shown in Figure 11-1110 and described in Table 11-2601.

ECC RAM ID to select the ECC RAM to control or read status.

Table 11-2600. PCIE_ECC_VECTOR Instances

Instance	Physical Address
PCIE_0_ECC_CFG	0233 0408h

Figure 11-1110. PCIE_ECC_VECTOR Register

31	30	29	28	27	26	25	24
RESERVED							READ_DONE
R-							R-0
23	22	21	20	19	18	17	16
READ_ADDRESS							
R/W-0							
15	14	13	12	11	10	9	8
TRIGGER_READ	RESERVED					RAM_ID	
R/W-0	R-					R/W-0	
7	6	5	4	3	2	1	0
RAM_ID							
R/W-0							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2601. PCIE_ECC_VECTOR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R		Reserved
24	READ_DONE	R	0	Status indicating that the serial VBUS read is complete
23-16	READ_ADDRESS	R/W	0	Read address. Can be any of the registers 0x10 - 0x24
15	TRIGGER_READ	R/W	0	Trigger a read operation to the specified read address that requires a serial VBUS access
14-11	RESERVED	R		Reserved
10-0	RAM_ID	R/W	0	ECC RAM ID to select which ECC RAM to control or read status from

Table 11-2602. Register Call Summary for PCIE_ECC_VECTOR

PCIE_ECC Registers • PCIE_ECC_VECTOR Register (Offset = 8h) [reset = 0h]: [0]
PCIe SS Registers • PCIE_ECC Registers: [0]
PCIe SS Functional Description • ECC Interrupts: [0][1][2] • Reads to ECC Control and Status Registers: [0][1][2] • ECC Wrapper and ECC Aggregator: [0][1][2][3]

11.14.5.2.3 PCIE_ECC_MISC_STATUS Register (Offset = Ch) [reset = 0h]

PCIE_ECC_MISC_STATUS is shown in Figure 11-1111 and described in Table 11-2604.

Miscellaneous status register.

Table 11-2603. PCIE_ECC_MISC_STATUS Instances

Instance	Physical Address
PCIE_0_ECC_CFG	0233 040Ch

Figure 11-1111. PCIE_ECC_MISC_STATUS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					NUM_RAMS										
R-																					R/W-										

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2604. PCIE_ECC_MISC_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0	Reserved
10-0	NUM_RAMS	R/W		Number of ECC RAMs serviced by the aggregator

Table 11-2605. Register Call Summary for PCIE_ECC_MISC_STATUS

PCIE_ECC Registers • PCIE_ECC_MISC_STATUS Register (Offset = Ch) [reset = 0h]: [0]
PCIe SS Registers • PCIE_ECC Registers: [0]
PCIe SS Functional Description • ECC Wrapper and ECC Aggregator: [0]

11.14.5.2.4 PCIE_ECC_WRAPPER_REVISION Register (Offset = 10h) [reset = 4E100001 h]

PCIE_ECC_WRAPPER_REVISION is shown in Figure 6-57 and described in Table 11-2607.

The Revision Register contains the ID and revision information for the ECC wrapper.

Table 11-2606. PCIE_ECC_WRAPPER_REVISION Instances

Instance	Physical Address
PCIE_0_ECC_CFG	0233 0410h

Figure 11-1112. PCIE_ECC_WRAPPER_REVISION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 4E100001h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2607. PCIE_ECC_WRAPPER_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4E100001	TI internal data. Identifies revision of peripheral.

Table 11-2608. Register Call Summary for PCIE_ECC_WRAPPER_REVISION

PCIE_ECC Registers • PCIE_ECC_WRAPPER_REVISION Register (Offset = 10h) [reset = 4E100001 h]: [0]
PCIe SS Registers • PCIE_ECC Registers: [0]
PCIe SS Functional Description • Reads to ECC Control and Status Registers: [0]

11.14.5.2.5 PCIE_ECC_CONTROL Register (Offset = 14h) [reset = 7h]

PCIE_ECC_CONTROL is shown in Figure 11-1113 and described in Table 11-2610.

The Control Register controls the ECC control bits for the selected ECC RAM.

Table 11-2609. PCIE_ECC_CONTROL Instances

Instance	Physical Address
PCIE_0_ECC_CFG	0233 0414h

Figure 11-1113. PCIE_ECC_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED	ERROR_ONCE	FORCE_N_ROW	FORCE_DED	FORCE_SEC	ENABLE_RMW	PCIE_ECC_CHECK	PCIE_ECC_ENABLE
R-	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2610. PCIE_ECC_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R		Reserved
6	ERROR_ONCE	R/W	0	If this bit is set, the force_sec/force_ded will inject an error to the specified row only once. The force_sec bit will be cleared once a writeback happens. If writeback is not enabled, this error will be cleared the cycle following the read when the data is corrected. For double-bit errors, the force_ded bit will be cleared the cycle following the double-bit error. Any subsequent reads will not force an error.
5	FORCE_N_ROW	R/W	0	Force single/double-bit error on the next RAM read
4	FORCE_DED	R/W	0	Force double-bit error. Cleared the cycle following the error if error_once is asserted.
3	FORCE_SEC	R/W	0	Force single-bit error. Cleared on a writeback or the cycle following the error if error_once is asserted
2	ENABLE_RMW	R/W	1	Enable read-modify-write on partial word writes
1	PCIE_ECC_CHECK	R/W	1	Enable ECC check. ECC is completely bypassed if both PCIE_ECC_enable and PCIE_ECC_check are 0
0	PCIE_ECC_ENABLE	R/W	1	Enable ECC generation

Table 11-2611. Register Call Summary for PCIE_ECC_CONTROL

PCIE_ECC Registers • PCIE_ECC_CONTROL Register (Offset = 14h) [reset = 7h]: [0]
PCIe SS Registers • PCIE_ECC Registers: [0]

11.14.5.2.6 PCIE_ECC_ERROR_CONTROL1 Register (Offset = 18h) [reset = 0h]

PCIE_ECC_ERROR_CONTROL1 is shown in Figure 11-1114 and described in Table 11-2613.

This register contains ECC error control bits for the selected ECC RAM.

Table 11-2612. PCIE_ECC_ERROR_CONTROL1 Instances

Instance	Physical Address
PCIE_0_ECC_CFG	0233 0418h

Figure 11-1114. PCIE_ECC_ERROR_CONTROL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCIE_ECC_BIT1																PCIE_ECC_ROW															
R/W-0																R/W-0															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2613. PCIE_ECC_ERROR_CONTROL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	PCIE_ECC_BIT1	R/W	0	Data bit that needs to be flipped when force_sec or force_ded is set
15-0	PCIE_ECC_ROW	R/W	0	Row address where force_sec or force_ded needs to be applied. This is ignored if force_n_row is set.

Table 11-2614. Register Call Summary for PCIE_ECC_ERROR_CONTROL1

PCIE_ECC Registers • PCIE_ECC_ERROR_CONTROL1 Register (Offset = 18h) [reset = 0h]: [0]
PCIe SS Registers • PCIE_ECC Registers: [0]

11.14.5.2.7 PCIE_ECC_ERROR_CONTROL2 Register (Offset = 1Ch) [reset = 0h]

PCIE_ECC_ERROR_CONTROL2 is shown in Figure 11-1115 and described in Table 11-2616.

This register contains ECC error control bits for the selected ECC RAM.

Table 11-2615. PCIE_ECC_ERROR_CONTROL2 Instances

Instance	Physical Address
PCIE_0_ECC_CFG	0233 041Ch

Figure 11-1115. PCIE_ECC_ERROR_CONTROL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PCIE_ECC_BIT2															
R-																R/W-0															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2616. PCIE_ECC_ERROR_CONTROL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R		Reserved
15-0	PCIE_ECC_BIT2	R/W	0	Data bit that needs to be flipped when force_ded is set

Table 11-2617. Register Call Summary for PCIE_ECC_ERROR_CONTROL2

PCIE_ECC Registers • PCIE_ECC_ERROR_CONTROL2 Register (Offset = 1Ch) [reset = 0h]: [0]
PCIe SS Registers • PCIE_ECC Registers: [0]

11.14.5.2.8 PCIE_ECC_ERROR_STATUS1 Register (Offset = 20h) [reset = 0h]

PCIE_ECC_ERROR_STATUS1 is shown in Figure 11-1116 and described in Table 11-2619.

This register contains ECC status bits for the selected ECC RAM.

Table 11-2618. PCIE_ECC_ERROR_STATUS1 Instances

Instance	Physical Address
PCIE_0_ECC_CFG	0233 0420h

Figure 11-1116. PCIE_ECC_ERROR_STATUS1 Register

31	30	29	28	27	26	25	24
PCIE_ECC_ROW							
R-0							
23	22	21	20	19	18	17	16
PCIE_ECC_ROW							
R-							
15	14	13	12	11	10	9	8
RESERVED					CLR_PCIE_ECC_OTHER	CLR_PCIE_ECC_DED	CLR_PCIE_ECC_SEC
R-					W1toCl-0	W1toCl-0	W1toCl-0
7	6	5	4	3	2	1	0
RESERVED					PCIE_ECC_OTHER	PCIE_ECC_DED	PCIE_ECC_SEC
R-					R/W-0	R/W-0	R/W-0

LEGEND: R = Read Only; R/W = Read/Write; W1toCl = Write 1 to Clear Bit; *-n* = value after reset

Table 11-2619. PCIE_ECC_ERROR_STATUS1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	PCIE_ECC_ROW	R	0	Indicates the row/address where the single or double-bit error occurred
15-11	RESERVED	R		Reserved
10	CLR_PCIE_ECC_OTHER	W1toCl	0	1 indicates a pending double-bit error. Writing a 1 clears the status bit.
9	CLR_PCIE_ECC_DED	W1toCl	0	1 indicates a pending successive single-bit error. Writing a 1 clears the status bit.
8	CLR_PCIE_ECC_SEC	W1toCl	0	1 indicates a pending single-bit error. Writing a 1 clears the status bit.
7-3	RESERVED	R		Reserved
2	PCIE_ECC_OTHER	R/W	0	1 indicates that successive single-bit errors have occurred while a writeback is still pending. Software can also write a 1 to set the pending status.
1	PCIE_ECC_DED	R/W	0	1 indicates a double-bit error. Software can also write a 1 to set the pending status.
0	PCIE_ECC_SEC	R/W	0	1 indicates a single-bit error. Software can also write a 1 to set the pending status.

Table 11-2620. Register Call Summary for PCIE_ECC_ERROR_STATUS1

PCIE_ECC Registers • PCIE_ECC_ERROR_STATUS1 Register (Offset = 20h) [reset = 0h]: [0]
PCIe SS Registers • PCIE_ECC Registers: [0]

Table 11-2620. Register Call Summary for PCIE_ECC_ERROR_STATUS1 (continued)

PCIe SS Functional Description
• ECC Interrupts: [0][1][2][3]

11.14.5.2.9 PCIE_ECC_ERROR_STATUS2 Register (Offset = 24h) [reset = 0h]

PCIE_ECC_ERROR_STATUS2 is shown in Figure 11-1117 and described in Table 11-2622.

This register contains ECC status bits for the selected ECC RAM.

Table 11-2621. PCIE_ECC_ERROR_STATUS2 Instances

Instance	Physical Address
PCIE_0_ECC_CFG	0233 0424h

Figure 11-1117. PCIE_ECC_ERROR_STATUS2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PCIE_ECC_BIT1															
R-0																R-0															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2622. PCIE_ECC_ERROR_STATUS2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R		Reserved
15-0	PCIE_ECC_BIT1	R	0	Indicates the data bit that is in error

Table 11-2623. Register Call Summary for PCIE_ECC_ERROR_STATUS2

PCIE_ECC Registers • PCIE_ECC_ERROR_STATUS2 Register (Offset = 24h) [reset = 0h]: [0]
PCIe SS Registers • PCIE_ECC Registers: [0]
PCIe SS Functional Description • ECC Interrupts: [0][1] • Reads to ECC Control and Status Registers: [0]

11.14.5.2.10 PCIE_ECC_EOI Register (Offset = 3Ch) [reset = 0h]

PCIE_ECC_EOI is shown in Figure 11-1118 and described in Table 11-2625.

This is the PCIE_ECC_EOI register for the interrupt to the host.

Table 11-2624. PCIE_ECC_EOI Instances

Instance	Physical Address
PCIE_0_ECC_CFG	0233 043Ch

Figure 11-1118. PCIE_ECC_EOI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0							
23	22	21	20	19	18	17	16
RESERVED							
R-0							
15	14	13	12	11	10	9	8
RESERVED							
R-0							
7	6	5	4	3	2	1	0
RESERVED							EOI_WR
R-0							W1toCl-0

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2625. PCIE_ECC_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R		Reserved
0	EOI_WR	R/W		Write to this register indicates that software has acknowledged the pending interrupt and the next interrupt can be sent to the host

Table 11-2626. Register Call Summary for PCIE_ECC_EOI

PCIE_ECC Registers • PCIE_ECC_EOI Register (Offset = 3Ch) [reset = 0h]: [0][1]
PCIe SS Registers • PCIE_ECC Registers: [0][1]
PCIe SS Functional Description • ECC Interrupts: [0]

11.14.5.2.11 PCIE_ECC_INT_STATUS_0 to PCIE_ECC_INT_STATUS_15 Register (Offset = 40h to 7Ch) [reset = 0h]

PCIE_ECC_INT_STATUS_0 to PCIE_ECC_INT_STATUS_15 is shown in Figure 11-1119 and described in Table 11-2628.

These are the raw level interrupt status bits where each bit corresponds to the pending status from an ECC RAM.

Table 11-2627. PCIE_ECC_INT_STATUS_0 to PCIE_ECC_INT_STATUS_15 Instances

Instance	Physical Address
PCIE_0_ECC_CFG	0233 0440h to 0233 047Ch

Figure 11-1119. PCIE_ECC_INT_STATUS_0 to PCIE_ECC_INT_STATUS_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITMASK																															
R-0																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2628. PCIE_ECC_INT_STATUS_0 to PCIE_ECC_INT_STATUS_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITMASK	R	0	Level interrupt status from each ECC RAM: • 0=not pending • 1=pending status

Table 11-2629. Register Call Summary for PCIE_ECC_INT_STATUS_0

PCIE_ECC Registers • PCIE_ECC_INT_STATUS_0 to PCIE_ECC_INT_STATUS_15 Register (Offset = 40h to 7Ch) [reset = 0h]: [0]
PCIe SS Registers • PCIE_ECC Registers: [0]
PCIe SS Functional Description • ECC Interrupts: [0]

11.14.5.2.12 PCIE_ECC_INT_ENABLE_0 to PCIE_ECC_INT_ENABLE_15 Register (Offset = 80h to BCh) [reset = 0h]

PCIE_ECC_INT_ENABLE_0 to PCIE_ECC_INT_ENABLE_15 is shown in Figure 11-1120 and described in Table 11-2631.

These are interrupt enables associated with the interrupt from each of the ECC RAMs. Writing a 1 to a bit position in the register enables the interrupt from the associated ECC RAM.

Table 11-2630. PCIE_ECC_INT_ENABLE_0 to PCIE_ECC_INT_ENABLE_15 Instances

Instance	Physical Address
PCIE_0_ECC_CFG	0233 0480h to 0233 04BCh

Figure 11-1120. PCIE_ECC_INT_ENABLE_0 to PCIE_ECC_INT_ENABLE_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITMASK																															
W-0																															

LEGEND: W = Write Only; *-n* = value after reset

Table 11-2631. PCIE_ECC_INT_ENABLE_0 to PCIE_ECC_INT_ENABLE_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITMASK	W	0	Write 1 to enable interrupt from the associated ECC RAM

Table 11-2632. Register Call Summary for PCIE_ECC_INT_ENABLE_0

PCIE_ECC Registers • PCIE_ECC_INT_ENABLE_0 to PCIE_ECC_INT_ENABLE_15 Register (Offset = 80h to BCh) [reset = 0h]: [0]
PCIe SS Registers • PCIE_ECC Registers: [0]
PCIe SS Functional Description • ECC Interrupts: [0]

11.14.5.2.13 PCIE_ECC_INT_CLEAR_0 to PCIE_ECC_INT_CLEAR_15 Register (Offset = C0h to FCh) [reset = 0h]

PCIE_ECC_INT_CLEAR_0 to PCIE_ECC_INT_CLEAR_15 is shown in Figure 11-1121 and described in Table 11-2634.

These are interrupt enable clear bits associated with the interrupt from each of the ECC RAMs. Writing a 1 to a bit position in the register disables the interrupt from the associated ECC RAM.

Table 11-2633. PCIE_ECC_INT_CLEAR_0 to PCIE_ECC_INT_CLEAR_15 Instances

Instance	Physical Address
PCIE_0_ECC_CFG	0233 04C0h to 0233 04FCh

Figure 11-1121. PCIE_ECC_INT_CLEAR_0 to PCIE_ECC_INT_CLEAR_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITMASK																															
W-0																															

LEGEND: W = Write Only; *-n* = value after reset

Table 11-2634. PCIE_ECC_INT_CLEAR_0 to PCIE_ECC_INT_CLEAR_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITMASK	W	0	Write 1 to disable interrupt from the associated ECC RAM

Table 11-2635. Register Call Summary for PCIE_ECC_INT_CLEAR_0

PCIE_ECC Registers • PCIE_ECC_INT_CLEAR_0 to PCIE_ECC_INT_CLEAR_15 Register (Offset = C0h to FCh) [reset = 0h]: [0]
PCIe SS Registers • PCIE_ECC Registers: [0]

11.14.5.3 Application Registers

The application registers are accessible in multiple ways depending on the point of origin of such accesses. When accessed from the internal bus slave, these registers are accessed via the 4KiB space in Address Space 0. When accessed from the PCIE serial link side, the application registers are mapped to PCIE_BAR0 of an EP as well as a RC. In addition, a RC can access these registers in a PCIESS EP via the upper 1KiB space of the PCIE configuration space. The offsets of the registers do not change irrespective of what the mode of accessing these registers is.

11.14.5.3.1 Register Summary

Table 11-2636. PCIe Instances

Instance	Base Address
PCIE	2180 0000h

Table 11-2637. PCI Express Application Registers

Offset	Acronym	Register Name	PCIE Physical Address	Section
0h	PCIE_PID	Peripheral Version and ID register	2180 0000h	Section 11.14.5.3.2
4h	PCIE_CMD_STATUS	Command Status Register	2180 0004h	Section 11.14.5.3.3
8h	PCIE_CFG_SETUP	Configuration Transaction Setup Register	2180 0008h	Section 11.14.5.3.4
Ch	PCIE_IOBASE	IO TLP Base Register	2180 000Ch	Section 11.14.5.3.5
10h	PCIE_TLPCFG	TLP Attribute Configuration Register	2180 0010h	Section 11.14.5.3.6
14h	PCIE_RSTCMD	Reset Command Register	2180 0014h	Section 11.14.5.3.7
20h	PCIE_PMCMD	Power Management Command Register	2180 0020h	Section 11.14.5.3.8
24h	PCIE_PMCFG	Power Management Configuration Register	2180 0024h	Section 11.14.5.3.9
28h	PCIE_ACT_STATUS	Activity Status Register	2180 0028h	Section 11.14.5.3.10
30h	PCIE_OB_SIZE	Outbound Size Register	2180 0030h	Section 11.14.5.3.11
34h	PCIE_DIAG_CTRL	Diagnostic Control Register	2180 0034h	Section 11.14.5.3.12
38h	PCIE_ENDIAN	Endian Mode Register	2180 0038h	Section 11.14.5.3.13
3Ch	PCIE_PRIORITY	Transaction Priority Register	2180 003Ch	Section 11.14.5.3.14
50h	PCIE_IRQ_EOI	End of Interrupt Register	2180 0050h	Section 11.14.5.3.15
54h	PCIE_MSI_IRQ	MSI Interrupt IRQ Register	2180 0054h	Section 11.14.5.3.16
64h	PCIE_EP_IRQ_SET	Endpoint Interrupt Request Set Register	2180 0064h	Section 11.14.5.3.17
68h	PCIE_EP_IRQ_CLR	Endpoint Interrupt Request Clear Register	2180 0068h	Section 11.14.5.3.18
6Ch	PCIE_EP_IRQ_STATUS	Endpoint Interrupt Status Register	2180 006Ch	Section 11.14.5.3.19
70h	PCIE_GPR0	General Purpose 0 Register	2180 0070h	Section 11.14.5.3.20
74h	PCIE_GPR1	General Purpose 1 Register	2180 0074h	Section 11.14.5.3.21
78h	PCIE_GPR2	General Purpose 2 Register	2180 0078h	Section 11.14.5.3.22
7Ch	PCIE_GPR3	General Purpose 3 Register	2180 007Ch	Section 11.14.5.3.23
100h	PCIE_MSI0_IRQ_STATUS_RAW	MSI 0 Raw Interrupt Status Register	2180 0100h	Section 11.14.5.3.24
104h	PCIE_MSI0_IRQ_STATUS	MSI 0 Interrupt Enabled Status Register	2180 0104h	Section 11.14.5.3.25
108h	PCIE_MSI0_IRQ_ENABLE_SET	MSI 0 Interrupt Enable Set Register	2180 0108h	Section 11.14.5.3.26
10Ch	PCIE_MSI0_IRQ_ENABLE_CLR	MSI 0 Interrupt Enable Clear Register	2180 010Ch	Section 11.14.5.3.27

Table 11-2637. PCI Express Application Registers (continued)

Offset	Acronym	Register Name	PCIE Physical Address	Section
110h	PCIE_MSI1_IRQ_STATUS_RAW	MSI 1 Raw Interrupt Status Register	2180 0110h	Section 11.14.5.3.28
114h	PCIE_MSI1_IRQ_STATUS	MSI 1 Interrupt Enabled Status Register	2180 0114h	Section 11.14.5.3.29
118h	PCIE_MSI1_IRQ_ENABLE_SET	MSI 1 Interrupt Enable Set Register	2180 0118h	Section 11.14.5.3.30
11Ch	PCIE_MSI1_IRQ_ENABLE_CLR	MSI 1 Interrupt Enable Clear Register	2180 011Ch	Section 11.14.5.3.31
120h	PCIE_MSI2_IRQ_STATUS_RAW	MSI 2 Raw Interrupt Status Register	2180 0120h	Section 11.14.5.3.32
124h	PCIE_MSI2_IRQ_STATUS	MSI 2 Interrupt Enabled Status Register	2180 0124h	Section 11.14.5.3.33
128h	PCIE_MSI2_IRQ_ENABLE_SET	MSI 2 Interrupt Enable Set Register	2180 0128h	Section 11.14.5.3.34
12Ch	PCIE_MSI2_IRQ_ENABLE_CLR	MSI 2 Interrupt Enable Clear Register	2180 012Ch	Section 11.14.5.3.35
130h	PCIE_MSI3_IRQ_STATUS_RAW	MSI 3 Raw Interrupt Status Register	2180 0130h	Section 11.14.5.3.36
134h	PCIE_MSI3_IRQ_STATUS	MSI 3 Interrupt Enabled Status Register	2180 0134h	Section 11.14.5.3.37
138h	PCIE_MSI3_IRQ_ENABLE_SET	MSI 3 Interrupt Enable Set Register	2180 0138h	Section 11.14.5.3.38
13Ch	PCIE_MSI3_IRQ_ENABLE_CLR	MSI 3 Interrupt Enable Clear Register	2180 013Ch	Section 11.14.5.3.39
140h	PCIE_MSI4_IRQ_STATUS_RAW	MSI 4 Raw Interrupt Status Register	2180 0140h	Section 11.14.5.3.40
144h	PCIE_MSI4_IRQ_STATUS	MSI 4 Interrupt Enabled Status Register	2180 0144h	Section 11.14.5.3.41
148h	PCIE_MSI4_IRQ_ENABLE_SET	MSI 4 Interrupt Enable Set Register	2180 0148h	Section 11.14.5.3.42
14Ch	PCIE_MSI4_IRQ_ENABLE_CLR	MSI 4 Interrupt Enable Clear Register	2180 014Ch	Section 11.14.5.3.43
150h	PCIE_MSI5_IRQ_STATUS_RAW	MSI 5 Raw Interrupt Status Register	2180 0150h	Section 11.14.5.3.44
154h	PCIE_MSI5_IRQ_STATUS	MSI 5 Interrupt Enabled Status Register	2180 0154h	Section 11.14.5.3.45
158h	PCIE_MSI5_IRQ_ENABLE_SET	MSI 5 Interrupt Enable Set Register	2180 0158h	Section 11.14.5.3.46
15Ch	PCIE_MSI5_IRQ_ENABLE_CLR	MSI 5 Interrupt Enable Clear Register	2180 015Ch	Section 11.14.5.3.47
160h	PCIE_MSI6_IRQ_STATUS_RAW	MSI 6 Raw Interrupt Status Register	2180 0160h	Section 11.14.5.3.48
164h	PCIE_MSI6_IRQ_STATUS	MSI 6 Interrupt Enabled Status Register	2180 0164h	Section 11.14.5.3.49
168h	PCIE_MSI6_IRQ_ENABLE_SET	MSI 6 Interrupt Enable Set Register	2180 0168h	Section 11.14.5.3.50
16Ch	PCIE_MSI6_IRQ_ENABLE_CLR	MSI 6 Interrupt Enable Clear Register	2180 016Ch	Section 11.14.5.3.51
170h	PCIE_MSI7_IRQ_STATUS_RAW	MSI 7 Raw Interrupt Status Register	2180 0170h	Section 11.14.5.3.52
174h	PCIE_MSI7_IRQ_STATUS	MSI 7 Interrupt Enabled Status Register	2180 0174h	Section 11.14.5.3.53
178h	PCIE_MSI7_IRQ_ENABLE_SET	MSI 7 Interrupt Enable Set Register	2180 0178h	Section 11.14.5.3.54

Table 11-2637. PCI Express Application Registers (continued)

Offset	Acronym	Register Name	PCIE Physical Address	Section
17Ch	PCIE_MSI7_IRQ_ENABLE_CLR	MSI 7 Interrupt Enable Clear Register	2180 017Ch	Section 11.14.5.3.55
180h	PCIE_LEGACY_A_IRQ_STATUS_RAW	Raw Interrupt Status Register	2180 0180h	Section 11.14.5.3.56
184h	PCIE_LEGACY_A_IRQ_STATUS	Interrupt Enabled Status Register	2180 0184h	Section 11.14.5.3.57
188h	PCIE_LEGACY_A_IRQ_ENABLE_SET	Interrupt Enable Set Register	2180 0188h	Section 11.14.5.3.58
18Ch	PCIE_LEGACY_A_IRQ_ENABLE_CLR	Interrupt Enable Clear Register	2180 018Ch	Section 11.14.5.3.59
190h	PCIE_LEGACY_B_IRQ_STATUS_RAW	Raw Interrupt Status Register	2180 0190h	Section 11.14.5.3.60
194h	PCIE_LEGACY_B_IRQ_STATUS	Interrupt Enabled Status Register	2180 0194h	Section 11.14.5.3.61
198h	PCIE_LEGACY_B_IRQ_ENABLE_SET	Interrupt Enable Set Register	2180 0198h	Section 11.14.5.3.62
19Ch	PCIE_LEGACY_B_IRQ_ENABLE_CLR	Interrupt Enable Clear Register	2180 019Ch	Section 11.14.5.3.63
1A0h	PCIE_LEGACY_C_IRQ_STATUS_RAW	Raw Interrupt Status Register	2180 01A0h	Section 11.14.5.3.64
1A4h	PCIE_LEGACY_C_IRQ_STATUS	Interrupt Enabled Status Register	2180 01A4h	Section 11.14.5.3.65
1A8h	PCIE_LEGACY_C_IRQ_ENABLE_SET	Interrupt Enable Set Register	2180 01A8h	Section 11.14.5.3.66
1ACh	PCIE_LEGACY_C_IRQ_ENABLE_CLR	Interrupt Enable Clear Register	2180 01ACh	Section 11.14.5.3.67
1B0h	PCIE_LEGACY_D_IRQ_STATUS_RAW	Raw Interrupt Status Register	2180 01B0h	Section 11.14.5.3.68
1B4h	PCIE_LEGACY_D_IRQ_STATUS	Interrupt Enabled Status Register	2180 01B4h	Section 11.14.5.3.69
1B8h	PCIE_LEGACY_D_IRQ_ENABLE_SET	Interrupt Enable Set Register	2180 01B8h	Section 11.14.5.3.70
1BCh	PCIE_LEGACY_D_IRQ_ENABLE_CLR	Interrupt Enable Clear Register	2180 01BCh	Section 11.14.5.3.71
1C0h	PCIE_ERR_IRQ_STATUS_RAW	Raw ERR Interrupt Status Register	2180 01C0h	Section 11.14.5.3.72
1C4h	PCIE_ERR_IRQ_STATUS	ERR Interrupt Enabled Status Register	2180 01C4h	Section 11.14.5.3.73
1C8h	PCIE_ERR_IRQ_ENABLE_SET	ERR Interrupt Enable Set Register	2180 01C8h	Section 11.14.5.3.74
1CCh	PCIE_ERR_IRQ_ENABLE_CLR	ERR Interrupt Enable Clear Register	2180 01CCh	Section 11.14.5.3.75
1D0h	PCIE_PMRST_IRQ_STATUS_RAW	Power Management and Reset Interrupt Status Register	2180 01D0h	Section 11.14.5.3.76
1D4h	PCIE_PMRST_IRQ_STATUS	Power Management and Reset Interrupt Enabled Status Register	2180 01D4h	Section 11.14.5.3.77
1D8h	PCIE_PMRST_ENABLE_SET	Power Management and Reset Interrupt Enable Set Register	2180 01D8h	Section 11.14.5.3.78
1DCh	PCIE_PMRST_ENABLE_CLR	Power Management and Reset Interrupt Enable Clear Register	2180 01DCh	Section 11.14.5.3.79
(0x200 + N*8)[(1)]	PCIE_OB_OFFSET_INDEXn	Outbound Translation Region N Offset Low and Index Register (0x200 + N*8)	2180 0000h + (0x200 + N*8)	Section 11.14.5.3.80
(0x200 + N*8 + 0x4)[(1)]	PCIE_OB_OFFSETn_HI	Outbound Translation Region N Offset High Register (0x200 + N*8 + 0x4)	2180 0204h + (0x200 + N*8)	Section 11.14.5.3.81
300h	PCIE_IB_BAR0	Inbound Translation Bar Match 0 Register	2180 0300h	Section 11.14.5.3.82
304h	PCIE_IB_START0_LO	Inbound Translation 0 Start Address Low Register	2180 0304h	Section 11.14.5.3.83
308h	PCIE_IB_START0_HI	Inbound Translation 0 Start Address High Register	2180 0308h	Section 11.14.5.3.84
30Ch	PCIE_IB_OFFSET0	Inbound Translation 0 Address Offset Register	2180 030Ch	Section 11.14.5.3.85
310h	PCIE_IB_BAR1	Inbound Translation Bar Match 1 Register	2180 0310h	Section 11.14.5.3.86

(1) N = 0 to 31

Table 11-2637. PCI Express Application Registers (continued)

Offset	Acronym	Register Name	PCIE Physical Address	Section
314h	PCIE_IB_START1_LO	Inbound Translation 1 Start Address Low Register	2180 0314h	Section 11.14.5.3.87
318h	PCIE_IB_START1_HI	Inbound Translation 1 Start Address High Register	2180 0318h	Section 11.14.5.3.88
31Ch	PCIE_IB_OFFSET1	Inbound Translation 1 Address Offset Register	2180 031Ch	Section 11.14.5.3.89
320h	PCIE_IB_BAR2	Inbound Translation Bar Match 2 Register	2180 0320h	Section 11.14.5.3.90
324h	PCIE_IB_START2_LO	Inbound Translation 2 Start Address Low Register	2180 0324h	Section 11.14.5.3.91
328h	PCIE_IB_START2_HI	Inbound Translation 2 Start Address High Register	2180 0328h	Section 11.14.5.3.92
32Ch	PCIE_IB_OFFSET2	Inbound Translation 2 Address Offset Register	2180 032Ch	Section 11.14.5.3.93
330h	PCIE_IB_BAR3	Inbound Translation Bar Match 3 Register	2180 0330h	Section 11.14.5.3.94
334h	PCIE_IB_START3_LO	Inbound Translation 3 Start Address Low Register	2180 0334h	Section 11.14.5.3.95
338h	PCIE_IB_START3_HI	Inbound Translation 3 Start Address High Register	2180 0338h	Section 11.14.5.3.96
33Ch	PCIE_IB_OFFSET3	Inbound Translation 3 Address Offset Register	2180 033Ch	Section 11.14.5.3.97

11.14.5.3.2 PCIE_PID Register (Offset = 0h) [reset = 4E330900h]

The Peripheral Version and ID Register (PCIE_PID) is shown in Figure 11-1122 and described in Table 11-2639.

Table 11-2638. PCIE_PID Instances

Instance	Physical Address
PCIE	2180 0000h

Figure 11-1122. PCIE_PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 4E330900h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2639. PCIE_PID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4E330900h	TI internal data. Identifies revision of peripheral.

Table 11-2640. Register Call Summary for PCIE_PID

Application Registers
• Register Summary: [0]

11.14.5.3.3 PCIE_CMD_STATUS Register (Offset = 4h) [reset = 0h]

The Command Status Register (PCIE_CMD_STATUS) is shown in and Figure 11-1123 described in Table 11-2642.

Table 11-2641. PCIE_CMD_STATUS Instances

Instance	Physical Address
PCIE	2180 0004h

Figure 11-1123. PCIE_CMD_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		DBI_CS2	APP_RETRY_EN	POSTED_WR_EN	IB_XLT_EN	OB_XLT_EN	LTSSM_EN
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2642. PCIE_CMD_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reads return 0 and writes have no effect.
5	DBI_CS2	R/W	0h	Set to enable writing to BAR mask registers that are overlaid on BAR registers. • 0 = Disable writing to BAR mask registers. • 1 = Enable writing to BAR mask registers.
4	APP_RETRY_EN	R/W	0h	Application request retry enable. This feature can be used if initialization can take longer than PCIe stipulated time frame. • 0 = Disable all incoming PCIe transactions to be returned with a retry response. • 1 = Enable all incoming PCIe transactions to be returned with a retry response.
3	POSTED_WR_EN	R/W	0h	Posted write enable. Default is 0 with all internal bus master writes defaulting to non-posted. • 0 = Disable the internal bus master to use posted write commands. • 1 = Enable the internal bus master to use posted write commands. NOTE: This bit is not applicable for KeyStone device.
2	IB_XLT_EN	R/W	0h	Inbound address translation enable. • 0 = Disable translation of inbound memory/IO read/write requests into memory read/write requests. • 1 = Enable translation of inbound memory/IO read/write requests into memory read/write requests.
1	OB_XLT_EN	R/W	0h	Outbound address translation enable. • 0 = Disable translation of outbound memory read/write requests into memory/IO/configuration read/write requests. • 1 = Enable translation of outbound memory read/write requests into memory/IO/configuration read/write requests.

Table 11-2642. PCIE_CMD_STATUS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	LTSSM_EN	R/W	0h	Link training enable. • 0 = Disable LTSSM in PCI Express core. • 1 = Enable LTSSM in PCI Express core and link negotiation with link partner will begin.

Table 11-2643. Register Call Summary for PCIE_CMD_STATUS

Configuration Type 0 Registers • PCIE_BAR5_MASK (64 bit BAR4) Register (Offset = 1024h) [reset = 0h]: [0] • PCIE_BAR4_MASK Register (Offset = 1020h) [reset = 0h]: [0] • PCIE_BAR3_MASK (64 bit BAR2) Register (Offset = 101Ch) [reset = 0h]: [0] • PCIE_BAR2_MASK Register (Offset = 1018h) [reset = 0h]: [0] • PCIE_BAR5_MASK Register (Offset = 1024h) [reset = 0h]: [0] • PCIE_BAR0_MASK Register (Offset = 1010h) [reset = 0h]: [0] • PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR3_MASK Register (Offset = 101Ch) [reset = 0h]: [0]
Application Registers • PCIE_CMD_STATUS Register (Offset = 4h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • BAR Mask Registers: [0][1] • Initialization Sequence: [0][1][2][3] • Accessing Read-Only Registers in Configuration Space: [0] • Error Reporting: [0] • Initialization Sequence: [0][1][2] • Hot Reset Request Interrupt: [0]
Configuration Type 1 Registers • PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR0_MASK Register (Offset = 1010h) [reset = 0h]: [0] • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0]

11.14.5.3.4 PCIE_CFG_SETUP Register (Offset = 8h) [reset = 0h]

The Configuration Transaction Setup Register (PCIE_CFG_SETUP) is shown in Figure 11-1124 and described in Table 11-2645.

Table 11-2644. PCIE_CFG_SETUP Instances

Instance	Physical Address
PCIE	2180 0008h

Figure 11-1124. PCIE_CFG_SETUP Register

31	30	29	28	27	26	25	24
RESERVED							CFG_TYPE
R-0h							R/W-0h
23	22	21	20	19	18	17	16
CFG_BUS							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED			CFG_DEVICE				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
RESERVED					CFG_FUNC		
R-0h					R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2645. PCIE_CFG_SETUP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reads return 0 and writes have no effect.
24	CFG_TYPE	R/W	0h	Configuration type for outbound configuration accesses. • 0 = Type 0 access. • 1 = Type 1 access.
23-16	CFG_BUS	R/W	0h	PCIe bus number for outbound configuration accesses (value = 0-FFh).
15-13	RESERVED	R	0h	Reads return 0 and writes have no effect.
12-8	CFG_DEVICE	R/W	0h	PCIe device number for outbound configuration accesses (value = 0-1Fh).
7-3	RESERVED	R	0h	Reads return 0 and writes have no effect.
2-0	CFG_FUNC	R/W	0h	PCIe function number for outbound configuration accesses (value = 0-7h).

Table 11-2646. Register Call Summary for PCIE_CFG_SETUP

Application Registers • PCIE_CFG_SETUP Register (Offset = 8h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • Address Space 0: [0]

11.14.5.3.5 PCIE_IOBASE Register (Offset = Ch) [reset = 0h]

The IO TLP Base Register (PCIE_IOBASE) is shown in Figure 11-1125 and described in Table 11-2648.

Table 11-2647. PCIE_IOBASE Instances

Instance	Physical Address
PCIE	2180 000Ch

Figure 11-1125. PCIE_IOBASE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IOBASE																				RESERVED											
R/W-0h																				R-0h										

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2648. PCIE_IOBASE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	IOBASE	R/W	0h	Bits [31-12] of outgoing IO TLP (value = 0-FFFFFh). RC mode only.
11-0	RESERVED	R	0h	Reads return 0 and writes have no effect.

Table 11-2649. Register Call Summary for PCIE_IOBASE

Application Registers • PCIE_IOBASE Register (Offset = Ch) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • Address Space 0: [0]

11.14.5.3.6 PCIE_TLPCFG Register (Offset = 10h) [reset = 0h]

The TLP Attribute Configuration Register (PCIE_TLPCFG) is shown in Figure 11-1126 and described in Table 11-2651.

Table 11-2650. PCIE_TLPCFG Instances

Instance	Physical Address
PCIE	2180 0010h

Figure 11-1126. PCIE_TLPCFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RELAXED	NO_SNOOP
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2651. PCIE_TLPCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reads return 0 and writes have no effect.
1	RELAXED	R/W	0h	Enable relaxed ordering for all outgoing TLPs. • 0 = Disable relaxed ordering for all outgoing TLPs. • 1 = Enable relaxed ordering for all outgoing TLPs.
0	NO_SNOOP	R/W	0h	Enable No Snoop attribute on all outgoing TLPs. • 0 = Disable no snoop attribute on all outgoing TLPs. • 1 = Enable no snoop attribute on all outgoing TLPs.

Table 11-2652. Register Call Summary for PCIE_TLPCFG

Application Registers
- PCIE_TLPCFG Register (Offset = 10h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.7 PCIE_RSTCMD Register (Offset = 14h) [reset = 10000h]

The Reset Command Register (PCIE_RSTCMD) is shown in Figure 11-1127 and described in Table 11-2654.

Table 11-2653. PCIE_RSTCMD Instances

Instance	Physical Address
PCIE	2180 0014h

Figure 11-1127. PCIE_RSTCMD Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							FLUSH_N
R-0h							R-1h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INIT_RST
R-0h							W1S-0h

LEGEND: R = Read Only; W1S = Write 1 to Set Bit; -*n* = value after reset

Table 11-2654. PCIE_RSTCMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reads return 0 and writes have no effect.
16	FLUSH_N	R	1h	Bridge flush status. Used to ensure no pending transactions prior to issuing warm reset. • 0 = No transaction is pending. • 1 = There are transactions pending.
15-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INIT_RST	W1S	0h	Write 1 to initiate a downstream hot reset sequence on downstream. • 0 = No effect. • 1 = Initiate a downstream hot rest sequence on downstream.

Table 11-2655. Register Call Summary for PCIE_RSTCMD

Application Registers • PCIE_RSTCMD Register (Offset = 14h) [reset = 10000h]: [0] • Register Summary: [0]
PCIe SS Functional Description • Software Reset Considerations: [0]

11.14.5.3.8 PCIE_PMCMD Register (Offset = 20h) [reset = 0h]

The Power Management Command Register (PCIE_PMCMD) is shown in Figure 11-1128 and described in Table 11-2657.

Table 11-2656. PCIE_PMCMD Instances

Instance	Physical Address
PCIE	2180 0020h

Figure 11-1128. PCIE_PMCMD Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						PM_XMT_TURNOFF	PM_XMT_PME
R-0h						W1S-0h	W1S-0h

LEGEND: R = Read Only; W1S = Write 1 to Set Bit; *-n* = value after reset

Table 11-2657. PCIE_PMCMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reads return 0 and writes have no effect.
1	PM_XMT_TURNOFF	W1S	0h	Write 1 to transmit a PM_TURNOFF message. Reads 0. Applicable in RC mode only. • 0 = No effect • 1 = Transmit a PM_TURNOFF message
0	PM_XMT_PME	W1S	0h	Write 1 to transmit a PM_PME message. Reads 0. Applicable to EP mode only. • 0 = No effect • 1 = Transmit a PM_PME message

Table 11-2658. Register Call Summary for PCIE_PMCMD

Application Registers
- PCIE_PMCMD Register (Offset = 20h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.9 PCIE_PMCFG Register (Offset = 24h) [reset = 0h]

The Power Management Configuration Register (PCIE_PMCFG) is shown in Figure 11-1129and described in Table 11-2660.

Table 11-2659. PCIE_PMCFG Instances

Instance	Physical Address
PCIE	2180 0024h

Figure 11-1129. PCIE_PMCFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ENTR_L23
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2660. PCIE_PMCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	ENTR_L23	R/W	0h	Write 1 to enable entry to L2/L3 ready state. Read to check L2/L3 entry readiness. Applicable to RC and EP. • 0 = Disable entry to L2/L3 ready state. • 1 = Enable entry to L2/L3 ready state.

Table 11-2661. Register Call Summary for PCIE_PMCFG

Application Registers • PCIE_PMCFG Register (Offset = 24h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • Software Initiated L2/L3 Ready state: [0]

11.14.5.3.10 PCIE_ACT_STATUS Register (Offset = 28h) [reset = 0h]

The Activity Status Register (PCIE_ACT_STATUS) is shown in Figure 11-1130and described in Table 11-2663.

Table 11-2662. PCIE_ACT_STATUS Instances

Instance	Physical Address
PCIE	2180 0028h

Figure 11-1130. PCIE_ACT_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						OB_NOT_EMPTY	IB_NOT_EMPTY
R-0h						R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2663. PCIE_ACT_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reads return 0 and writes have no effect.
1	OB_NOT_EMPTY	R	0h	• 0 = Outbound buffers are empty • 1 = Outbound buffers are not empty
0	IB_NOT_EMPTY	R	0h	• 0 = Inbound buffers are empty • 1 = Inbound buffers are not empty

Table 11-2664. Register Call Summary for PCIE_ACT_STATUS

Application Registers
- PCIE_ACT_STATUS Register (Offset = 28h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.11 PCIE_OB_SIZE Register (Offset = 30h) [reset = 0h]

The Outbound Size Register (PCIE_OB_SIZE) is shown in Figure 11-1131 and described in Table 11-2666.

Table 11-2665. PCIE_OB_SIZE Instances

Instance	Physical Address
PCIE	2180 0030h

Figure 11-1131. PCIE_OB_SIZE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					OB_SIZE		
R-0h					R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2666. PCIE_OB_SIZE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reads return 0 and writes have no effect.
2-0	OB_SIZE	R/W	0h	Set each outbound translation window size. Applicable to RC and EP. • 0 = 1MB • 1h = 2MB • 2h = 4MB • 3h = 8MB • 4-7h = Reserved

Table 11-2667. Register Call Summary for PCIE_OB_SIZE

Application Registers • PCIE_OB_SIZE Register (Offset = 30h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • Outbound Address Translation: [0] • Outbound Address Translation: [0][1][2]

11.14.5.3.12 PCIE_DIAG_CTRL Register (Offset = 34h) [reset = 0h]

The Diagnostic Control Register (PCIE_DIAG_CTRL) is shown in Figure 11-1132 and described in Table 11-2669.

Table 11-2668. PCIE_DIAG_CTRL Instances

Instance	Physical Address
PCIE	2180 0034h

Figure 11-1132. PCIE_DIAG_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						INV_ECRC	INV_LCRC
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2669. PCIE_DIAG_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reads return 0 and writes have no effect.
1	INV_ECRC	R/W	0h	Write 1 to force inversion of LSB of ECRC for the next one packet. It is self cleared when the ECRC error has been injected on one TLP.
0	INV_LCRC	R/W	0h	Write 1 to force inversion of LSB of LCRC for the next one packet. It is self cleared when the LCRC error has been injected on one TLP.

Table 11-2670. Register Call Summary for PCIE_DIAG_CTRL

Application Registers
- PCIE_DIAG_CTRL Register (Offset = 34h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.13 PCIE_ENDIAN Register (Offset = 38h) [reset = 0h]

The Endian Mode Register (PCIE_ENDIAN) is shown in Figure 11-1133 and described in Table 11-2672.

Table 11-2671. PCIE_ENDIAN Instances

Instance	Physical Address
PCIE	2180 0038h

Figure 11-1133. PCIE_ENDIAN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ENDIAN_MODE	
R-0h						R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2672. PCIE_ENDIAN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reads return 0 and writes have no effect.
1-0	ENDIAN_MODE	R/W	0h	Endian mode.

Table 11-2673. Register Call Summary for PCIE_ENDIAN

Application Registers
- PCIE_ENDIAN Register (Offset = 38h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.14 PCIE_PRIORITY Register (Offset = 3Ch) [reset = 0h]

The Transaction Priority Register is shown in Figure 11-1134 and described in Table 11-2675.

Table 11-2674. PCIE_PRIORITY Instances

Instance	Physical Address
PCIE	2180 003Ch

Figure 11-1134. PCIE_PRIORITY Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							MST_PRIV
R-0h							R/W-0h
15	14	13	12	11	10	9	8
RESERVED				MST_PRIVID			
R-0h				R-0h			
7	6	5	4	3	2	1	0
RESERVED					MST_PRIORITY		
R-0h					R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2675. PCIE_PRIORITY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reads return 0 and writes have no effect.
16	MST_PRIV	R/W	0h	Master transaction mode. PCIe transaction needs to be in supervisor mode to access the device registers. • 0 = PCIe transaction is in user mode. • 1 = PCIe transaction is in supervisor mode.
15-12	RESERVED	R	0h	Reads return 0 and writes have no effect.
11-8	MST_PRIVID	R	0h	Master PRIVID value on master transactions.
7-3	RESERVED	R	0h	Reads return 0 and writes have no effect.
2-0	MST_PRIORITY	R/W	0h	Priority level for each inbound transaction on the internal bus master port. 0 is the highest priority level and 7h is the lowest priority level.

Table 11-2676. Register Call Summary for PCIE_PRIORITY

Application Registers
- Register Summary: [0]

11.14.5.3.15 PCIE_IRQ_EOI Register (Offset = 50h) [reset = 0h]

The End of Interrupt Register (PCIE_IRQ_EOI) is shown in Figure 11-1135 and described in Table 11-2678.

Table 11-2677. PCIE_IRQ_EOI Instances

Instance	Physical Address
PCIE	2180 0050h

Figure 11-1135. PCIE_IRQ_EOI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												EOI			
R-0h																												W-0h			

LEGEND: R = Read Only; W = Write Only; *-n* = value after reset

Table 11-2678. PCIE_IRQ_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	EOI	W	0h	EOI for interrupts. Write the interrupt event number to indicate end-of-interrupt for the interrupt events. Write 0 to mark EOI for INTA, 1 for INTB and so on. Please see the Table 11-2571 for the interrupt event number.

Table 11-2679. Register Call Summary for PCIE_IRQ_EOI

Application Registers • PCIE_IRQ_EOI Register (Offset = 50h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • MSI Interrupts Reception in RC Mode: [0] • Interrupts: [0] • Interrupt Reception in EP Mode: [0]

11.14.5.3.16 PCIE_MSI_IRQ Register (Offset = 54h) [reset = 0h]

The MSI Interrupt IRQ Register (PCIE_MSI_IRQ) is shown in Figure 11-1136 and described in Table 11-2681.

Table 11-2680. PCIE_MSI_IRQ Instances

Instance	Physical Address
PCIE	2180 0054h

Figure 11-1136. PCIE_MSI_IRQ Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSI_IRQ																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 11-2681. PCIE_MSI_IRQ Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MSI_IRQ	W	0h	This register is written to by the remote device. Writes initiated by an EP over PCIe link that target PCIE_BAR0 of the RC land to this register if the offset matches. EP should write MSI vector value to this register to generate corresponding MSI Interrupt, such as writing 0x0 to generate MSI_0 interrupt (with vector 0), writing 0x1F to generate MSI_7 interrupt (with vector 31).

Table 11-2682. Register Call Summary for PCIE_MSI_IRQ

Application Registers • PCIE_MSI_IRQ Register (Offset = 54h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • MSI Interrupt Generation in EP Mode: [0][1][2][3] • Interrupt Generation in RC Mode: [0] • Interrupt Reception in EP Mode: [0]

11.14.5.3.17 PCIE_EP_IRQ_SET Register (Offset = 64h) [reset = 0h]

The Endpoint Interrupt Request Set Register (PCIE_EP_IRQ_SET) is shown in Figure 11-1137 and described in Table 11-2684.

Table 11-2683. PCIE_EP_IRQ_SET Instances

Instance	Physical Address
PCIE	2180 0064h

Figure 11-1137. PCIE_EP_IRQ_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EP_IRQ_SET
R-0h							R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2684. PCIE_EP_IRQ_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	EP_IRQ_SET	R/W1S	0h	Write 1 to generate assert interrupt message. If MSI is disabled, legacy interrupt assert message will be generated. On read, a 1 indicates currently asserted interrupt.

Table 11-2685. Register Call Summary for PCIE_EP_IRQ_SET

Application Registers • PCIE_EP_IRQ_SET Register (Offset = 64h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • Legacy Interrupt Generation in EP Mode: [0]

11.14.5.3.18 PCIE_EP_IRQ_CLR Register (Offset = 68h) [reset = 0h]

The Endpoint Interrupt Request Clear Register (PCIE_EP_IRQ_CLR) is shown in Figure 11-1138 and described in Table 11-2687.

Table 11-2686. PCIE_EP_IRQ_CLR Instances

Instance	Physical Address
PCIE	2180 0068h

Figure 11-1138. PCIE_EP_IRQ_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EP_IRQ_CLR
R-0h							R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-2687. PCIE_EP_IRQ_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	EP_IRQ_CLR	R/W1C	0h	Write 1 to generate deassert interrupt message. If MSI is disabled, legacy interrupt deassert message will be generated. On read, a 1 indicates currently asserted interrupt.

Table 11-2688. Register Call Summary for PCIE_EP_IRQ_CLR

Application Registers • PCIE_EP_IRQ_CLR Register (Offset = 68h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • Legacy Interrupt Generation in EP Mode: [0]

11.14.5.3.19 PCIE_EP_IRQ_STATUS Register (Offset = 6Ch) [reset = 0h]

The Endpoint Interrupt Status Register (PCIE_EP_IRQ_STATUS) is shown in Figure 11-1139 and described in Table 11-2690.

Table 11-2689. PCIE_EP_IRQ_STATUS Instances

Instance	Physical Address
PCIE	2180 006Ch

Figure 11-1139. PCIE_EP_IRQ_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EP_IRQ_STATUS
R-0h							R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2690. PCIE_EP_IRQ_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	EP_IRQ_STATUS	R	0h	Indicates whether interrupt for function 0 is asserted or not. • 0 = Interrupt for function 0 is not asserted. • 1 = Interrupt for function 0 is asserted.

Table 11-2691. Register Call Summary for PCIE_EP_IRQ_STATUS

Application Registers • PCIE_EP_IRQ_STATUS Register (Offset = 6Ch) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • Legacy Interrupt Generation in EP Mode: [0]

11.14.5.3.20 PCIE_GPR0 Register (Offset = 70h) [reset = 0h]

The General Purpose 0 Register (PCIE_GPR0) is shown in Figure 11-1140 and described in Table 11-2693.

Table 11-2692. PCIE_GPR0 Instances

Instance	Physical Address
PCIE	2180 0070h

Figure 11-1140. PCIE_GPR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GENERIC0																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2693. PCIE_GPR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GENERIC0	R/W	0h	Generic info field 0 (value = 0-FFFFFFFFh).

Table 11-2694. Register Call Summary for PCIE_GPR0

Application Registers
- PCIE_GPR0 Register (Offset = 70h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.21 PCIE_GPR1 Register (Offset = 74h) [reset = 0h]

The General Purpose 1 Register (PCIE_GPR1) is shown in Figure 11-1141 and described in Table 11-2696.

Table 11-2695. PCIE_GPR1 Instances

Instance	Physical Address
PCIE	2180 0074h

Figure 11-1141. PCIE_GPR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GENERIC1																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2696. PCIE_GPR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GENERIC1	R/W	0h	Generic info field 1 (value = 0-FFFFFFFFh).

Table 11-2697. Register Call Summary for PCIE_GPR1

Application Registers
- Register Summary: [0]
- PCIE_GPR1 Register (Offset = 74h) [reset = 0h]: [0]

11.14.5.3.22 PCIE_GPR2 Register (Offset = 78h) [reset = 0h]

The General Purpose 2 Register (PCIE_GPR2) is shown in Figure 11-1142 and described in Table 11-2699.

Table 11-2698. PCIE_GPR2 Instances

Instance	Physical Address
PCIE	2180 0078h

Figure 11-1142. PCIE_GPR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GENERIC2																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2699. PCIE_GPR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GENERIC2	R/W	0h	Generic info field 2 (value = 0-FFFFFFFFh).

Table 11-2700. Register Call Summary for PCIE_GPR2

Application Registers
- PCIE_GPR2 Register (Offset = 78h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.23 PCIE_GPR3 Register (Offset = 7Ch) [reset = 0h]

The General Purpose 3 Register (PCIE_GPR3) is shown in Figure 11-1143 and described in Table 11-2702.

Table 11-2701. PCIE_GPR3 Instances

Instance	Physical Address
PCIE	2180 007Ch

Figure 11-1143. PCIE_GPR3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GENERIC3																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2702. PCIE_GPR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GENERIC3	R/W	0h	Generic info field 3 (value = 0-FFFFFFFFh).

Table 11-2703. Register Call Summary for PCIE_GPR3

Application Registers
- PCIE_GPR3 Register (Offset = 7Ch) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.24 PCIE_MSI0_IRQ_STATUS_RAW Register (Offset = 100h) [reset = 0h]

The MSI 0 Interrupt Raw Status Register (MSI0_RAW_STATUS) is shown in Figure 11-1144 and described in Table 11-2705.

Table 11-2704. PCIE_MSI0_IRQ_STATUS_RAW Instances

Instance	Physical Address
PCIE	2180 0100h

Figure 11-1144. PCIE_MSI0_IRQ_STATUS_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI0_RAW_STATUS			
R-0h				R/W1S-0h			

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2705. PCIE_MSI0_IRQ_STATUS_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI0_RAW_STATUS	R/W1S	0h	Each bit indicates raw status of MSI vectors (24, 16, 8, 0) associated with the bit. Typically, writes to this register are only done for debug purposes. • bit 3 = MSI vector 24 status • bit 2 = MSI vector 16 status • bit 1 = MSI vector 8 status • bit 0 = MSI vector 0 status

Table 11-2706. Register Call Summary for PCIE_MSI0_IRQ_STATUS_RAW

Application Registers
• Register Summary: [0]

11.14.5.3.25 PCIE_MSI0_IRQ_STATUS Register (Offset = 104h) [reset = 0h]

The MSI 0 Interrupt Enabled Status Register (PCIE_MSI0_IRQ_STATUS) is shown in Figure 11-1145 and described in Table 11-2708.

Table 11-2707. PCIE_MSI0_IRQ_STATUS Instances

Instance	Physical Address
PCIE	2180 0104h

Figure 11-1145. PCIE_MSI0_IRQ_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI0_IRQ_STATUS			
R-0h				R/W1C-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-2708. PCIE_MSI0_IRQ_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI0_IRQ_STATUS	R/W1C	0h	Each bit indicates status of MSI vector (24, 16, 8, 0) associated with the bit. Each of the bits can be written with 1 to clear the respective interrupt status bit. • bit 3 = MSI vector 24 status • bit 2 = MSI vector 16 status • bit 1 = MSI vector 8 status • bit 0 = MSI vector 0 status

Table 11-2709. Register Call Summary for PCIE_MSI0_IRQ_STATUS

Application Registers • PCIE_MSI0_IRQ_STATUS Register (Offset = 104h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • MSI Interrupts Reception in RC Mode: [0][1] • Interrupt Reception in EP Mode: [0][1]

11.14.5.3.26 PCIE_MSI0_IRQ_ENABLE_SET Register (Offset = 108h) [reset = 0h]

The MSI 0 Interrupt Enable Set Register (PCIE_MSI0_IRQ_ENABLE_SET) is shown in Figure 11-1146 and described in Table 11-2711.

Table 11-2710. PCIE_MSI0_IRQ_ENABLE_SET Instances

Instance	Physical Address
PCIE	2180 0108h

Figure 11-1146. PCIE_MSI0_IRQ_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI0_IRQ_EN_SET			
R-0h				R/W1S-0h			

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2711. PCIE_MSI0_IRQ_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI0_IRQ_EN_SET	R/W1S	0h	Each bit, when written to, enables the MSI interrupt (24, 16, 8, 0) associated with the bit. • bit 3 = MSI vector 24 enable • bit 2 = MSI vector 16 enable • bit 1 = MSI vector 8 enable • bit 0 = MSI vector 0 enable

Table 11-2712. Register Call Summary for PCIE_MSI0_IRQ_ENABLE_SET

Application Registers • Register Summary: [0] • PCIE_MSI0_IRQ_ENABLE_SET Register (Offset = 108h) [reset = 0h]: [0]
PCIe SS Functional Description • MSI Interrupts Reception in RC Mode: [0] • Interrupt Reception in EP Mode: [0]

11.14.5.3.27 PCIE_MSI0_IRQ_ENABLE_CLR Register (Offset = 10Ch) [reset = 0h]

The MSI 0 Interrupt Enable Clear Register (PCIE_MSI0_IRQ_ENABLE_CLR) is shown in Figure 11-1147 and described in Table 11-2714.

Table 11-2713. PCIE_MSI0_IRQ_ENABLE_CLR Instances

Instance	Physical Address
PCIE	2180 010Ch

Figure 11-1147. PCIE_MSI0_IRQ_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI0_IRQ_EN_CLR			
R-0h				R/W1C-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2714. PCIE_MSI0_IRQ_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI0_IRQ_EN_CLR	R/W1C	0h	Each bit, when written to, disables the MSI interrupt (24, 16, 8, 0) associated with the bit. • bit 3 = MSI vector 24 disable • bit 2 = MSI vector 16 disable • bit 1 = MSI vector 8 disable • bit 0 = MSI vector 0 disable

Table 11-2715. Register Call Summary for PCIE_MSI0_IRQ_ENABLE_CLR

Application Registers • PCIE_MSI0_IRQ_ENABLE_CLR Register (Offset = 10Ch) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • MSI Interrupts Reception in RC Mode: [0] • Interrupt Reception in EP Mode: [0]

11.14.5.3.28 PCIE_MSI1_IRQ_STATUS_RAW Register (Offset = 110h) [reset = 0h]

The MSI 1 Interrupt Raw Status Register (MSI1_RAW_STATUS) is shown in Figure 11-1148 and described in Table 11-2717.

Table 11-2716. PCIE_MSI1_IRQ_STATUS_RAW Instances

Instance	Physical Address
PCIE	2180 0110h

Figure 11-1148. PCIE_MSI1_IRQ_STATUS_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI1_RAW_STATUS			
R-0h				R/W1S-0h			

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2717. PCIE_MSI1_IRQ_STATUS_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI1_RAW_STATUS	R/W1S	0h	Each bit indicates raw status of MSI vectors (25, 17, 9, 1) associated with the bit. Typically, writes to this register are only done for debug purposes. • bit 3 = MSI vector 25 status • bit 2 = MSI vector 17 status • bit 1 = MSI vector 9 status • bit 0 = MSI vector 1 status

Table 11-2718. Register Call Summary for PCIE_MSI1_IRQ_STATUS_RAW

Application Registers
• Register Summary: [0]

11.14.5.3.29 PCIE_MSI1_IRQ_STATUS Register (Offset = 114h) [reset = 0h]

The MSI 1 Interrupt Enabled Status Register (PCIE_MSI1_IRQ_STATUS) is shown in Figure 11-1149 and described in Table 11-2720.

Table 11-2719. PCIE_MSI1_IRQ_STATUS Instances

Instance	Physical Address
PCIE	2180 0114h

Figure 11-1149. PCIE_MSI1_IRQ_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI1_IRQ_STATUS			
R-0h				R/W1C-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-2720. PCIE_MSI1_IRQ_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI1_IRQ_STATUS	R/W1C	0h	Each bit indicates status of MSI vector (25, 17, 9, 1) associated with the bit. Each of the bits can be written with one to clear the respective interrupt status bit. • bit 3 = MSI vector 25 status • bit 2 = MSI vector 17 status • bit 1 = MSI vector 9 status • bit 0 = MSI vector 1 status

Table 11-2721. Register Call Summary for PCIE_MSI1_IRQ_STATUS

Application Registers
- PCIE_MSI1_IRQ_STATUS Register (Offset = 114h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.30 PCIE_MSI1_IRQ_ENABLE_SET Register (Offset = 118h) [reset = 0h]

The MSI 1 Interrupt Enable Set Register (PCIE_MSI1_IRQ_ENABLE_SET) is shown in Figure 11-1150 and described in Table 11-2723.

Table 11-2722. PCIE_MSI1_IRQ_ENABLE_SET Instances

Instance	Physical Address
PCIE	2180 0118h

Figure 11-1150. PCIE_MSI1_IRQ_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI1_IRQ_EN_SET			
R-0h				R/W1S-0h			

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2723. PCIE_MSI1_IRQ_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI1_IRQ_EN_SET	R/W1S	0h	Each bit, when written to, enables the MSI interrupt (25, 17, 9, 1) associated with the bit. • bit 3 = MSI vector 25 enable • bit 2 = MSI vector 17 enable • bit 1 = MSI vector 9 enable • bit 0 = MSI vector 1 enable

Table 11-2724. Register Call Summary for PCIE_MSI1_IRQ_ENABLE_SET

Application Registers
- PCIE_MSI1_IRQ_ENABLE_SET Register (Offset = 118h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.31 PCIE_MSI1_IRQ_ENABLE_CLR Register (Offset = 11Ch) [reset = 0h]

The MSI 1 Interrupt Enable Clear Register (PCIE_MSI1_IRQ_ENABLE_CLR) is shown in Figure 11-1151 and described in Table 11-2726.

Table 11-2725. PCIE_MSI1_IRQ_ENABLE_CLR Instances

Instance	Physical Address
PCIE	2180 011Ch

Figure 11-1151. PCIE_MSI1_IRQ_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI1_IRQ_EN_CLR			
R-0h				R/W1C-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2726. PCIE_MSI1_IRQ_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI1_IRQ_EN_CLR	R/W1C	0h	Each bit, when written to, disables the MSI interrupt (25, 17, 9, 1) associated with the bit. • bit 3 = MSI vector 25 disable • bit 2 = MSI vector 17 disable • bit 1 = MSI vector 9 disable • bit 0 = MSI vector 1 disable

Table 11-2727. Register Call Summary for PCIE_MSI1_IRQ_ENABLE_CLR

Application Registers
- PCIE_MSI1_IRQ_ENABLE_CLR Register (Offset = 11Ch) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.32 PCIE_MSI2_IRQ_STATUS_RAW Register (Offset = 120h) [reset = 0h]

The MSI 2 Interrupt Raw Status Register (MSI2_RAW_STATUS) is shown in Figure 11-1152 and described in Table 11-2729.

Table 11-2728. PCIE_MSI2_IRQ_STATUS_RAW Instances

Instance	Physical Address
PCIE	2180 0120h

Figure 11-1152. PCIE_MSI2_IRQ_STATUS_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI2_RAW_STATUS			
R-0h				R/W1S-0h			

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2729. PCIE_MSI2_IRQ_STATUS_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI2_RAW_STATUS	R/W1S	0h	Each bit indicates raw status of MSI vectors (26, 18, 10, 2) associated with the bit. Typically, writes to this register are only done for debug purposes. • bit 3 = MSI vector 26 status • bit 2 = MSI vector 18 status • bit 1 = MSI vector 10 status • bit 0 = MSI vector 2 status

Table 11-2730. Register Call Summary for PCIE_MSI2_IRQ_STATUS_RAW

Application Registers
• Register Summary: [0]

11.14.5.3.33 PCIE_MSI2_IRQ_STATUS Register (Offset = 124h) [reset = 0h]

The MSI 2 Interrupt Enabled Status Register (PCIE_MSI2_IRQ_STATUS) is shown in Figure 11-1153 and described in Table 11-2732.

Table 11-2731. PCIE_MSI2_IRQ_STATUS Instances

Instance	Physical Address
PCIE	2180 0124h

Figure 11-1153. PCIE_MSI2_IRQ_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI2_IRQ_STATUS			
R-0h				R/W1C-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-2732. PCIE_MSI2_IRQ_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI2_IRQ_STATUS	R/W1C	0h	Each bit indicates status of MSI vector (26, 18, 10, 2) associated with the bit. Each of the bits can be written with one to clear the respective interrupt status bit. • bit 3 = MSI vector 26 status • bit 2 = MSI vector 18 status • bit 1 = MSI vector 10 status • bit 0 = MSI vector 2 status

Table 11-2733. Register Call Summary for PCIE_MSI2_IRQ_STATUS

Application Registers
- PCIE_MSI2_IRQ_STATUS Register (Offset = 124h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.34 PCIE_MSI2_IRQ_ENABLE_SET Register (Offset = 128h) [reset = 0h]

The MSI 2 Interrupt Enable Set Register (PCIE_MSI2_IRQ_ENABLE_SET) is shown in Figure 11-1154 and described in Table 11-2735.

Table 11-2734. PCIE_MSI2_IRQ_ENABLE_SET Instances

Instance	Physical Address
PCIE	2180 0128h

Figure 11-1154. PCIE_MSI2_IRQ_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI2_IRQ_EN_SET			
R-0h				R/W1S-0h			

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2735. PCIE_MSI2_IRQ_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI2_IRQ_EN_SET	R/W1S	0h	Each bit, when written to, enables the MSI interrupt (26, 18, 10, 2) associated with the bit. • bit 3 = MSI vector 26 enable • bit 2 = MSI vector 18 enable • bit 1 = MSI vector 10 enable • bit 0 = MSI vector 2 enable

Table 11-2736. Register Call Summary for PCIE_MSI2_IRQ_ENABLE_SET

Application Registers
- PCIE_MSI2_IRQ_ENABLE_SET Register (Offset = 128h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.35 PCIE_MSI2_IRQ_ENABLE_CLR Register (Offset = 12Ch) [reset = 0h]

The MSI 2 Interrupt Enable Clear Register (PCIE_MSI2_IRQ_ENABLE_CLR) is shown in Figure 11-1155 and described in Table 11-2738.

Table 11-2737. PCIE_MSI2_IRQ_ENABLE_CLR Instances

Instance	Physical Address
PCIE	2180 012Ch

Figure 11-1155. PCIE_MSI2_IRQ_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI2_IRQ_EN_CLR			
R-0h				R/W1C-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2738. PCIE_MSI2_IRQ_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI2_IRQ_EN_CLR	R/W1C	0h	Each bit, when written to, disables the MSI interrupt (26, 18, 10, 2) associated with the bit. • bit 3 = MSI vector 26 disable • bit 2 = MSI vector 18 disable • bit 1 = MSI vector 10 disable • bit 0 = MSI vector 2 disable

Table 11-2739. Register Call Summary for PCIE_MSI2_IRQ_ENABLE_CLR

Application Registers
- PCIE_MSI2_IRQ_ENABLE_CLR Register (Offset = 12Ch) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.36 PCIE_MSI3_IRQ_STATUS_RAW Register (Offset = 130h) [reset = 0h]

The MSI 3 Interrupt Raw Status Register (MSI3_RAW_STATUS) is shown in Figure 11-1156 and described in Table 11-2741.

Table 11-2740. PCIE_MSI3_IRQ_STATUS_RAW Instances

Instance	Physical Address
PCIE	2180 0130h

Figure 11-1156. PCIE_MSI3_IRQ_STATUS_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI3_RAW_STATUS			
R-0h				R/W1S-0h			

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2741. PCIE_MSI3_IRQ_STATUS_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI3_RAW_STATUS	R/W1S	0h	Each bit indicates raw status of MSI vectors (27, 19, 11, 3) associated with the bit. Typically, writes to this register are only done for debug purposes. • bit 3 = MSI vector 27 status • bit 2 = MSI vector 19 status • bit 1 = MSI vector 11 status • bit 0 = MSI vector 3 status

Table 11-2742. Register Call Summary for PCIE_MSI3_IRQ_STATUS_RAW

Application Registers
• Register Summary: [0]

11.14.5.3.37 PCIE_MSI3_IRQ_STATUS Register (Offset = 134h) [reset = 0h]

The MSI 3 Interrupt Enabled Status Register (PCIE_MSI3_IRQ_STATUS) is shown in Figure 11-1157 and described in Table 11-2744.

Table 11-2743. PCIE_MSI3_IRQ_STATUS Instances

Instance	Physical Address
PCIE	2180 0134h

Figure 11-1157. PCIE_MSI3_IRQ_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI3_IRQ_STATUS			
R-0h				R/W1C-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-2744. PCIE_MSI3_IRQ_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI3_IRQ_STATUS	R/W1C	0h	Each bit indicates status of MSI vector (27, 19, 11, 3) associated with the bit. Each of the bits can be written with one to clear the respective interrupt status bit. • bit 3 = MSI vector 27 status • bit 2 = MSI vector 19 status • bit 1 = MSI vector 11 status • bit 0 = MSI vector 3 status

Table 11-2745. Register Call Summary for PCIE_MSI3_IRQ_STATUS

Application Registers
- PCIE_MSI3_IRQ_STATUS Register (Offset = 134h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.38 PCIE_MSI3_IRQ_ENABLE_SET Register (Offset = 138h) [reset = 0h]

The MSI 3 Interrupt Enable Set Register (PCIE_MSI3_IRQ_ENABLE_SET) is shown in Figure 11-1158 and described in Table 11-2747.

Table 11-2746. PCIE_MSI3_IRQ_ENABLE_SET Instances

Instance	Physical Address
PCIE	2180 0138h

Figure 11-1158. PCIE_MSI3_IRQ_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI3_IRQ_EN_SET			
R-0h				R/W1S-0h			

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2747. PCIE_MSI3_IRQ_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI3_IRQ_EN_SET	R/W1S	0h	Each bit, when written to, enables the MSI interrupt (27, 19, 11, 3) associated with the bit. • bit 3 = MSI vector 27 enable • bit 2 = MSI vector 19 enable • bit 1 = MSI vector 11 enable • bit 0 = MSI vector 3 enable

Table 11-2748. Register Call Summary for PCIE_MSI3_IRQ_ENABLE_SET

Application Registers
- PCIE_MSI3_IRQ_ENABLE_SET Register (Offset = 138h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.39 PCIE_MSI3_IRQ_ENABLE_CLR Register (Offset = 13Ch) [reset = 0h]

The MSI 3 Interrupt Enable Clear Register (PCIE_MSI3_IRQ_ENABLE_CLR) is shown in Figure 11-1159 and described in Table 11-2750.

Table 11-2749. PCIE_MSI3_IRQ_ENABLE_CLR Instances

Instance	Physical Address
PCIE	2180 013Ch

Figure 11-1159. PCIE_MSI3_IRQ_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI3_IRQ_EN_CLR			
R-0h				R/W1C-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2750. PCIE_MSI3_IRQ_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI3_IRQ_EN_CLR	R/W1C	0h	Each bit, when written to, disables the MSI interrupt (27, 19, 11, 3) associated with the bit. • bit 3 = MSI vector 27 disable • bit 2 = MSI vector 19 disable • bit 1 = MSI vector 11 disable • bit 0 = MSI vector 3 disable

Table 11-2751. Register Call Summary for PCIE_MSI3_IRQ_ENABLE_CLR

Application Registers
- PCIE_MSI3_IRQ_ENABLE_CLR Register (Offset = 13Ch) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.40 PCIE_MSI4_IRQ_STATUS_RAW Register (Offset = 140h) [reset = 0h]

The MSI 4 Interrupt Raw Status Register (MSI4_RAW_STATUS) is shown in Figure 11-1160 and described in Table 11-2753.

Table 11-2752. PCIE_MSI4_IRQ_STATUS_RAW Instances

Instance	Physical Address
PCIE	2180 0140h

Figure 11-1160. PCIE_MSI4_IRQ_STATUS_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI4_RAW_STATUS			
R-0h				R/W1S-0h			

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2753. PCIE_MSI4_IRQ_STATUS_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI4_RAW_STATUS	R/W1S	0h	Each bit indicates raw status of MSI vectors (28, 20, 12, 4) associated with the bit. Typically, writes to this register are only done for debug purposes. • bit 3 = MSI vector 28 status • bit 2 = MSI vector 20 status • bit 1 = MSI vector 12 status • bit 0 = MSI vector 4 status

Table 11-2754. Register Call Summary for PCIE_MSI4_IRQ_STATUS_RAW

Application Registers
• Register Summary: [0]

11.14.5.3.41 PCIE_MSI4_IRQ_STATUS Register (Offset = 144h) [reset = 0h]

The MSI 4 Interrupt Enabled Status Register (PCIE_MSI4_IRQ_STATUS) is shown in Figure 11-1161 and described in Table 11-2756.

Table 11-2755. PCIE_MSI4_IRQ_STATUS Instances

Instance	Physical Address
PCIE	2180 0144h

Figure 11-1161. PCIE_MSI4_IRQ_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI4_IRQ_STATUS			
R-0h				R/W1C-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-2756. PCIE_MSI4_IRQ_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI4_IRQ_STATUS	R/W1C	0h	Each bit indicates status of MSI vector (28, 20, 12, 4) associated with the bit. Each of the bits can be written with one to clear the respective interrupt status bit. • bit 3 = MSI vector 28 status • bit 2 = MSI vector 20 status • bit 1 = MSI vector 12 status • bit 0 = MSI vector 4 status

Table 11-2757. Register Call Summary for PCIE_MSI4_IRQ_STATUS

Application Registers
- PCIE_MSI4_IRQ_STATUS Register (Offset = 144h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.42 PCIE_MSI4_IRQ_ENABLE_SET Register (Offset = 148h) [reset = 0h]

The MSI 4 Interrupt Enable Set Register (PCIE_MSI4_IRQ_ENABLE_SET) is shown in Figure 11-1162 and described in Table 11-2759.

Table 11-2758. PCIE_MSI4_IRQ_ENABLE_SET Instances

Instance	Physical Address
PCIE	2180 0148h

Figure 11-1162. PCIE_MSI4_IRQ_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI4_IRQ_EN_SET			
R-0h				R/W1S-0h			

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2759. PCIE_MSI4_IRQ_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI4_IRQ_EN_SET	R/W1S	0h	Each bit, when written to, enables the MSI interrupt (28, 20, 12, 4) associated with the bit. • bit 3 = MSI vector 28 enable • bit 2 = MSI vector 20 enable • bit 1 = MSI vector 12 enable • bit 0 = MSI vector 4 enable

Table 11-2760. Register Call Summary for PCIE_MSI4_IRQ_ENABLE_SET

Application Registers
- PCIE_MSI4_IRQ_ENABLE_SET Register (Offset = 148h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.43 PCIE_MSI4_IRQ_ENABLE_CLR Register (Offset = 14Ch) [reset = 0h]

The MSI 4 Interrupt Enable Clear Register (PCIE_MSI4_IRQ_ENABLE_CLR) is shown in Figure 11-1163 and described in Table 11-2762.

Table 11-2761. PCIE_MSI4_IRQ_ENABLE_CLR Instances

Instance	Physical Address
PCIE	2180 014Ch

Figure 11-1163. PCIE_MSI4_IRQ_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI4_IRQ_EN_CLR			
R-0h				R/W1C-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2762. PCIE_MSI4_IRQ_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI4_IRQ_EN_CLR	R/W1C	0h	Each bit, when written to, disables the MSI interrupt (28, 20, 12, 4) associated with the bit. • bit 3 = MSI vector 28 disable • bit 2 = MSI vector 20 disable • bit 1 = MSI vector 12 disable • bit 0 = MSI vector 4 disable

Table 11-2763. Register Call Summary for PCIE_MSI4_IRQ_ENABLE_CLR

Application Registers
- PCIE_MSI4_IRQ_ENABLE_CLR Register (Offset = 14Ch) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.44 PCIE_MSI5_IRQ_STATUS_RAW Register (Offset = 150h) [reset = 0h]

The MSI 5 Interrupt Raw Status Register (MSI5_RAW_STATUS) is shown in Figure 11-1164 and described in Table 11-2765.

Table 11-2764. PCIE_MSI5_IRQ_STATUS_RAW Instances

Instance	Physical Address
PCIE	2180 0150h

Figure 11-1164. PCIE_MSI5_IRQ_STATUS_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI5_RAW_STATUS			
R-0h				R/W1S-0h			

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2765. PCIE_MSI5_IRQ_STATUS_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI5_RAW_STATUS	R/W1S	0h	Each bit indicates raw status of MSI vectors (29, 21, 13, 5) associated with the bit. Typically, writes to this register are only done for debug purposes. • bit 3 = MSI vector 29 status • bit 2 = MSI vector 21 status • bit 1 = MSI vector 13 status • bit 0 = MSI vector 5 status

Table 11-2766. Register Call Summary for PCIE_MSI5_IRQ_STATUS_RAW

Application Registers
• Register Summary: [0]

11.14.5.3.45 PCIE_MSI5_IRQ_STATUS Register (Offset = 154h) [reset = 0h]

The MSI 5 Interrupt Enabled Status Register (PCIE_MSI5_IRQ_STATUS) is shown in Figure 11-1165 and described in Table 11-2768.

Table 11-2767. PCIE_MSI5_IRQ_STATUS Instances

Instance	Physical Address
PCIE	2180 0154h

Figure 11-1165. PCIE_MSI5_IRQ_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI5_IRQ_STATUS			
R-0h				R/W1C-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-2768. PCIE_MSI5_IRQ_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI5_IRQ_STATUS	R/W1C	0h	Each bit indicates status of MSI vector (29, 21, 13, 5) associated with the bit. Each of the bits can be written with one to clear the respective interrupt status bit. • bit 3 = MSI vector 29 status • bit 2 = MSI vector 21 status • bit 1 = MSI vector 13 status • bit 0 = MSI vector 5 status

Table 11-2769. Register Call Summary for PCIE_MSI5_IRQ_STATUS

Application Registers
- PCIE_MSI5_IRQ_STATUS Register (Offset = 154h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.46 PCIE_MSI5_IRQ_ENABLE_SET Register (Offset = 158h) [reset = 0h]

The MSI 5 Interrupt Enable Set Register (PCIE_MSI5_IRQ_ENABLE_SET) is shown in Figure 11-1166 and described in Table 11-2771.

Table 11-2770. PCIE_MSI5_IRQ_ENABLE_SET Instances

Instance	Physical Address
PCIE	2180 0158h

Figure 11-1166. PCIE_MSI5_IRQ_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI5_IRQ_EN_SET			
R-0h				R/W1S-0h			

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2771. PCIE_MSI5_IRQ_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI5_IRQ_EN_SET	R/W1S	0h	Each bit, when written to, enables the MSI interrupt (29, 21, 13, 5) associated with the bit. • bit 3 = MSI vector 29 enable • bit 2 = MSI vector 21 enable • bit 1 = MSI vector 13 enable • bit 0 = MSI vector 5 enable

Table 11-2772. Register Call Summary for PCIE_MSI5_IRQ_ENABLE_SET

Application Registers
- PCIE_MSI5_IRQ_ENABLE_SET Register (Offset = 158h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.47 PCIE_MSI5_IRQ_ENABLE_CLR Register (Offset = 15Ch) [reset = 0h]

The MSI 5 Interrupt Enable Clear Register (PCIE_MSI5_IRQ_ENABLE_CLR) is shown in Figure 11-1167 and described in Table 11-2774.

Table 11-2773. PCIE_MSI5_IRQ_ENABLE_CLR Instances

Instance	Physical Address
PCIE	2180 015Ch

Figure 11-1167. PCIE_MSI5_IRQ_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI5_IRQ_EN_CLR			
R-0h				R/W1C-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2774. PCIE_MSI5_IRQ_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI5_IRQ_EN_CLR	R/W1C	0h	Each bit, when written to, disables the MSI interrupt (29, 21, 13, 5) associated with the bit. • bit 3 = MSI vector 29 disable • bit 2 = MSI vector 21 disable • bit 1 = MSI vector 13 disable • bit 0 = MSI vector 5 disable

Table 11-2775. Register Call Summary for PCIE_MSI5_IRQ_ENABLE_CLR

Application Registers
- PCIE_MSI5_IRQ_ENABLE_CLR Register (Offset = 15Ch) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.48 PCIE_MSI6_IRQ_STATUS_RAW Register (Offset = 160h) [reset = 0h]

The MSI 6 Interrupt Raw Status Register (MSI6_RAW_STATUS) is shown in Figure 11-1168 and described in Table 11-2777.

Table 11-2776. PCIE_MSI6_IRQ_STATUS_RAW Instances

Instance	Physical Address
PCIE	2180 0160h

Figure 11-1168. PCIE_MSI6_IRQ_STATUS_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI6_RAW_STATUS			
R-0h				R/W1S-0h			

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2777. PCIE_MSI6_IRQ_STATUS_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI6_RAW_STATUS	R/W1S	0h	Each bit indicates raw status of MSI vectors (30, 22, 14, 6) associated with the bit. Typically, writes to this register are only done for debug purposes. • bit 3 = MSI vector 30 status • bit 2 = MSI vector 22 status • bit 1 = MSI vector 14 status • bit 0 = MSI vector 6 status

Table 11-2778. Register Call Summary for PCIE_MSI6_IRQ_STATUS_RAW

Application Registers
• Register Summary: [0]

11.14.5.3.49 PCIE_MSI6_IRQ_STATUS Register (Offset = 164h) [reset = 0h]

The MSI 6 Interrupt Enabled Status Register (PCIE_MSI6_IRQ_STATUS) is shown in Figure 11-1169 and described in Table 11-2780.

Table 11-2779. PCIE_MSI6_IRQ_STATUS Instances

Instance	Physical Address
PCIE	2180 0164h

Figure 11-1169. PCIE_MSI6_IRQ_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI6_IRQ_STATUS			
R-0h				R/W1C-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-2780. PCIE_MSI6_IRQ_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI6_IRQ_STATUS	R/W1C	0h	Each bit indicates status of MSI vector (30, 22, 14, 6) associated with the bit. Each of the bits can be written with one to clear the respective interrupt status bit. • bit 3 = MSI vector 30 status • bit 2 = MSI vector 22 status • bit 1 = MSI vector 14 status • bit 0 = MSI vector 6 status

Table 11-2781. Register Call Summary for PCIE_MSI6_IRQ_STATUS

Application Registers
- Register Summary: [0]
- PCIE_MSI6_IRQ_STATUS Register (Offset = 164h) [reset = 0h]: [0]

11.14.5.3.50 PCIE_MSI6_IRQ_ENABLE_SET Register (Offset = 168h) [reset = 0h]

The MSI 6 Interrupt Enable Set Register (PCIE_MSI6_IRQ_ENABLE_SET) is shown in Figure 11-1170 and described in Table 11-2783.

Table 11-2782. PCIE_MSI6_IRQ_ENABLE_SET Instances

Instance	Physical Address
PCIE	2180 0168h

Figure 11-1170. PCIE_MSI6_IRQ_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI6_IRQ_EN_SET			
R-0h				R/W1S-0h			

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2783. PCIE_MSI6_IRQ_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI6_IRQ_EN_SET	R/W1S	0h	Each bit, when written to, enables the MSI interrupt (30, 22, 14, 6) associated with the bit. • bit 3 = MSI vector 30 enable • bit 2 = MSI vector 22 enable • bit 1 = MSI vector 14 enable • bit 0 = MSI vector 6enable

Table 11-2784. Register Call Summary for PCIE_MSI6_IRQ_ENABLE_SET

Application Registers
- PCIE_MSI6_IRQ_ENABLE_SET Register (Offset = 168h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.51 PCIE_MSI6_IRQ_ENABLE_CLR Register (Offset = 16Ch) [reset = 0h]

The MSI 6 Interrupt Enable Clear Register (PCIE_MSI6_IRQ_ENABLE_CLR) is shown in Figure 11-1171 and described in Table 11-2786.

Table 11-2785. PCIE_MSI6_IRQ_ENABLE_CLR Instances

Instance	Physical Address
PCIE	2180 016Ch

Figure 11-1171. PCIE_MSI6_IRQ_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI6_IRQ_EN_CLR			
R-0h				R/W1C-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2786. PCIE_MSI6_IRQ_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI6_IRQ_EN_CLR	R/W1C	0h	Each bit, when written to, disables the MSI interrupt (30, 22, 14, 6) associated with the bit. • bit 3 = MSI vector 30 disable • bit 2 = MSI vector 22 disable • bit 1 = MSI vector 14 disable • bit 0 = MSI vector 6 disable

Table 11-2787. Register Call Summary for PCIE_MSI6_IRQ_ENABLE_CLR

Application Registers
- PCIE_MSI6_IRQ_ENABLE_CLR Register (Offset = 16Ch) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.52 PCIE_MSI7_IRQ_STATUS_RAW Register (Offset = 170h) [reset = 0h]

The MSI 7 Interrupt Raw Status Register (MSI7_RAW_STATUS) is shown in Figure 11-1172 and described in Table 11-2789.

Table 11-2788. PCIE_MSI7_IRQ_STATUS_RAW Instances

Instance	Physical Address
PCIE	2180 0170h

Figure 11-1172. PCIE_MSI7_IRQ_STATUS_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI7_RAW_STATUS			
R-0h				R/W1S-0h			

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2789. PCIE_MSI7_IRQ_STATUS_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI7_RAW_STATUS	R/W1S	0h	Each bit indicates raw status of MSI vectors (31, 23, 15, 7) associated with the bit. Typically, writes to this register are only done for debug purposes. • bit 3 = MSI vector 31 status • bit 2 = MSI vector 23 status • bit 1 = MSI vector 15 status • bit 0 = MSI vector 7status

Table 11-2790. Register Call Summary for PCIE_MSI7_IRQ_STATUS_RAW

Application Registers
• Register Summary: [0]

11.14.5.3.53 PCIE_MSI7_IRQ_STATUS Register (Offset = 174h) [reset = 0h]

The MSI 7 Interrupt Enabled Status Register (PCIE_MSI7_IRQ_STATUS) is shown in Figure 11-1173 and described in Table 11-2792.

Table 11-2791. PCIE_MSI7_IRQ_STATUS Instances

Instance	Physical Address
PCIE	2180 0174h

Figure 11-1173. PCIE_MSI7_IRQ_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI7_IRQ_STATUS			
R-0h				R/W1C-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-2792. PCIE_MSI7_IRQ_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI7_IRQ_STATUS	R/W1C	0h	Each bit indicates status of MSI vector (31, 23, 15, 7) associated with the bit. Each of the bits can be written with one to clear the respective interrupt status bit. • bit 3 = MSI vector 31 status • bit 2 = MSI vector 23 status • bit 1 = MSI vector 15 status • bit 0 = MSI vector 7 status

Table 11-2793. Register Call Summary for PCIE_MSI7_IRQ_STATUS

Application Registers • PCIE_MSI7_IRQ_STATUS Register (Offset = 174h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • MSI Interrupts Reception in RC Mode: [0][1] • Interrupt Reception in EP Mode: [0][1]

11.14.5.3.54 PCIE_MSI7_IRQ_ENABLE_SET Register (Offset = 178h) [reset = 0h]

The MSI 7 Interrupt Enable Set Register (PCIE_MSI7_IRQ_ENABLE_SET) is shown in Figure 11-1174 and described in Table 11-2795.

Table 11-2794. PCIE_MSI7_IRQ_ENABLE_SET Instances

Instance	Physical Address
PCIE	2180 0178h

Figure 11-1174. PCIE_MSI7_IRQ_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI7_IRQ_EN_SET			
R-0h				R/W1S-0h			

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2795. PCIE_MSI7_IRQ_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI7_IRQ_EN_SET	R/W1S	0h	Each bit, when written to, enables the MSI interrupt (31, 23, 15, 7) associated with the bit. • bit 3 = MSI vector 31 enable • bit 2 = MSI vector 23 enable • bit 1 = MSI vector 15 enable • bit 0 = MSI vector 7 enable

Table 11-2796. Register Call Summary for PCIE_MSI7_IRQ_ENABLE_SET

Application Registers • PCIE_MSI7_IRQ_ENABLE_SET Register (Offset = 178h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • MSI Interrupts Reception in RC Mode: [0]

11.14.5.3.55 PCIE_MSI7_IRQ_ENABLE_CLR Register (Offset = 17Ch) [reset = 0h]

The MSI 7 Interrupt Enable Clear Register (PCIE_MSI7_IRQ_ENABLE_CLR) is shown in Figure 11-1175 and described in Table 11-2798.

Table 11-2797. PCIE_MSI7_IRQ_ENABLE_CLR Instances

Instance	Physical Address
PCIE	2180 017Ch

Figure 11-1175. PCIE_MSI7_IRQ_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MSI7_IRQ_EN_CLR			
R-0h				R/W1C-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2798. PCIE_MSI7_IRQ_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3-0	MSI7_IRQ_EN_CLR	R/W1C	0h	Each bit, when written to, disables the MSI interrupt (31, 23, 15, 7) associated with the bit. • bit 3 = MSI vector 31 disable • bit 2 = MSI vector 23 disable • bit 1 = MSI vector 15 disable • bit 0 = MSI vector 7 disable

Table 11-2799. Register Call Summary for PCIE_MSI7_IRQ_ENABLE_CLR

Application Registers • Register Summary: [0] • PCIE_MSI7_IRQ_ENABLE_CLR Register (Offset = 17Ch) [reset = 0h]: [0]
PCIe SS Functional Description • MSI Interrupts Reception in RC Mode: [0] • Interrupt Reception in EP Mode: [0]

11.14.5.3.56 PCIE_LEGACY_A_IRQ_STATUS_RAW Register (Offset = 180h) [reset = 0h]

The Legacy A Raw Interrupt Status Register (PCIE_LEGACY_A_IRQ_STATUS_RAW) is shown in Figure 11-1176 and described in Table 11-2801.

Table 11-2800. PCIE_LEGACY_A_IRQ_STATUS_RAW Instances

Instance	Physical Address
PCIE	2180 0180h

Figure 11-1176. PCIE_LEGACY_A_IRQ_STATUS_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTA
R-0h							R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2801. PCIE_LEGACY_A_IRQ_STATUS_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INTA	R/W1S	0h	Legacy Interrupt A raw status. RC mode only. • 0 = interrupt is not active • 1 = interrupt is active

Table 11-2802. Register Call Summary for PCIE_LEGACY_A_IRQ_STATUS_RAW

Application Registers
- PCIE_LEGACY_A_IRQ_STATUS_RAW Register (Offset = 180h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.57 PCIE_LEGACY_A_IRQ_STATUS Register (Offset = 184h) [reset = 0h]

The Legacy A Interrupt Enabled Status Register (PCIE_LEGACY_A_IRQ_STATUS) is shown in Figure 11-1177 and described in Table 11-2804.

Table 11-2803. PCIE_LEGACY_A_IRQ_STATUS Instances

Instance	Physical Address
PCIE	2180 0184h

Figure 11-1177. PCIE_LEGACY_A_IRQ_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTA
R-0h							R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2804. PCIE_LEGACY_A_IRQ_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INTA	R/W1C	0h	Legacy Interrupt A status. Set when interrupt is active. Write one to clear the interrupt event. RC mode only.

Table 11-2805. Register Call Summary for PCIE_LEGACY_A_IRQ_STATUS

Application Registers
- PCIE_LEGACY_A_IRQ_STATUS Register (Offset = 184h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.58 PCIE_LEGACY_A_IRQ_ENABLE_SET Register (Offset = 188h) [reset = 0h]

The Legacy A Interrupt Enabled Set Register (PCIE_LEGACY_A_IRQ_ENABLE_SET) is shown in Figure 11-1178 and described in Table 11-2807.

Table 11-2806. PCIE_LEGACY_A_IRQ_ENABLE_SET Instances

Instance	Physical Address
PCIE	2180 0188h

Figure 11-1178. PCIE_LEGACY_A_IRQ_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTA
R-0h							R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2807. PCIE_LEGACY_A_IRQ_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INTA	R/W1S	0h	Legacy Interrupt A enable. Set to enable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled

Table 11-2808. Register Call Summary for PCIE_LEGACY_A_IRQ_ENABLE_SET

Application Registers • PCIE_LEGACY_A_IRQ_ENABLE_SET Register (Offset = 188h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • Legacy Interrupt Generation in EP Mode: [0]

11.14.5.3.59 PCIE_LEGACY_A_IRQ_ENABLE_CLR Register (Offset = 18Ch) [reset = 0h]

The Legacy A Interrupt Enabled Clear Register (PCIE_LEGACY_A_IRQ_ENABLE_CLR) is shown in Figure 11-1179 and described in Table 11-2810.

Table 11-2809. PCIE_LEGACY_A_IRQ_ENABLE_CLR Instances

Instance	Physical Address
PCIE	2180 018Ch

Figure 11-1179. PCIE_LEGACY_A_IRQ_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTA
R-0h							R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2810. PCIE_LEGACY_A_IRQ_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INTA	R/W1C	0h	Legacy Interrupt A disable. Set to disable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled

Table 11-2811. Register Call Summary for PCIE_LEGACY_A_IRQ_ENABLE_CLR

Application Registers
- PCIE_LEGACY_A_IRQ_ENABLE_CLR Register (Offset = 18Ch) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.60 PCIE_LEGACY_B_IRQ_STATUS_RAW Register (Offset = 190h) [reset = 0h]

The Legacy B Raw Interrupt Status Register (PCIE_LEGACY_B_IRQ_STATUS_RAW) is shown in Figure 11-1180 and described in Table 11-2813.

Table 11-2812. PCIE_LEGACY_B_IRQ_STATUS_RAW Instances

Instance	Physical Address
PCIE	2180 0190h

Figure 11-1180. PCIE_LEGACY_B_IRQ_STATUS_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTB
R-0h							R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2813. PCIE_LEGACY_B_IRQ_STATUS_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INTB	R/W1S	0h	Legacy Interrupt B raw status. RC mode only. • 0 = interrupt is not active • 1 = interrupt is active

Table 11-2814. Register Call Summary for PCIE_LEGACY_B_IRQ_STATUS_RAW

Application Registers
- PCIE_LEGACY_B_IRQ_STATUS_RAW Register (Offset = 190h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.61 PCIE_LEGACY_B_IRQ_STATUS Register (Offset = 194h) [reset = 0h]

The Legacy B Interrupt Enabled Status Register (PCIE_LEGACY_B_IRQ_STATUS) is shown in Figure 11-1181 and described in Table 11-2816.

Table 11-2815. PCIE_LEGACY_B_IRQ_STATUS Instances

Instance	Physical Address
PCIE	2180 0194h

Figure 11-1181. PCIE_LEGACY_B_IRQ_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTB
R-0h							R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2816. PCIE_LEGACY_B_IRQ_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INTB	R/W1C	0h	Legacy Interrupt B status. Set when interrupt is active. Write one to clear the interrupt event. RC mode only.

Table 11-2817. Register Call Summary for PCIE_LEGACY_B_IRQ_STATUS

Application Registers
- PCIE_LEGACY_B_IRQ_STATUS Register (Offset = 194h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.62 PCIE_LEGACY_B_IRQ_ENABLE_SET Register (Offset = 198h) [reset = 0h]

The Legacy B Interrupt Enabled Set Register (PCIE_LEGACY_B_IRQ_ENABLE_SET) is shown in Figure 11-1182 and described in Table 11-2819.

Table 11-2818. PCIE_LEGACY_B_IRQ_ENABLE_SET Instances

Instance	Physical Address
PCIE	2180 0198h

Figure 11-1182. PCIE_LEGACY_B_IRQ_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTB
R-0h							R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2819. PCIE_LEGACY_B_IRQ_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INTB	R/W1S	0h	Legacy Interrupt B enable. Set to enable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled

Table 11-2820. Register Call Summary for PCIE_LEGACY_B_IRQ_ENABLE_SET

Application Registers • PCIE_LEGACY_B_IRQ_ENABLE_SET Register (Offset = 198h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • Legacy Interrupt Generation in EP Mode: [0]

11.14.5.3.63 PCIE_LEGACY_B_IRQ_ENABLE_CLR Register (Offset = 19Ch) [reset = 0h]

The Legacy B Interrupt Enabled Clear Register (PCIE_LEGACY_B_IRQ_ENABLE_CLR) is shown in Figure 11-1183 and described in Table 11-2822.

Table 11-2821. PCIE_LEGACY_B_IRQ_ENABLE_CLR Instances

Instance	Physical Address
PCIE	2180 019Ch

Figure 11-1183. PCIE_LEGACY_B_IRQ_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTB
R-0h							R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2822. PCIE_LEGACY_B_IRQ_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INTB	R/W1C	0h	Legacy Interrupt B disable. Set to disable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled

Table 11-2823. Register Call Summary for PCIE_LEGACY_B_IRQ_ENABLE_CLR

Application Registers • PCIE_LEGACY_B_IRQ_ENABLE_CLR Register (Offset = 19Ch) [reset = 0h]: [0] • Register Summary: [0]

11.14.5.3.64 PCIE_LEGACY_C_IRQ_STATUS_RAW Register (Offset = 1A0h) [reset = 0h]

The Legacy C Raw Interrupt Status Register (PCIE_LEGACY_C_IRQ_STATUS_RAW) is shown in Figure 11-1184 and described in Table 11-2825. The offset is 1A0h.

Table 11-2824. PCIE_LEGACY_C_IRQ_STATUS_RAW Instances

Instance	Physical Address
PCIE	2180 01A0h

Figure 11-1184. PCIE_LEGACY_C_IRQ_STATUS_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTC
R-0h							R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2825. PCIE_LEGACY_C_IRQ_STATUS_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INTC	R/W1S	0h	Legacy Interrupt C raw status. RC mode only. • 0 = interrupt is not active • 1 = interrupt is active

Table 11-2826. Register Call Summary for PCIE_LEGACY_C_IRQ_STATUS_RAW

Application Registers
- PCIE_LEGACY_C_IRQ_STATUS_RAW Register (Offset = 1A0h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.65 PCIE_LEGACY_C_IRQ_STATUS Register (Offset = 1A4h) [reset = 0h]

The Legacy C Interrupt Enabled Status Register (PCIE_LEGACY_C_IRQ_STATUS) is shown in Figure 11-1185 and described in Table 11-2828. The offset is 1A4h.

Table 11-2827. PCIE_LEGACY_C_IRQ_STATUS Instances

Instance	Physical Address
PCIE	2180 01A4h

Figure 11-1185. PCIE_LEGACY_C_IRQ_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTC
R-0h							R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2828. PCIE_LEGACY_C_IRQ_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INTC	R/W1C	0h	Legacy Interrupt C status. Set when interrupt is active. Write one to clear the interrupt event. RC mode only.

Table 11-2829. Register Call Summary for PCIE_LEGACY_C_IRQ_STATUS

Application Registers
- PCIE_LEGACY_C_IRQ_STATUS Register (Offset = 1A4h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.66 PCIE_LEGACY_C_IRQ_ENABLE_SET Register (Offset = 1A8h) [reset = 0h]

The Legacy C Interrupt Enabled Set Register (PCIE_LEGACY_C_IRQ_ENABLE_SET) is shown in Figure 11-1186 and described in Table 11-2831. The offset is 1A8h.

Table 11-2830. PCIE_LEGACY_C_IRQ_ENABLE_SET Instances

Instance	Physical Address
PCIE	2180 01A8h

Figure 11-1186. PCIE_LEGACY_C_IRQ_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTC
R-0h							R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2831. PCIE_LEGACY_C_IRQ_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INTC	R/W1S	0h	Legacy Interrupt C enable. Set to enable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled

Table 11-2832. Register Call Summary for PCIE_LEGACY_C_IRQ_ENABLE_SET

Application Registers • Register Summary: [0] • PCIE_LEGACY_C_IRQ_ENABLE_SET Register (Offset = 1A8h) [reset = 0h]: [0]
PCIe SS Functional Description • Legacy Interrupt Generation in EP Mode: [0]

11.14.5.3.67 PCIE_LEGACY_C_IRQ_ENABLE_CLR Register (Offset = 1ACh) [reset = 0h]

The Legacy C Interrupt Enabled Clear Register (PCIE_LEGACY_C_IRQ_ENABLE_CLR) is shown in Figure 11-1187 and described in Table 11-2834.

Table 11-2833. PCIE_LEGACY_C_IRQ_ENABLE_CLR Instances

Instance	Physical Address
PCIE	2180 01ACh

Figure 11-1187. PCIE_LEGACY_C_IRQ_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTC
R-0h							R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2834. PCIE_LEGACY_C_IRQ_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INTC	R/W1C	0h	Legacy Interrupt C disable. Set to disable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled

Table 11-2835. Register Call Summary for PCIE_LEGACY_C_IRQ_ENABLE_CLR

Application Registers
- PCIE_LEGACY_C_IRQ_ENABLE_CLR Register (Offset = 1ACh) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.68 PCIE_LEGACY_D_IRQ_STATUS_RAW Register (Offset = 1B0h) [reset = 0h]

The Legacy D Raw Interrupt Status Register (PCIE_LEGACY_D_IRQ_STATUS_RAW) is shown in Figure 11-1188 and described in Table 11-2837. The offset is 1B0h.

Table 11-2836. PCIE_LEGACY_D_IRQ_STATUS_RAW Instances

Instance	Physical Address
PCIE	2180 01B0h

Figure 11-1188. PCIE_LEGACY_D_IRQ_STATUS_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTD
R-0h							R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2837. PCIE_LEGACY_D_IRQ_STATUS_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INTD	R/W1S	0h	Legacy Interrupt D raw status. RC mode only. • 0 = interrupt is not active • 1 = interrupt is active

Table 11-2838. Register Call Summary for PCIE_LEGACY_D_IRQ_STATUS_RAW

Application Registers
- Register Summary: [0]
- PCIE_LEGACY_D_IRQ_STATUS_RAW Register (Offset = 1B0h) [reset = 0h]: [0]

11.14.5.3.69 PCIE_LEGACY_D_IRQ_STATUS Register (Offset = 1B4h) [reset = 0h]

The Legacy D Interrupt Enabled Status Register (PCIE_LEGACY_D_IRQ_STATUS) is shown in Figure 11-1189 and described in Table 11-2840. The offset is 1B4h.

Table 11-2839. PCIE_LEGACY_D_IRQ_STATUS Instances

Instance	Physical Address
PCIE	2180 01B4h

Figure 11-1189. PCIE_LEGACY_D_IRQ_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTD
R-0h							R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2840. PCIE_LEGACY_D_IRQ_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INTD	R/W1C	0h	Legacy Interrupt D status. Set when interrupt is active. Write one to clear the interrupt event. RC mode only.

Table 11-2841. Register Call Summary for PCIE_LEGACY_D_IRQ_STATUS

Application Registers
- PCIE_LEGACY_D_IRQ_STATUS Register (Offset = 1B4h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.70 PCIE_LEGACY_D_IRQ_ENABLE_SET Register (Offset = 1B8h) [reset = 0h]

The Legacy D Interrupt Enabled Set Register (PCIE_LEGACY_D_IRQ_ENABLE_SET) is shown in Figure 11-1190 and described in Table 11-2843. The offset is 1B8h.

Table 11-2842. PCIE_LEGACY_D_IRQ_ENABLE_SET Instances

Instance	Physical Address
PCIE	2180 01B8h

Figure 11-1190. PCIE_LEGACY_D_IRQ_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTD
R-0h							R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2843. PCIE_LEGACY_D_IRQ_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INTD	R/W1S	0h	Legacy Interrupt D enable. Set to enable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled

Table 11-2844. Register Call Summary for PCIE_LEGACY_D_IRQ_ENABLE_SET

Application Registers • PCIE_LEGACY_D_IRQ_ENABLE_SET Register (Offset = 1B8h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • Legacy Interrupt Generation in EP Mode: [0]

11.14.5.3.71 PCIE_LEGACY_D_IRQ_ENABLE_CLR Register (Offset = 1BCh) [reset = 0h]

The Legacy D Interrupt Enabled Clear Register (PCIE_LEGACY_D_IRQ_ENABLE_CLR) is shown in Figure 11-1191 and described in Table 11-2846.

Table 11-2845. PCIE_LEGACY_D_IRQ_ENABLE_CLR Instances

Instance	Physical Address
PCIE	2180 01BCh

Figure 11-1191. PCIE_LEGACY_D_IRQ_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTD
R-0h							R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2846. PCIE_LEGACY_D_IRQ_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	INTD	R/W1C	0h	Legacy Interrupt D disable. Set to disable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled

Table 11-2847. Register Call Summary for PCIE_LEGACY_D_IRQ_ENABLE_CLR

Application Registers
- Register Summary: [0]
- PCIE_LEGACY_D_IRQ_ENABLE_CLR Register (Offset = 1BCh) [reset = 0h]: [0]

11.14.5.3.72 PCIE_ERR_IRQ_STATUS_RAW Register (Offset = 1C0h) [reset = 0h]

The Raw ERR Interrupt Status Register (PCIE_ERR_IRQ_STATUS_RAW) is shown in Figure 11-1192 and described in Table 11-2849.The offset is 1C0h.

Table 11-2848. PCIE_ERR_IRQ_STATUS_RAW Instances

Instance	Physical Address
PCIE	2180 01C0h

Figure 11-1192. PCIE_ERR_IRQ_STATUS_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		ERR_AER	ERR_AXI	ERR_CORR	ERR_NONFATAL	ERR_FATAL	ERR_SYS
R-0h		R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; -*n* = value after reset

Table 11-2849. PCIE_ERR_IRQ_STATUS_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reads return 0 and writes have no effect.
5	ERR_AER	R/W1S	0h	ECRC error raw status.
4	ERR_AXI	R/W1S	0h	AXI tag lookup fatal error raw status.
3	ERR_CORR	R/W1S	0h	Correctable error raw status.
2	ERR_NONFATAL	R/W1S	0h	Nonfatal error raw status.
1	ERR_FATAL	R/W1S	0h	Fatal error raw status.
0	ERR_SYS	R/W1S	0h	System error (fatal, nonfatal or correctable error) raw status.

Table 11-2850. Register Call Summary for PCIE_ERR_IRQ_STATUS_RAW

Application Registers
- Register Summary: [0]
- PCIE_ERR_IRQ_STATUS_RAW Register (Offset = 1C0h) [reset = 0h]: [0]

11.14.5.3.73 PCIE_ERR_IRQ_STATUS Register (Offset = 1C4h) [reset = 0h]

The ERR Interrupt Enabled Status Register (PCIE_ERR_IRQ_STATUS) is shown in Figure 11-1193 and described in Table 11-2852. The offset is 1C4h.

Table 11-2851. PCIE_ERR_IRQ_STATUS Instances

Instance	Physical Address
PCIE	2180 01C4h

Figure 11-1193. PCIE_ERR_IRQ_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		ERR_AER	ERR_AXI	ERR_CORR	ERR_NONFATAL	ERR_FATAL	ERR_SYS
R-0h		R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2852. PCIE_ERR_IRQ_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reads return 0 and writes have no effect.
5	ERR_AER	R/W1C	0h	ECRC error status.
4	ERR_AXI	R/W1C	0h	AXI tag lookup fatal error.
3	ERR_CORR	R/W1C	0h	Correctable error status.
2	ERR_NONFATAL	R/W1C	0h	Nonfatal error status.
1	ERR_FATAL	R/W1C	0h	Fatal error status.
0	ERR_SYS	R/W1C	0h	System Error (fatal, nonfatal or correctable error).

Table 11-2853. Register Call Summary for PCIE_ERR_IRQ_STATUS

Application Registers
- PCIE_ERR_IRQ_STATUS Register (Offset = 1C4h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.74 PCIE_ERR_IRQ_ENABLE_SET Register (Offset = 1C8h) [reset = 0h]

The ERR Interrupt Enable Set Register (PCIE_ERR_IRQ_ENABLE_SET) is shown in Figure 11-1194 and described in Table 11-2855. The offset is 1C8h.

Table 11-2854. PCIE_ERR_IRQ_ENABLE_SET Instances

Instance	Physical Address
PCIE	2180 01C8h

Figure 11-1194. PCIE_ERR_IRQ_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		ERR_AER	ERR_AXI	ERR_CORR	ERR_NONFATAL	ERR_FATAL	ERR_SYS
R-0h		R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; -*n* = value after reset

Table 11-2855. PCIE_ERR_IRQ_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reads return 0 and writes have no effect.
5	ERR_AER	R/W1S	0h	ECRC error interrupt enable. Set to enable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled
4	ERR_AXI	R/W1S	0h	AXI tag lookup fatal error interrupt enable. Set to enable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled
3	ERR_CORR	R/W1S	0h	Correctable error interrupt enable. Set to enable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled
2	ERR_NONFATAL	R/W1S	0h	Nonfatal error interrupt enable. Set to enable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled
1	ERR_FATAL	R/W1S	0h	Fatal error interrupt enable. Set to enable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled

Table 11-2855. PCIE_ERR_IRQ_ENABLE_SET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	ERR_SYS	R/W1S	0h	System Error (fatal, nonfatal or correctable error) interrupt enable. Set to enable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled

Table 11-2856. Register Call Summary for PCIE_ERR_IRQ_ENABLE_SET

Application Registers
- PCIE_ERR_IRQ_ENABLE_SET Register (Offset = 1C8h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.75 PCIE_ERR_IRQ_ENABLE_CLR Register (Offset = 1CCh) [reset = 0h]

The ERR Interrupt Enable Clear Register (PCIE_ERR_IRQ_ENABLE_CLR) is shown in Figure 11-1195 and described in Table 11-2858.

Table 11-2857. PCIE_ERR_IRQ_ENABLE_CLR Instances

Instance	Physical Address
PCIE	2180 01CCh

Figure 11-1195. PCIE_ERR_IRQ_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		ERR_AER	ERR_AXI	ERR_CORR	ERR_NONFATAL	ERR_FATAL	ERR_SYS
R-0h		R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-2858. PCIE_ERR_IRQ_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reads return 0 and writes have no effect.
5	ERR_AER	R/W1C	0h	ECRC error interrupt disable. Set to disable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled
4	ERR_AXI	R/W1C	0h	AXI tag lookup fatal error interrupt disable. Set to disable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled
3	ERR_CORR	R/W1C	0h	Correctable error interrupt disable. Set to disable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled
2	ERR_NONFATAL	R/W1C	0h	Nonfatal error interrupt disable. Set to disable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled
1	ERR_FATAL	R/W1C	0h	Fatal error interrupt disable. Set to disable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled

Table 11-2858. PCIE_ERR_IRQ_ENABLE_CLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	ERR_SYS	R/W1C	0h	System error (fatal, nonfatal or correctable error) interrupt disable. Set to disable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled

Table 11-2859. Register Call Summary for PCIE_ERR_IRQ_ENABLE_CLR

Application Registers
- PCIE_ERR_IRQ_ENABLE_CLR Register (Offset = 1CCh) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.76 PCIE_PMRST_IRQ_STATUS_RAW Register (Offset = 1D0h) [reset = 0h]

The Power Management and Reset Interrupt Status Register (PCIE_PMRST_IRQ_STATUS_RAW) is shown in Figure 11-1196 and described in Table 11-2861. The offset is 1D0h.

Table 11-2860. PCIE_PMRST_IRQ_STATUS_RAW Instances

Instance	Physical Address
PCIE	2180 01D0h

Figure 11-1196. PCIE_PMRST_IRQ_STATUS_RAW Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				LINK_RST_REQ	PM_PME	PM_TO_ACK	PM_TURNOFF
R-0h				R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; -*n* = value after reset

Table 11-2861. PCIE_PMRST_IRQ_STATUS_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3	LINK_RST_REQ	R/W1S	0h	Link Request Reset interrupt raw status.
2	PM_PME	R/W1S	0h	Power management PME message received interrupt raw status.
1	PM_TO_ACK	R/W1S	0h	Power management ACK received interrupt raw status.
0	PM_TURNOFF	R/W1S	0h	Power management turnoff message received raw status.

Table 11-2862. Register Call Summary for PCIE_PMRST_IRQ_STATUS_RAW

Application Registers
- PCIE_PMRST_IRQ_STATUS_RAW Register (Offset = 1D0h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.77 PCIE_PMRST_IRQ_STATUS Register (Offset = 1D4h) [reset = 0h]

The Power Management and Reset Interrupt Enabled Status Register (PCIE_PMRST_IRQ_STATUS) is shown in Figure 11-1197 and described in Table 11-2864. The offset is 1D4h.

Table 11-2863. PCIE_PMRST_IRQ_STATUS Instances

Instance	Physical Address
PCIE	2180 01D4h

Figure 11-1197. PCIE_PMRST_IRQ_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				LINK_RST_REQ	PM_PME	PM_TO_ACK	PM_TURNOFF
R-0h				R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2864. PCIE_PMRST_IRQ_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3	LINK_RST_REQ	R/W1C	0h	Link request reset interrupt status.
2	PM_PME	R/W1C	0h	Power management PME message received interrupt status.
1	PM_TO_ACK	R/W1C	0h	Power management ACK received interrupt status.
0	PM_TURNOFF	R/W1C	0h	Power management turnoff message received status.

Table 11-2865. Register Call Summary for PCIE_PMRST_IRQ_STATUS

Application Registers
- PCIE_PMRST_IRQ_STATUS Register (Offset = 1D4h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.78 PCIE_PMRST_ENABLE_SET Register (Offset = 1D8h) [reset = 0h]

The Power Management and Reset Interrupt Enable Set Register (PCIE_PMRST_ENABLE_SET) is shown in Figure 11-1198 and described in Table 11-2867. The offset is 1D8h.

Table 11-2866. PCIE_PMRST_ENABLE_SET Instances

Instance	Physical Address
PCIE	2180 01D8h

Figure 11-1198. PCIE_PMRST_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				LINK_RST_REQ	PM_PME	PM_TO_ACK	PM_TURNOFF
R-0h				R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

LEGEND: R = Read Only; R/W1S = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-2867. PCIE_PMRST_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3	LINK_RST_REQ	R/W1S	0h	Link request reset interrupt enable. Set to enable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled
2	PM_PME	R/W1S	0h	Power management PME message received interrupt enable. Set to enable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled
1	PM_TO_ACK	R/W1S	0h	Power management ACK received interrupt enable. Set to enable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled
0	PM_TURNOFF	R/W1S	0h	Power management turnoff message received enable. Set to enable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled

Table 11-2868. Register Call Summary for PCIE_PMRST_ENABLE_SET

Application Registers
- PCIE_PMRST_ENABLE_SET Register (Offset = 1D8h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.79 PCIE_PMRST_ENABLE_CLR Register (Offset = 1DCh) [reset = 0h]

The Power Management and Reset Interrupt Enable Clear Register (PCIE_PMRST_ENABLE_CLR) is shown in Figure 11-1199 and described in Table 11-2870.

Table 11-2869. PCIE_PMRST_ENABLE_CLR Instances

Instance	Physical Address
PCIE	2180 01DCh

Figure 11-1199. PCIE_PMRST_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				LINK_RST_REQ	PM_PME	PM_TO_ACK	PM_TURNOFF
R-0h				R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-2870. PCIE_PMRST_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3	LINK_RST_REQ	R/W1C	0h	Link Request Reset interrupt disable. Set to disable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled
2	PM_PME	R/W1C	0h	Power management PME message received interrupt disable. Set to disable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled
1	PM_TO_ACK	R/W1C	0h	Power management ACK received interrupt disable. Set to disable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled
0	PM_TURNOFF	R/W1C	0h	Power management Turnoff message received disable. Set to disable the interrupt. On read: • 0 = interrupt is disabled • 1 = interrupt is enabled

Table 11-2871. Register Call Summary for PCIE_PMRST_ENABLE_CLR

Application Registers
- PCIE_PMRST_ENABLE_CLR Register (Offset = 1DCh) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.80 PCIE_OB_OFFSET_INDEXn Register (Offset = 200h) [reset = 0h]

The Outbound Translation Region N Offset Low and Index Register (OB_OFFSET_INDEXn) is shown in Figure 11-1200 and described in Table 11-2873. The offset is (200 + N × 8)h.

Table 11-2872. PCIE_OB_OFFSET_INDEXn Instances

Instance	Physical Address
PCIE	2180 0200h

Figure 11-1200. PCIE_OB_OFFSET_INDEXn Register

31	30	29	28	27	26	25	24
OB_OFFSETN_LO							
R/W-0h							
23	22	21	20	19	18	17	16
OB_OFFSETN_LO				RESERVED			
R/W-0h				R-0h			
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							OB_ENABLEN
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2873. PCIE_OB_OFFSET_INDEXn Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	OB_OFFSETN_LO	R/W	0h	Offset bits [31-20] for translation region N (N=0 to 31).
19-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	OB_ENABLEN	R/W	0h	Enable translation region N (N=0 to 31). • 0 = outbound translation is disabled • 1 = outbound translation is enabled

Table 11-2874. Register Call Summary for PCIE_OB_OFFSET_INDEXn

Application Registers • Register Summary: [0]
PCIe SS Functional Description • Outbound Address Translation: [0][1][2][3]

11.14.5.3.81 PCIE_OB_OFFSETn_HI Register (Offset = 204h) [reset = 0h]

The Outbound Translation Region N Offset High Register (OB_OFFSETn_HI) is shown in Figure 11-1201 and described in Table 11-2876. The offset is (204+N*8)h.

Table 11-2875. PCIE_OB_OFFSETn_HI Instances

Instance	Physical Address
PCIE	2180 0204h

Figure 11-1201. PCIE_OB_OFFSETn_HI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OB_OFFSETN_HI																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2876. PCIE_OB_OFFSETn_HI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	OB_OFFSETN_HI	R/W	0h	Offset bits [63-32] for translation region N (N=0 to 31).

Table 11-2877. Register Call Summary for PCIE_OB_OFFSETn_HI

Application Registers • Register Summary: [0]
PCIe SS Functional Description • Outbound Address Translation: [0][1][2][3]

11.14.5.3.82 PCIE_IB_BAR0 Register (Offset = 300h) [reset = 0h]

The Inbound Translation Bar Match 0 Register (PCIE_IB_BAR0) is shown in Figure 11-1202 and described in Table 11-2879.

Table 11-2878. PCIE_IB_BAR0 Instances

Instance	Physical Address
PCIE	2180 0300h

Figure 11-1202. PCIE_IB_BAR0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					IB_BAR0		
R-0h					R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2879. PCIE_IB_BAR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reads return 0 and writes have no effect.
2-0	IB_BAR0	R/W	0h	BAR number to match for inbound translation region 0.

Table 11-2880. Register Call Summary for PCIE_IB_BAR0

Application Registers
- PCIE_IB_BAR0 Register (Offset = 300h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.83 PCIE_IB_START0_LO Register (Offset = 304h) [reset = 0h]

The Inbound Translation 0 Start Address Low Register (PCIE_IB_START0_LO) is shown in Figure 11-1203 and described in Table 11-2882.

Table 11-2881. PCIE_IB_START0_LO Instances

Instance	Physical Address
PCIE	2180 0304h

Figure 11-1203. PCIE_IB_START0_LO Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IB_START0_LO																								RESERVED							
R/W-0h																								R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2882. PCIE_IB_START0_LO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	IB_START0_LO	R/W	0h	Start address bits [31-8] for inbound translation region 0.
7-0	RESERVED	R	0h	Reads return 0 and writes have no effect.

Table 11-2883. Register Call Summary for PCIE_IB_START0_LO

Application Registers
- PCIE_IB_START0_LO Register (Offset = 304h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.84 PCIE_IB_START0_HI Register (Offset = 308h) [reset = 0h]

The Inbound Translation 0 Start Address High Register (PCIE_IB_START0_HI) is shown in Figure 11-1204 and described in Table 11-2885.

Table 11-2884. PCIE_IB_START0_HI Instances

Instance	Physical Address
PCIE	2180 0308h

Figure 11-1204. PCIE_IB_START0_HI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IB_START0_HI																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2885. PCIE_IB_START0_HI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IB_START0_HI	R/W	0h	Start address bits [63-32] for inbound translation region 0.

Table 11-2886. Register Call Summary for PCIE_IB_START0_HI

Application Registers
- Register Summary: [0]
- PCIE_IB_START0_HI Register (Offset = 308h) [reset = 0h]: [0]

11.14.5.3.85 PCIE_IB_OFFSET0 Register (Offset = 30Ch) [reset = 0h]

The Inbound Translation 0 Address Offset Register (PCIE_IB_OFFSET0) is shown in Figure 11-1205 and described in Table 11-2888.

Table 11-2887. PCIE_IB_OFFSET0 Instances

Instance	Physical Address
PCIE	2180 030Ch

Figure 11-1205. PCIE_IB_OFFSET0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IB_OFFSET0																								RESERVED							
R/W-0h																								R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2888. PCIE_IB_OFFSET0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	IB_OFFSET0	R/W	0h	Offset address bits [31-8] for inbound translation region 0.
7-0	RESERVED	R	0h	Reads return 0 and writes have no effect.

Table 11-2889. Register Call Summary for PCIE_IB_OFFSET0

Application Registers
- PCIE_IB_OFFSET0 Register (Offset = 30Ch) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.86 PCIE_IB_BAR1 Register (Offset = 310h) [reset = 0h]

The Inbound Translation Bar Match 1 Register (PCIE_IB_BAR1)is shown in Figure 11-1206 and described in Table 11-2891.

Table 11-2890. PCIE_IB_BAR1 Instances

Instance	Physical Address
PCIE	2180 0310h

Figure 11-1206. PCIE_IB_BAR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					IB_BAR1		
R-0h					R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2891. PCIE_IB_BAR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reads return 0 and writes have no effect.
2-0	IB_BAR1	R/W	0h	BAR number to match for inbound translation region 1.

Table 11-2892. Register Call Summary for PCIE_IB_BAR1

Application Registers • PCIE_IB_BAR1 Register (Offset = 310h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • Mapping Multiple Non-Contiguous Memory Ranges To One Region: [0] • Inbound Address Translation: [0][1]

11.14.5.3.87 PCIE_IB_START1_LO Register (Offset = 314h) [reset = 0h]

The Inbound Translation 1 Start Address Low Register (PCIE_IB_START1_LO) is shown in Figure 11-1207 and described in Table 11-2894.

Table 11-2893. PCIE_IB_START1_LO Instances

Instance	Physical Address
PCIE	2180 0314h

Figure 11-1207. PCIE_IB_START1_LO Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IB_START1_LO																								RESERVED							
R/W-0h																								R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2894. PCIE_IB_START1_LO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	IB_START1_LO	R/W	0h	Start address bits [31-8] for inbound translation region 1.
7-0	RESERVED	R	0h	Reads return 0 and writes have no effect.

Table 11-2895. Register Call Summary for PCIE_IB_START1_LO

Application Registers • PCIE_IB_START1_LO Register (Offset = 314h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • Inbound Address Translation: [0]

11.14.5.3.88 PCIE_IB_START1_HI Register (Offset = 318h) [reset = 0h]

The Inbound Translation 1 Start Address High Register (PCIE_IB_START1_HI) is shown in Figure 11-1208 and described in Table 11-2897.

Table 11-2896. PCIE_IB_START1_HI Instances

Instance	Physical Address
PCIE	2180 0318h

Figure 11-1208. PCIE_IB_START1_HI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IB_START1_HI																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2897. PCIE_IB_START1_HI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IB_START1_HI	R/W	0h	Start address bits [63-32] for inbound translation region 1.

Table 11-2898. Register Call Summary for PCIE_IB_START1_HI

Application Registers • PCIE_IB_START1_HI Register (Offset = 318h) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • Inbound Address Translation: [0]

11.14.5.3.89 PCIE_IB_OFFSET1 Register (Offset = 31Ch) [reset = 0h]

The Inbound Translation 1 Address Offset Register (PCIE_IB_OFFSET1) is shown in Figure 11-1209 and described in Table 11-2900.

Table 11-2899. PCIE_IB_OFFSET1 Instances

Instance	Physical Address
PCIE	2180 031Ch

Figure 11-1209. PCIE_IB_OFFSET1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IB_OFFSET1																								RESERVED							
R/W-0h																								R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2900. PCIE_IB_OFFSET1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	IB_OFFSET1	R/W	0h	Offset address bits [31-8] for inbound translation region 1.
7-0	RESERVED	R	0h	Reads return 0 and writes have no effect.

Table 11-2901. Register Call Summary for PCIE_IB_OFFSET1

Application Registers • PCIE_IB_OFFSET1 Register (Offset = 31Ch) [reset = 0h]: [0] • Register Summary: [0]
PCIe SS Functional Description • Inbound Address Translation: [0]

11.14.5.3.90 PCIE_IB_BAR2 Register (Offset = 320h) [reset = 0h]

The Inbound Translation Bar Match 2 Register (PCIE_IB_BAR2) is shown in Figure 11-1210 and described in Table 11-2903.

Table 11-2902. PCIE_IB_BAR2 Instances

Instance	Physical Address
PCIE	2180 0320h

Figure 11-1210. PCIE_IB_BAR2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					IB_BAR2		
R-0h					R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2903. PCIE_IB_BAR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reads return 0 and writes have no effect.
2-0	IB_BAR2	R/W	0h	BAR number to match for inbound translation region 2.

Table 11-2904. Register Call Summary for PCIE_IB_BAR2

Application Registers • Register Summary: [0] • PCIE_IB_BAR2 Register (Offset = 320h) [reset = 0h]: [0]
PCIe SS Functional Description • Mapping Multiple Non-Contiguous Memory Ranges To One Region: [0]

11.14.5.3.91 PCIE_IB_START2_LO Register (Offset = 324h) [reset = 0h]

The Inbound Translation 2 Start Address Low Register (PCIE_IB_START2_LO) is shown in Figure 11-1211 and described in Table 11-2906.

Table 11-2905. PCIE_IB_START2_LO Instances

Instance	Physical Address
PCIE	2180 0324h

Figure 11-1211. PCIE_IB_START2_LO Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IB_START2_LO																								RESERVED							
R/W-0h																								R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2906. PCIE_IB_START2_LO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	IB_START2_LO	R/W	0h	Start address bits [31-8] for inbound translation region 2.
7-0	RESERVED	R	0h	Reads return 0 and writes have no effect.

Table 11-2907. Register Call Summary for PCIE_IB_START2_LO

Application Registers
- PCIE_IB_START2_LO Register (Offset = 324h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.92 PCIE_IB_START2_HI Register (Offset = 328h) [reset = 0h]

The Inbound Translation 2 Start Address High Register (PCIE_IB_START2_HI) is shown in Figure 11-1212 and described in Table 11-2909.

Table 11-2908. PCIE_IB_START2_HI Instances

Instance	Physical Address
PCIE	2180 0328h

Figure 11-1212. PCIE_IB_START2_HI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IB_START2_HI																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2909. PCIE_IB_START2_HI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IB_START2_HI	R/W	0h	Start address bits [63-32] for inbound translation region 2.

Table 11-2910. Register Call Summary for PCIE_IB_START2_HI

Application Registers
- PCIE_IB_START2_HI Register (Offset = 328h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.93 PCIE_IB_OFFSET2 Register (Offset = 32Ch) [reset = 0h]

The Inbound Translation 2 Address Offset Register (PCIE_IB_OFFSET2) is shown in Figure 11-1213 and described in Table 11-2912.

Table 11-2911. PCIE_IB_OFFSET2 Instances

Instance	Physical Address
PCIE	2180 032Ch

Figure 11-1213. PCIE_IB_OFFSET2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IB_OFFSET2																								RESERVED							
R/W-0h																								R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2912. PCIE_IB_OFFSET2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	IB_OFFSET2	R/W	0h	Offset address bits [31-8] for inbound translation region 2.
7-0	RESERVED	R	0h	Reads return 0 and writes have no effect.

Table 11-2913. Register Call Summary for PCIE_IB_OFFSET2

Application Registers
- PCIE_IB_OFFSET2 Register (Offset = 32Ch) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.94 PCIE_IB_BAR3 Register (Offset = 330h) [reset = 0h]

The Inbound Translation Bar Match 3 Register (PCIE_IB_BAR3) is shown in Figure 11-1214 and described in Table 11-2915.

Table 11-2914. PCIE_IB_BAR3 Instances

Instance	Physical Address
PCIE	2180 0330h

Figure 11-1214. PCIE_IB_BAR3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					IB_BAR3		
R-0h					R/W-0h		

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2915. PCIE_IB_BAR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reads return 0 and writes have no effect.
2-0	IB_BAR3	R/W	0h	BAR number to match for inbound translation region 3.

Table 11-2916. Register Call Summary for PCIE_IB_BAR3

Application Registers
- PCIE_IB_BAR3 Register (Offset = 330h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.95 PCIE_IB_START3_LO Register (Offset = 334h) [reset = 0h]

The Inbound Translation 3 Start Address Low Register (PCIE_IB_START3_LO) is shown in Figure 11-1215 and described in Table 11-2918.

Table 11-2917. PCIE_IB_START3_LO Instances

Instance	Physical Address
PCIE	2180 0334h

Figure 11-1215. PCIE_IB_START3_LO Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IB_START3_LO																								RESERVED							
R/W-0h																								R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2918. PCIE_IB_START3_LO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	IB_START3_LO	R/W	0h	Start address bits [31-8] for inbound translation region 3.
7-0	RESERVED	R	0h	Reads return 0 and writes have no effect.

Table 11-2919. Register Call Summary for PCIE_IB_START3_LO

Application Registers
- PCIE_IB_START3_LO Register (Offset = 334h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.96 PCIE_IB_START3_HI Register (Offset = 338h) [reset = 0h]

The Inbound Translation 3 Start Address High Register (PCIE_IB_START3_HI) is shown in Figure 11-1216 and described in Table 11-2921.

Table 11-2920. PCIE_IB_START3_HI Instances

Instance	Physical Address
PCIE	2180 0338h

Figure 11-1216. PCIE_IB_START3_HI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IB_START3_HI																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2921. PCIE_IB_START3_HI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IB_START3_HI	R/W	0h	Start address bits [63-32] for inbound translation region 3.

Table 11-2922. Register Call Summary for PCIE_IB_START3_HI

Application Registers
- PCIE_IB_START3_HI Register (Offset = 338h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.3.97 PCIE_IB_OFFSET3 Register (Offset = 33Ch) [reset = 0h]

The Inbound Translation 3 Address Offset Register (PCIE_IB_OFFSET3) is shown in Figure 11-1217 and described in Table 11-2924.

Table 11-2923. PCIE_IB_OFFSET3 Instances

Instance	Physical Address
PCIE	2180 033Ch

Figure 11-1217. PCIE_IB_OFFSET3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IB_OFFSET3																								RESERVED							
R/W-0h																								R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2924. PCIE_IB_OFFSET3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	IB_OFFSET3	R/W	0h	Offset address bits [31-8] for inbound translation region 3.
7-0	RESERVED	R	0h	Reads return 0 and writes have no effect.

Table 11-2925. Register Call Summary for PCIE_IB_OFFSET3

Application Registers
- PCIE_IB_OFFSET3 Register (Offset = 33Ch) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.4 Configuration Registers Common to Type 0 and Type 1 Headers

11.14.5.4.1 Register Summary

Table 11-2926. PCIe Instances

Instance	Base Address
PCIE	2180 0000h

Table 11-2927. Configuration Registers Common to Type 0 and Type 1 Headers

Offset	Acronym	Register Name	PCIE Physical Address	Section
1000h	PCIE_VENDOR_DEVICE_ID	Vendor and Device Identification Register	2180 1000h	Section 11.14.5.4.2
1004h	PCIE_STATUS_COMMAND	Status and Command Register	2180 1004h	Section 11.14.5.4.3
1008h	PCIE_CLASSCODE_REVID	Class Code and Revision ID Register	2180 1008h	Section 11.14.5.4.4

11.14.5.4.2 PCIE_VENDOR_DEVICE_ID Register (Offset = 1000h) [reset = 104Ch]

The Vendor and Device Identification Register (PCIE_VENDOR_DEVICE_ID) is shown in Figure 11-1218 and described in Table 11-2929.

Table 11-2928. PCIE_VENDOR_DEVICE_ID Instances

Instance	Physical Address
PCIE	2180 1000h

Figure 11-1218. PCIE_VENDOR_DEVICE_ID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 104Ch																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2929. PCIE_VENDOR_DEVICE_ID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	104Ch	TI internal data. Identifies revision of peripheral.

Table 11-2930. Register Call Summary for PCIE_VENDOR_DEVICE_ID

Configuration Registers Common to Type 0 and Type 1 Headers

- PCIE_VENDOR_DEVICE_ID Register (Offset = 1000h) [reset = 104Ch]: [0]
- Register Summary: [0]

11.14.5.4.3 PCIE_STATUS_COMMAND Register (Offset = 1004h) [reset = 100000h]

The Status and Command Register (PCIE_STATUS_COMMAND) is shown in Figure 11-1219 and described in Table 11-2932.

Table 11-2931. PCIE_STATUS_COMMAND Instances

Instance	Physical Address
PCIE	2180 1004h

Figure 11-1219. PCIE_STATUS_COMMAND Register

31	30	29	28	27	26	25	24
PARITY_ERROR	SIG_SYS_ERROR	RX_MST_ABORT	RX_TGT_ABORT	SIG_TGT_ABORT	RESERVED		DAT_PAR_ERROR
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R-0h		R/W1C-0h
23	22	21	20	19	18	17	16
RESERVED			CAP_LIST	INT_STAT	RESERVED		
R-0h			R-1h	R-0h	R-0h		
15	14	13	12	11	10	9	8
RESERVED					INTX_DIS	RESERVED	SERR_EN
R-0h					R/W-0h	R-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	PAR_ERR_RESP	RESERVED			BUS_MS	MEM_SP	IO_SP
R-0h	R/W-0h	R-0h			R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-2932. PCIE_STATUS_COMMAND Register Field Descriptions

Bit	Field	Type	Reset	Description
31	PARITY_ERROR	R/W1C	0h	Set if function receives poisoned TLP.
30	SIG_SYS_ERROR	R/W1C	0h	Set if function sends an ERR_FATAL or ERR_NONFATAL message and SERR enable bit is set to one.
29	RX_MST_ABORT	R/W1C	0h	Set when a requester receives a completion with unsupported request completion status.
28	RX_TGT_ABORT	R/W1C	0h	Set when a requester receives a completion with completer abort status.
27	SIG_TGT_ABORT	R/W1C	0h	Set when a function acting as a completer terminates a request by issuing completer abort completion status to the requester.
26-25	RESERVED	R	0h	Reads return 0 and writes have no effect.
24	DAT_PAR_ERROR	R/W1C	0h	This bit is set by a requester if the Parity Error Enable bit is set in its Command register and either the condition that the requester receives a poisoned completion or the condition that the requester poisons a write request is true.
23-21	RESERVED	R	0h	Reads return 0 and writes have no effect.
20	CAP_LIST	R	1h	For PCIe, this field must be set to 1.
19	INT_STAT	R	0h	Indicates that the function has received an interrupt.
18-11	RESERVED	R	0h	Reads return 0 and writes have no effect.
10	INTX_DIS	R/W	0h	Setting this bit disables generation of INTx messages.
9	RESERVED	R	0h	Reads return 0 and writes have no effect.
8	SERR_EN	R/W	0h	When set, it enables generation of the appropriate PCI Express error messages to the Root Complex.
7	RESERVED	R	0h	Reads return 0 and writes have no effect.

Table 11-2932. PCIE_STATUS_COMMAND Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	PAR_ERR_RESP	R/W	0h	This bit controls whether or not the device responds to detected parity errors (poisoned TLP). This error is typically reported as an unsupported request and may also result in a non-fatal error message if SERR_EN=1. If this bit is set, the PCIESS will respond normally to parity errors. If this bit is cleared, the PCIESS will ignore detected parity errors.
5-3	RESERVED	R	0h	Reads return 0 and writes have no effect.
2	BUS_MS	R/W	0h	Enables mastership of the bus.
1	MEM_SP	R/W	0h	This bit is set to enable the device to respond to memory accesses.
0	IO_SP	R/W	0h	This bit is set to enable the device to respond to I/O accesses. This functionality is not supported in PCIESS and therefore this bit is set to 0.

Table 11-2933. Register Call Summary for PCIE_STATUS_COMMAND

PCIe SS Functional Description • Bus Mastering: [0] • PCI-Compatible Error Handling: [0][1]
Configuration Registers Common to Type 0 and Type 1 Headers • Register Summary: [0] • PCIE_STATUS_COMMAND Register (Offset = 1004h) [reset = 100000h]: [0]

11.14.5.4.4 PCIE_CLASSCODE_REVID Register (Offset = 1008h) [reset = 1h]

The Class Code and Revision ID Register (PCIE_CLASSCODE_REVID) is shown in Figure 11-1220 and described in Table 11-2935.

Table 11-2934. PCIE_CLASSCODE_REVID Instances

Instance	Physical Address
PCIE	2180 1008h

Figure 11-1220. PCIE_CLASSCODE_REVID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R - 1h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2935. PCIE_CLASSCODE_REVID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	1h	TI internal data. Identifies revision of peripheral.

Table 11-2936. Register Call Summary for PCIE_CLASSCODE_REVID

Configuration Registers Common to Type 0 and Type 1 Headers
- PCIE_CLASSCODE_REVID Register (Offset = 1008h) [reset = 1h]: [0]
- Register Summary: [0]

11.14.5.5 Configuration Type 0 Registers

11.14.5.5.1 Register Summary

Table 11-2937. PCIe Instances

Instance	Base Address
PCIE	2180 0000h

Table 11-2938. Configuration Type 0 Registers

Offset	Acronym	Register Name	PCIE Physical Address	Section
100Ch	PCIE_BIST_HEADER	BIST, Header Type, Latency Time and Cache Line Size Register	2180 100Ch	Section 11.14.5.5.2
1010h	PCIE_BAR0	Base Address Register 0	2180 1010h	Section 11.14.5.5.3
1010h	PCIE_BAR0_MASK	Register	2180 1010h	Section 11.14.5.5.4
1014h	PCIE_BAR1	Base Address Register 1	2180 1014h	Section 11.14.5.5.5
1014h	PCIE_BAR1_MASK	Register	2180 1014h	Section 11.14.5.5.6
1014h	PCIE_BAR1 (64 bit PCIE_BAR0)	Base Address Register 1 (64 bit PCIE_BAR0)	2180 1014h	Section 11.14.5.5.7
1014h	PCIE_BAR1_MASK (64 bit PCIE_BAR0)	Register (64 bit PCIE_BAR0)	2180 1014h	Section 11.14.5.5.8
1018h	PCIE_BAR2	Base Address Register 2	2180 1018h	Section 11.14.5.5.9
1018h	PCIE_BAR2_MASK	Register	2180 1018h	Section 11.14.5.5.10
101Ch	PCIE_BAR3	Base Address Register 3	2180 101Ch	Section 11.14.5.5.11
101Ch	PCIE_BAR3_MASK	Register	2180 101Ch	Section 11.14.5.5.12
101Ch	PCIE_BAR3 (64 bit PCIE_BAR1)	Base Address Register 3 (64 bit PCIE_BAR1)	2180 101Ch	Section 11.14.5.5.13
101Ch	PCIE_BAR3_MASK (64 bit PCIE_BAR1)	Register (64 bit PCIE_BAR1)	2180 101Ch	Section 11.14.5.5.14
1020h	PCIE_BAR4	Base Address Register 4	2180 1020h	Section 11.14.5.5.15
1020h	PCIE_BAR4_MASK	Register	2180 1020h	Section 11.14.5.5.16
1024h	PCIE_BAR5	Base Address Register 5	2180 1024h	Section 11.14.5.5.17
1024h	PCIE_BAR5_MASK	Register	2180 1024h	Section 11.14.5.5.18
1024h	PCIE_BAR5 (64 bit PCIE_BAR4)	Base Address Register 5 (64 bit PCIE_BAR4)	2180 1024h	Section 11.14.5.5.19
1024h	PCIE_BAR5_MASK (64 bit PCIE_BAR4)	Register (64 bit PCIE_BAR4)	2180 1024h	Section 11.14.5.5.20
102Ch	PCIE_SUBSYS_VNDR_ID	Subsystem and Subsystem Vendor ID	2180 102Ch	Section 11.14.5.5.21
1030h	PCIE_EXPNSN_ROM	Expansion ROM Base Address	2180 1030h	Section 11.14.5.5.22
1034h	PCIE_CAP_PTR	Capabilities Pointer	2180 1034h	Section 11.14.5.5.23
103Ch	PCIE_INT_PIN	Interrupt Pin Register	2180 103Ch	Section 11.14.5.5.24

11.14.5.5.2 PCIE_BIST_HEADER Register (Offset = 100Ch) [reset = 0h]

The BIST, Header Type, Latency Time and Cache Line Size Register (PCIE_BIST_HEADER) is shown in Figure 11-1221 and described in Table 11-2940.

Table 11-2939. PCIE_BIST_HEADER Instances

Instance	Physical Address
PCIE	2180 100Ch

Figure 11-1221. PCIE_BIST_HEADER Register

31	30	29	28	27	26	25	24
BIST_CAP	START_BIST	RESERVED		COMP_CODE			
R-0h	R-0h	R-0h		R-0h			
23	22	21	20	19	18	17	16
MULFUN_DEV	HDR_TYPE						
R-0h	R-0h						
15	14	13	12	11	10	9	8
LAT_TMR							
R-0h							
7	6	5	4	3	2	1	0
CACHELN_SIZ							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2940. PCIE_BIST_HEADER Register Field Descriptions

Bit	Field	Type	Reset	Description
31	BIST_CAP	R	0h	Returns a one for BIST capability and zero otherwise. Not supported by PCIESS.
30	START_BIST	R	0h	Write a one to start BIST. Not supported by PCIESS.
29-28	RESERVED	R	0h	Reads return 0 and writes have no effect.
27-24	COMP_CODE	R	0h	Completion code. Not supported by PCIESS.
23	MULFUN_DEV	R	0h	Returns 1 if it is a multi-function device. Writable from internal bus interface.
22-16	HDR_TYPE	R	0h	Configuration header format. • 0 = EP mode • 1 = RC mode
15-8	LAT_TMR	R	0h	Not applicable in PCIe
7-0	CACHELN_SIZ	R/W	0h	Not applicable in PCIe

Table 11-2941. Register Call Summary for PCIE_BIST_HEADER

Configuration Type 0 Registers • PCIE_BIST_HEADER Register (Offset = 100Ch) [reset = 0h]: [0] • Register Summary: [0]
Configuration Type 1 Registers • PCIE_BIST_HEADER Register (Offset = 100Ch) [reset = 10000h]: [0]

11.14.5.5.3 PCIE_BAR0 Register (Offset = 1010h) [reset = 0h]

The Base Address Register 0 (PCIE_BAR0) is shown in Figure 11-1222 and described in Table 11-2943.

Table 11-2942. PCIE_BAR0 Instances

Instance	Physical Address
PCIE	2180 1010h

Figure 11-1222. PCIE_BAR0 Register

31	30	29	28	27	26	25	24
BASE_ADDR							
R/W-0h							
23	22	21	20	19	18	17	16
BASE_ADDR							
R/W-0h							
15	14	13	12	11	10	9	8
BASE_ADDR							
R/W-0h							
7	6	5	4	3	2	1	0
BASE_ADDR				PREFETCH	TYPE		MEM_SPACE
R/W-0h				R-0h	R-0h		R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2943. PCIE_BAR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	BASE_ADDR	R/W	0h	Base Address. Based on the configuration of Register, not all bits may be writable.
3	PREFETCH	R	0h	For memory BARs, it indicates whether the region is prefetchable. Writable from internal bus interface. • 0 = Non-prefetchable. • 1= Prefetchable. For I/O Bars, it is used as second least significant bit (LSB) of the base address. Writable from internal bus interface.
2-1	TYPE	R	0h	For memory BARs, they determine the BAR type. Writable from internal bus interface. • 00 = 32-bit BAR. • 10 = 64-bit BAR. • Others = Reserved. For I/O BARs, bit 2 is the least significant bit (LSB) of the base address and bit 1 is 0. Writable from internal bus interface.
0	MEM_SPACE	R	0h	Writable from internal bus interface. • 0 = Memory BAR. • 1 = I/O BAR.

Table 11-2944. Register Call Summary for PCIE_BAR0

Configuration Type 0 Registers • PCIE_BAR0 Register (Offset = 1010h) [reset = 0h]: [0] • Register Summary: [0][1][2][3][4] • PCIE_BAR1 (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0][1][2] • PCIE_BAR0_MASK Register (Offset = 1010h) [reset = 0h]: [0][1][2][3][4] • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0][1][2]
Application Registers • PCIE_MSI_IRQ Register (Offset = 54h) [reset = 0h]: [0]

Table 11-2944. Register Call Summary for PCIE_BAR0 (continued)

PCIe SS Registers • Application Registers: [0]
PCIe SS Functional Description • PHY Loopback: [0] • PCIE_IB_BAR0 Exception for In-Bound Address Translation: [0][1][2][3][4] • Inbound Address Translation: [0][1][2][3][4][5][6][7][8] • MSI Interrupt Generation in EP Mode: [0] • Interrupt Reception in EP Mode: [0] • Address Space 0: [0] • BAR Mask Registers: [0][1][2][3][4][5][6]
Configuration Type 1 Registers • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0][1][2] • PCIE_BAR0 Register (Offset = 1010h) [reset = 0h]: [0] • PCIE_BAR0_MASK Register (Offset = 1010h) [reset = 0h]: [0][1][2][3][4] • PCIE_BAR1 (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0][1][2]

11.14.5.5.4 PCIE_BAR0_MASK Register (Offset = 1010h) [reset = 0h]

The Register (PCIE_BAR0_MASK) is shown in Figure 11-1223 and described in Table 11-2946. (same as PCIE_BAR0, but requires DBI_CS2 bit set in PCIE_CMD_STATUS register, see for details).

Table 11-2945. PCIE_BAR0_MASK Instances

Instance	Physical Address
PCIE	2180 1010h

Figure 11-1223. PCIE_BAR0_MASK Register

31	30	29	28	27	26	25	24
BAR_MASK							
W-0h							
23	22	21	20	19	18	17	16
BAR_MASK							
W-0h							
15	14	13	12	11	10	9	8
BAR_MASK							
W-0h							
7	6	5	4	3	2	1	0
BAR_MASK							BAR_ENABLED
W-0h							W-0h

LEGEND: W = Write Only; *-n* = value after reset

Table 11-2946. PCIE_BAR0_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	BAR_MASK	W	0h	Indicates which PCIE_BAR0 bits to mask (non-writeable) from host, which determines the size of the BAR. The bits are writeable only, not readable.
0	BAR_ENABLED	W	0h	PCIE_BAR0 Enable. The bit is writeable only, not readable. • 0 = PCIE_BAR0 is disabled. • 1 = PCIE_BAR0 is enabled.

Table 11-2947. Register Call Summary for PCIE_BAR0_MASK

Configuration Type 0 Registers • Register Summary: [0] • PCIE_BAR0_MASK Register (Offset = 1010h) [reset = 0h]: [0]
PCIe SS Functional Description • BAR Mask Registers: [0]
Configuration Type 1 Registers • PCIE_BAR0_MASK Register (Offset = 1010h) [reset = 0h]: [0]

11.14.5.5.5 PCIE_BAR1 Register (Offset = 1014h) [reset = 8h]

The Base Address Register 1 (PCIE_BAR1) is shown in Figure 11-1224 and described in Table 11-2949.

Table 11-2948. PCIE_BAR1 Instances

Instance	Physical Address
PCIE	2180 1014h

Figure 11-1224. PCIE_BAR1 Register

31	30	29	28	27	26	25	24
BASE_ADDR							
R/W-0h							
23	22	21	20	19	18	17	16
BASE_ADDR							
R/W-0h							
15	14	13	12	11	10	9	8
BASE_ADDR							
R/W-0h							
7	6	5	4	3	2	1	0
BASE_ADDR				PREFETCH	TYPE		MEM_SPACE
R/W-0h				R-1h	R-0h		R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2949. PCIE_BAR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	BASE_ADDR	R/W	0h	Base Address. Based on the configuration of Register, not all bits may be writable.
3	PREFETCH	R	1h	For memory BARs, it indicates whether the region is prefetchable. Writable from internal bus interface. • 0 = Non-prefetchable. • 1= Prefetchable. For I/O Bars, it is used as second least significant bit (LSB) of the base address. Writable from internal bus interface.
2-1	TYPE	R	0h	For memory BARs, they determine the BAR type. Writable from internal bus interface. • 00 = 32-bit BAR. • 10 = 64-bit BAR. • Others = Reserved. For I/O BARs, bit 2 is the least significant bit (LSB) of the base address and bit 1 is 0. Writable from internal bus interface.
0	MEM_SPACE	R	0h	Writable from internal bus interface. • 0 = Memory BAR. • 1 = I/O BAR.

Table 11-2950. Register Call Summary for PCIE_BAR1

Configuration Type 0 Registers

- PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0][1][2][3][4]
- Register Summary: [0][1][2][3][4][5]
- PCIE_BAR1 Register (Offset = 1014h) [reset = 8h]: [0]
- PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0]
- PCIE_BAR1 (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0]

Table 11-2950. Register Call Summary for PCIE_BAR1 (continued)

PCIe SS Functional Description • Using PCIE_IB_BAR1 Value As Start Address: [0][1][2][3][4][5][6] • PHY Loopback: [0] • Mapping Multiple Non-Contiguous Memory Ranges To One Region: [0][1] • Inbound Address Translation: [0][1][2][3][4][5][6][7]
Configuration Type 1 Registers • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0][1][2][3][4] • PCIE_BAR1 (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR1 Register (Offset = 1014h) [reset = 8h]: [0]

11.14.5.5.6 PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]

The Register (PCIE_BAR1_MASK) is shown in Figure 11-1225 and described in Table 11-2952. The offset is 1014h (same as PCIE_BAR1, but requires DBI_CS2 bit set in PCIE_CMD_STATUS register, see for details).

Table 11-2951. PCIE_BAR1_MASK Instances

Instance	Physical Address
PCIE	2180 1014h

Figure 11-1225. PCIE_BAR1_MASK Register

31	30	29	28	27	26	25	24
BAR_MASK							
W-0h							
23	22	21	20	19	18	17	16
BAR_MASK							
W-0h							
15	14	13	12	11	10	9	8
BAR_MASK							
W-0h							
7	6	5	4	3	2	1	0
BAR_MASK							BAR_ENABLED
W-0h							W-0h

LEGEND: W = Write Only; *-n* = value after reset

Table 11-2952. PCIE_BAR1_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	BAR_MASK	W	0h	Indicates which PCIE_BAR1 bits to mask (non-writeable) from host, which determines the size of the BAR. The bits are writeable only, not readable.
0	BAR_ENABLED	W	0h	PCIE_BAR1 Enable. The bit is writeable only, not readable. • 0 = PCIE_BAR1 is disabled. • 1 = PCIE_BAR1 is enabled.

Table 11-2953. Register Call Summary for PCIE_BAR1_MASK

Configuration Type 0 Registers • PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0] • Register Summary: [0][1] • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0]
Configuration Type 1 Registers • PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0]

11.14.5.5.7 PCIE_BAR1 (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]

The Base Address Register 1 (PCIE_BAR1) (64bit PCIE_BAR0) is shown in Figure 11-1226 and described in Table 11-2955.

Table 11-2954. PCIE_BAR1 (64 bit BAR0) Instances

Instance	Physical Address
PCIE	2180 1014h

Figure 11-1226. PCIE_BAR1 (64 bit BAR0) Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2955. PCIE_BAR1 (64 bit BAR0) Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BASE_ADDR	R/W	0h	Upper 32 bits of PCIE_BAR0 address if PCIE_BAR0 is a 64-bit BAR. Based on the configuration of Register, not all bits may be writable.

Table 11-2956. Register Call Summary for PCIE_BAR1

Configuration Type 0 Registers • PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0][1][2][3][4] • Register Summary: [0][1][2][3][4][5] • PCIE_BAR1 Register (Offset = 1014h) [reset = 8h]: [0] • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR1 (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0]
PCIe SS Functional Description • Using PCIE_IB_BAR1 Value As Start Address: [0][1][2][3][4][5][6] • PHY Loopback: [0] • Mapping Multiple Non-Contiguous Memory Ranges To One Region: [0][1] • Inbound Address Translation: [0][1][2][3][4][5][6][7]
Configuration Type 1 Registers • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0][1][2][3][4] • PCIE_BAR1 (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR1 Register (Offset = 1014h) [reset = 8h]: [0]

11.14.5.5.8 PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]

The Register (PCIE_BAR1_MASK) (64bit PCIE_BAR0) is shown in Figure 11-1227 and described in Table 11-2958. The offset is 1014h (same as PCIE_BAR1, but requires DBI_CS2 bit set in PCIE_CMD_STATUS register, see for details).

Table 11-2957. PCIE_BAR1_MASK (64 bit BAR0) Instances

Instance	Physical Address
PCIE	2180 1014h

Figure 11-1227. PCIE_BAR1_MASK (64 bit BAR0) Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BAR_MASK																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 11-2958. PCIE_BAR1_MASK (64 bit BAR0) Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BAR_MASK	W	0h	Upper 32 bits of PCIE_BAR0 mask value if PCIE_BAR0 is a 64-bit BAR. Indicates which BAR bits to mask (non-writeable) from host, which determines the size of the BAR. The bits are writeable only, not readable.

Table 11-2959. Register Call Summary for PCIE_BAR1_MASK

Configuration Type 0 Registers • PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0] • Register Summary: [0][1] • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0]
Configuration Type 1 Registers • PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0]

11.14.5.5.9 PCIE_BAR2 Register (Offset = 1018h) [reset = 8h]

The Base Address Register 2 (PCIE_BAR2) is shown in Figure 11-1228 and described in Table 11-2961.

Table 11-2960. PCIE_BAR2 Instances

Instance	Physical Address
PCIE	2180 1018h

Figure 11-1228. PCIE_BAR2 Register

31	30	29	28	27	26	25	24
BASE_ADDR							
R/W-0h							
23	22	21	20	19	18	17	16
BASE_ADDR							
R/W-0h							
15	14	13	12	11	10	9	8
BASE_ADDR							
R/W-0h							
7	6	5	4	3	2	1	0
BASE_ADDR				PREFETCH	TYPE		MEM_SPACE
R/W-0h				R-1h	R-0h		R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2961. PCIE_BAR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	BASE_ADDR	R/W	0h	Base Address. Based on the configuration of Register, not all bits may be writable.
3	PREFETCH	R	1h	For memory BARs, it indicates whether the region is prefetchable. Writable from internal bus interface. • 0 = Non-prefetchable. • 1= Prefetchable. For I/O Bars, it is used as second least significant bit (LSB) of the base address. Writable from internal bus interface.
2-1	TYPE	R	0h	For memory BARs, they determine the BAR type. Writable from internal bus interface. • 0 = 32-bit BAR. • 2h = 64-bit BAR. • Others = Reserved. For I/O BARs, bit 2 is the least significant bit (LSB) of the base address and bit 1 is 0. Writable from internal bus interface.
0	MEM_SPACE	R	0h	Writable from internal bus interface. • 0 = Memory BAR. • 1 = I/O BAR.

Table 11-2962. Register Call Summary for PCIE_BAR2

Configuration Type 0 Registers
- PCIE_BAR2_MASK Register (Offset = 1018h) [reset = 0h]: [0][1][2][3][4]
- PCIE_BAR3 (64 bit BAR2) Register (Offset = 101Ch) [reset = 0h]: [0][1][2]
- Register Summary: [0]
- PCIE_BAR3_MASK (64 bit BAR2) Register (Offset = 101Ch) [reset = 0h]: [0][1][2]
- PCIE_BAR2 Register (Offset = 1018h) [reset = 8h]: [0]

Table 11-2962. Register Call Summary for PCIE_BAR2 (continued)

PCIe SS Functional Description
• Mapping Multiple Non-Contiguous Memory Ranges To One Region: [0][1] • Inbound Address Translation: [0][1][2] • Inbound Address Translation: [0]

11.14.5.5.10 PCIE_BAR2_MASK Register (Offset = 1018h) [reset = 0h]

The Register (PCIE_BAR2_MASK) is shown in Figure 11-1229 and described in Table 11-2964. The offset is 1018h (same as PCIE_BAR2, but requires DBI_CS2 bit set in PCIE_CMD_STATUS register, see for details).

Table 11-2963. PCIE_BAR2_MASK Instances

Instance	Physical Address
PCIE	2180 1018h

Figure 11-1229. PCIE_BAR2_MASK Register

31	30	29	28	27	26	25	24
BAR_MASK							
W-0h							
23	22	21	20	19	18	17	16
BAR_MASK							
W-0h							
15	14	13	12	11	10	9	8
BAR_MASK							
W-0h							
7	6	5	4	3	2	1	0
BAR_MASK							BAR_ENABLED
W-0h							W-0h

LEGEND: W = Write Only; *-n* = value after reset

Table 11-2964. PCIE_BAR2_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	BAR_MASK	W	0h	Indicates which PCIE_BAR2 bits to mask (non-writeable) from host, which determines the size of the BAR. The bits are writeable only, not readable.
0	BAR_ENABLED	W	0h	PCIE_BAR2 Enable. The bit is writeable only, not readable. • 0 = PCIE_BAR2 is disabled. • 1 = PCIE_BAR2 is enabled.

Table 11-2965. Register Call Summary for PCIE_BAR2_MASK

Configuration Type 0 Registers
- PCIE_BAR2_MASK Register (Offset = 1018h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.5.11 PCIE_BAR3 Register (Offset = 101Ch) [reset = 8h]

The Base Address Register 3 (PCIE_BAR3) is shown in Figure 11-1230 and described in Table 11-2967.

Table 11-2966. PCIE_BAR3 Instances

Instance	Physical Address
PCIE	2180 101Ch

Figure 11-1230. PCIE_BAR3 Register

31	30	29	28	27	26	25	24
BASE_ADDR							
R/W-0h							
23	22	21	20	19	18	17	16
BASE_ADDR							
R/W-0h							
15	14	13	12	11	10	9	8
BASE_ADDR							
R/W-0h							
7	6	5	4	3	2	1	0
BASE_ADDR				PREFETCH	TYPE		MEM_SPACE
R/W-0h				R-1h	R-0h		R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2967. PCIE_BAR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	BASE_ADDR	R/W	0h	Base Address. Based on the configuration of Register, not all bits may be writable.
3	PREFETCH	R	1h	For memory BARs, it indicates whether the region is prefetchable. Writable from internal bus interface. • 0 = Non-prefetchable. • 1= Prefetchable. For I/O Bars, it is used as second least significant bit (LSB) of the base address. Writable from internal bus interface.
2-1	TYPE	R	0h	For memory BARs, they determine the BAR type. Writable from internal bus interface. • 00 = 32-bit BAR. • 10 = 64-bit BAR. • Others = Reserved. For I/O BARs, bit 2 is the least significant bit (LSB) of the base address and bit 1 is 0. Writable from internal bus interface.
0	MEM_SPACE	R	0h	Writable from internal bus interface. • 0 = Memory BAR. • 1 = I/O BAR.

Table 11-2968. Register Call Summary for PCIE_BAR3

Configuration Type 0 Registers
- PCIE_BAR3_MASK Register (Offset = 101Ch) [reset = 0h]: [0][1][2][3][4]
- Register Summary: [0][1]
- PCIE_BAR3 (64 bit BAR2) Register (Offset = 101Ch) [reset = 0h]: [0]
- PCIE_BAR3_MASK (64 bit BAR2) Register (Offset = 101Ch) [reset = 0h]: [0]
- PCIE_BAR3 Register (Offset = 101Ch) [reset = 8h]: [0]

Table 11-2968. Register Call Summary for PCIE_BAR3 (continued)

PCIe SS Functional Description
• Inbound Address Translation: [0][1] • Inbound Address Translation: [0]

11.14.5.5.12 PCIE_BAR3_MASK Register (Offset = 101Ch) [reset = 0h]

The Register (PCIE_BAR3_MASK) is shown in Figure 11-1231 and described in Table 11-2970. The offset is 101Ch (same as PCIE_BAR3, but requires DBI_CS2 bit set in PCIE_CMD_STATUS register, see for details).

Table 11-2969. PCIE_BAR3_MASK Instances

Instance	Physical Address
PCIE	2180 101Ch

Figure 11-1231. PCIE_BAR3_MASK Register

31	30	29	28	27	26	25	24
BAR_MASK							
W-0h							
23	22	21	20	19	18	17	16
BAR_MASK							
W-0h							
15	14	13	12	11	10	9	8
BAR_MASK							
W-0h							
7	6	5	4	3	2	1	0
BAR_MASK							BAR_ENABLED
W-0h							W-0h

LEGEND: W = Write Only; *-n* = value after reset

Table 11-2970. PCIE_BAR3_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	BAR_MASK	W	0h	Indicates which PCIE_BAR3 bits to mask (non-writeable) from host, which determines the size of the BAR. The bits are writeable only, not readable.
0	BAR_ENABLED	W	0h	PCIE_BAR3 Enable. The bit is writeable only, not readable. • 0 = PCIE_BAR3 is disabled. • 1 = PCIE_BAR3 is enabled.

Table 11-2971. Register Call Summary for PCIE_BAR3_MASK

Configuration Type 0 Registers
- PCIE_BAR3_MASK (64 bit BAR2) Register (Offset = 101Ch) [reset = 0h]: [0]
- Register Summary: [0][1]
- PCIE_BAR3_MASK Register (Offset = 101Ch) [reset = 0h]: [0]

11.14.5.5.13 PCIE_BAR3 (64 bit BAR2) Register (Offset = 101Ch) [reset = 0h]

The Base Address Register 3 (PCIE_BAR3) (64bit PCIE_BAR2) is shown in Figure 11-1232 and described in Table 11-2973.

Table 11-2972. PCIE_BAR3 (64 bit BAR2) Instances

Instance	Physical Address
PCIE	2180 101Ch

Figure 11-1232. PCIE_BAR3 (64 bit BAR2) Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2973. PCIE_BAR3 (64 bit BAR2) Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BASE_ADDR	R/W	0h	Upper 32 bits of PCIE_BAR2 address if PCIE_BAR2 is a 64-bit BAR. Based on the configuration of Register, not all bits may be writable.

Table 11-2974. Register Call Summary for PCIE_BAR3

Configuration Type 0 Registers • PCIE_BAR3_MASK Register (Offset = 101Ch) [reset = 0h]: [0][1][2][3][4] • Register Summary: [0][1] • PCIE_BAR3 (64 bit BAR2) Register (Offset = 101Ch) [reset = 0h]: [0] • PCIE_BAR3_MASK (64 bit BAR2) Register (Offset = 101Ch) [reset = 0h]: [0] • PCIE_BAR3 Register (Offset = 101Ch) [reset = 8h]: [0]
PCIe SS Functional Description • Inbound Address Translation: [0][1] • Inbound Address Translation: [0]

11.14.5.5.14 PCIE_BAR3_MASK (64 bit BAR2) Register (Offset = 101Ch) [reset = 0h]

The Register (PCIE_BAR3_MASK) (64bit PCIE_BAR2) is shown in Figure 11-1233 and described in Table 11-2976. The offset is 101Ch (same as PCIE_BAR3, but requires DBI_CS2 bit set in PCIE_CMD_STATUS register, see for details).

Table 11-2975. PCIE_BAR3_MASK (64 bit BAR2) Instances

Instance	Physical Address
PCIE	2180 101Ch

Figure 11-1233. PCIE_BAR3_MASK (64 bit BAR2) Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BAR_MASK																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 11-2976. PCIE_BAR3_MASK (64 bit BAR2) Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BAR_MASK	W	0h	Upper 32 bits of PCIE_BAR2 mask value if PCIE_BAR2 is a 64-bit BAR. Indicates which BAR bits to mask (non-writeable) from host, which determines the size of the BAR. The bits are writeable only, not readable.

Table 11-2977. Register Call Summary for PCIE_BAR3_MASK

Configuration Type 0 Registers

- PCIE_BAR3_MASK (64 bit BAR2) Register (Offset = 101Ch) [reset = 0h]: [0]
- Register Summary: [0][1]
- PCIE_BAR3_MASK Register (Offset = 101Ch) [reset = 0h]: [0]

11.14.5.5.15 PCIE_BAR4 Register (Offset = 1020h) [reset = 8h]

The Base Address Register 4 (PCIE_BAR4) is shown in Figure 11-1234 and described in Table 11-2979.

Table 11-2978. PCIE_BAR4 Instances

Instance	Physical Address
PCIE	2180 1020h

Figure 11-1234. PCIE_BAR4 Register

31	30	29	28	27	26	25	24
BASE_ADDR							
R/W-0h							
23	22	21	20	19	18	17	16
BASE_ADDR							
R/W-0h							
15	14	13	12	11	10	9	8
BASE_ADDR							
R/W-0h							
7	6	5	4	3	2	1	0
BASE_ADDR				PREFETCH	TYPE		MEM_SPACE
R/W-0h				R-1h	R-0h		R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2979. PCIE_BAR4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	BASE_ADDR	R/W	0h	Base Address. Based on the configuration of Register, not all bits may be writable.
3	PREFETCH	R	1h	For memory BARs, it indicates whether the region is prefetchable. Writable from internal bus interface. • 0 = Non-prefetchable. • 1= Prefetchable. For I/O Bars, it is used as second least significant bit (LSB) of the base address. Writable from internal bus interface.
2-1	TYPE	R	0h	For memory BARs, they determine the BAR type. Writable from internal bus interface. • 0 = 32-bit BAR. • 2h = 64-bit BAR. • Others = Reserved. For I/O BARs, bit 2 is the least significant bit (LSB) of the base address and bit 1 is 0. Writable from internal bus interface.
0	MEM_SPACE	R	0h	Writable from internal bus interface. • 0 = Memory BAR. • 1 = I/O BAR.

Table 11-2980. Register Call Summary for PCIE_BAR4

Configuration Type 0 Registers • PCIE_BAR4 Register (Offset = 1020h) [reset = 8h]: [0] • PCIE_BAR5 (64 bit BAR4) Register (Offset = 1024h) [reset = 0h]: [0][1][2] • PCIE_BAR4_MASK Register (Offset = 1020h) [reset = 0h]: [0][1][2][3][4] • Register Summary: [0][1][2][3][4] • PCIE_BAR5_MASK (64 bit BAR4) Register (Offset = 1024h) [reset = 0h]: [0][1][2]
PCIe SS Functional Description • Inbound Address Translation: [0]

11.14.5.5.16 PCIE_BAR4_MASK Register (Offset = 1020h) [reset = 0h]

The Register (PCIE_BAR4_MASK) is shown in Figure 11-1235 and described in Table 11-2982. The offset is 1020h (same as PCIE_BAR4, but requires DBI_CS2 bit set in PCIE_CMD_STATUS register, see for details).

Table 11-2981. PCIE_BAR4_MASK Instances

Instance	Physical Address
PCIE	2180 1020h

Figure 11-1235. PCIE_BAR4_MASK Register

31	30	29	28	27	26	25	24
BAR_MASK							
W-0h							
23	22	21	20	19	18	17	16
BAR_MASK							
W-0h							
15	14	13	12	11	10	9	8
BAR_MASK							
W-0h							
7	6	5	4	3	2	1	0
BAR_MASK							BAR_ENABLED
W-0h							W-0h

LEGEND: W = Write Only; *-n* = value after reset

Table 11-2982. PCIE_BAR4_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	BAR_MASK	W	0h	Indicates which PCIE_BAR4 bits to mask (non-writeable) from host, which determines the size of the BAR. The bits are writeable only, not readable.
0	BAR_ENABLED	W	0h	PCIE_BAR4 Enable. The bit is writeable only, not readable. • 0 = PCIE_BAR4 is disabled. • 1 = PCIE_BAR4 is enabled.

Table 11-2983. Register Call Summary for PCIE_BAR4_MASK

Configuration Type 0 Registers
- PCIE_BAR4_MASK Register (Offset = 1020h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.5.17 PCIE_BAR5 Register (Offset = 1024h) [reset = 8h]

The Base Address Register 5 (PCIE_BAR5) is shown in Figure 11-1236 and described in Table 11-2985.

Table 11-2984. PCIE_BAR5 Instances

Instance	Physical Address
PCIE	2180 1024h

Figure 11-1236. PCIE_BAR5 Register

31	30	29	28	27	26	25	24
BASE_ADDR							
R/W-0h							
23	22	21	20	19	18	17	16
BASE_ADDR							
R/W-0h							
15	14	13	12	11	10	9	8
BASE_ADDR							
R/W-0h							
7	6	5	4	3	2	1	0
BASE_ADDR				PREFETCH	TYPE		MEM_SPACE
R/W-0h				R-1h	R-0h		R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-2985. PCIE_BAR5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	BASE_ADDR	R/W	0h	Base Address. Based on the configuration of Register, not all bits may be writable.
3	PREFETCH	R	1h	For memory BARs, it indicates whether the region is prefetchable. Writable from internal bus interface. • 0 = Non-prefetchable. • 1= Prefetchable. For I/O Bars, it is used as second least significant bit (LSB) of the base address. Writable from internal bus interface.
2-1	TYPE	R	0h	For memory BARs, they determine the BAR type. Writable from internal bus interface. • 0 = 32-bit BAR. • 2h = 64-bit BAR. • Others = Reserved. For I/O BARs, bit 2 is the least significant bit (LSB) of the base address and bit 1 is 0. Writable from internal bus interface.
0	MEM_SPACE	R	0h	Writable from internal bus interface. • 0 = Memory BAR. • 1 = I/O BAR.

Table 11-2986. Register Call Summary for PCIE_BAR5

Configuration Type 0 Registers • PCIE_BAR5 Register (Offset = 1024h) [reset = 8h]: [0] • Register Summary: [0][1] • PCIE_BAR5_MASK Register (Offset = 1024h) [reset = 0h]: [0][1][2][3][4] • PCIE_BAR5_MASK (64 bit BAR4) Register (Offset = 1024h) [reset = 0h]: [0] • PCIE_BAR5 (64 bit BAR4) Register (Offset = 1024h) [reset = 0h]: [0]
PCIe SS Functional Description • Inbound Address Translation: [0][1][2][3]

11.14.5.5.18 PCIE_BAR5_MASK Register (Offset = 1024h) [reset = 0h]

The Register (PCIE_BAR5_MASK) is shown in Figure 11-1237 and described in Table 11-2988. The offset is 1024h (same as PCIE_BAR5, but requires DBI_CS2 bit set in PCIE_CMD_STATUS register, see for details).

Table 11-2987. PCIE_BAR5_MASK Instances

Instance	Physical Address
PCIE	2180 1024h

Figure 11-1237. PCIE_BAR5_MASK Register

31	30	29	28	27	26	25	24
BAR_MASK							
W-0h							
23	22	21	20	19	18	17	16
BAR_MASK							
W-0h							
15	14	13	12	11	10	9	8
BAR_MASK							
W-0h							
7	6	5	4	3	2	1	0
BAR_MASK							BAR_ENABLED
W-0h							W-0h

LEGEND: W = Write Only; *-n* = value after reset

Table 11-2988. PCIE_BAR5_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	BAR_MASK	W	0h	Indicates which PCIE_BAR5 bits to mask (non-writeable) from host, which determines the size of the BAR. The bits are writeable only, not readable.
0	BAR_ENABLED	W	0h	PCIE_BAR5 Enable. The bit is writeable only, not readable. • 0 = PCIE_BAR5 is disabled. • 1 = PCIE_BAR5 is enabled.

Table 11-2989. Register Call Summary for PCIE_BAR5_MASK

Configuration Type 0 Registers
- PCIE_BAR5_MASK (64 bit BAR4) Register (Offset = 1024h) [reset = 0h]: [0]
- Register Summary: [0][1]
- PCIE_BAR5_MASK Register (Offset = 1024h) [reset = 0h]: [0]

11.14.5.5.19 PCIE_BAR5 (64 bit BAR4) Register (Offset = 1024h) [reset = 0h]

The Base Address Register 5 (PCIE_BAR5) (64bit PCIE_BAR4) is shown in Figure 11-1238 and described in Table 11-2991.

Table 11-2990. PCIE_BAR5 (64 bit BAR4) Instances

Instance	Physical Address
PCIE	2180 1024h

Figure 11-1238. PCIE_BAR5 (64 bit BAR4) Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-2991. PCIE_BAR5 (64 bit BAR4) Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BASE_ADDR	R/W	0h	Upper 32 bits of PCIE_BAR4 address if PCIE_BAR4 is a 64-bit BAR. Based on the configuration of Register, not all bits may be writable.

Table 11-2992. Register Call Summary for PCIE_BAR5

Configuration Type 0 Registers • PCIE_BAR5 Register (Offset = 1024h) [reset = 8h]: [0] • Register Summary: [0][1] • PCIE_BAR5_MASK Register (Offset = 1024h) [reset = 0h]: [0][1][2][3][4] • PCIE_BAR5_MASK (64 bit BAR4) Register (Offset = 1024h) [reset = 0h]: [0] • PCIE_BAR5 (64 bit BAR4) Register (Offset = 1024h) [reset = 0h]: [0]
PCIe SS Functional Description • Inbound Address Translation: [0][1][2][3]

11.14.5.5.20 PCIE_BAR5_MASK (64 bit BAR4) Register (Offset = 1024h) [reset = 0h]

The Register (PCIE_BAR5_MASK) (64bit PCIE_BAR4) is shown in Figure 11-1239 and described in Table 11-2994. The offset is 1024h (same as PCIE_BAR5, but requires DBI_CS2 bit set in PCIE_CMD_STATUS register, see for details).

Table 11-2993. PCIE_BAR5_MASK (64 bit BAR4) Instances

Instance	Physical Address
PCIE	2180 1024h

Figure 11-1239. PCIE_BAR5_MASK (64 bit BAR4) Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BAR_MASK																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 11-2994. PCIE_BAR5_MASK (64 bit BAR4) Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BAR_MASK	W	0h	Upper 32 bits of PCIE_BAR4 mask value if PCIE_BAR4 is a 64-bit BAR. Indicates which BAR bits to mask (non-writeable) from host, which determines the size of the BAR. The bits are writeable only, not readable.

Table 11-2995. Register Call Summary for PCIE_BAR5_MASK

Configuration Type 0 Registers

- PCIE_BAR5_MASK (64 bit BAR4) Register (Offset = 1024h) [reset = 0h]: [0]
- Register Summary: [0][1]
- PCIE_BAR5_MASK Register (Offset = 1024h) [reset = 0h]: [0]

11.14.5.5.21 PCIE_SUBSYS_VNDR_ID Register (Offset = 102Ch) [reset = 10000h]

The Subsystem and Subsystem Vendor ID (PCIE_SUBSYS_VNDR_ID) is shown in Figure 11-1240 and described in Table 11-2997.

Table 11-2996. PCIE_SUBSYS_VNDR_ID Instances

Instance	Physical Address
PCIE	2180 102Ch

Figure 11-1240. PCIE_SUBSYS_VNDR_ID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R -10000h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-2997. PCIE_SUBSYS_VNDR_ID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	10000h	TI internal data. Identifies revision of peripheral.

Table 11-2998. Register Call Summary for PCIE_SUBSYS_VNDR_ID

Configuration Type 0 Registers
- PCIE_SUBSYS_VNDR_ID Register (Offset = 102Ch) [reset = 10000h]: [0]
- Register Summary: [0]

11.14.5.5.22 PCIE_EXPNSN_ROM Register (Offset = 1030h) [reset = 0h]

The Expansion ROM Base Address (PCIE_EXPNSN_ROM) is shown in Figure 11-1241 and described in Table 11-3000.

Table 11-2999. PCIE_EXPNSN_ROM Instances

Instance	Physical Address
PCIE	2180 1030h

Figure 11-1241. PCIE_EXPNSN_ROM Register

31	30	29	28	27	26	25	24
EXP_ROM_BASE_ADDR							
R-0h							
23	22	21	20	19	18	17	16
EXP_ROM_BASE_ADDR							
R-0h							
15	14	13	12	11	10	9	8
EXP_ROM_BASE_ADDR					RESERVED		
R-0h					R-0h		
7	6	5	4	3	2	1	0
RESERVED							EXP_ROM_EN
R-0h							R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3000. PCIE_EXPNSN_ROM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	EXP_ROM_BASE_ADDR	R	0h	Address of Expansion ROM
10-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	EXP_ROM_EN	R	0h	Expansion ROM Enable

Table 11-3001. Register Call Summary for PCIE_EXPNSN_ROM

Configuration Type 0 Registers • PCIE_EXPNSN_ROM Register (Offset = 1030h) [reset = 0h]: [0] • Register Summary: [0]
Configuration Type 1 Registers • PCIE_EXPNSN_ROM Register (Offset = 1038h) [reset = 0h]: [0]

11.14.5.5.23 PCIE_CAP_PTR Register (Offset = 1034h) [reset = 40h]

The Capabilities Pointer (PCIE_CAP_PTR) is shown in Figure 11-1242 and described in Table 11-3003.

Table 11-3002. PCIE_CAP_PTR Instances

Instance	Physical Address
PCIE	2180 1034h

Figure 11-1242. PCIE_CAP_PTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CAP_PTR							
R-0h																								R-40h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3003. PCIE_CAP_PTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reads return 0 and writes have no effect.
7-0	CAP_PTR	R	40h	First Capability Pointer. By default, it points to Power Management Capability structure. Writable from internal bus interface.

Table 11-3004. Register Call Summary for PCIE_CAP_PTR

Configuration Type 0 Registers • Register Summary: [0] • PCIE_CAP_PTR Register (Offset = 1034h) [reset = 40h]: [0]
Configuration Type 1 Registers • PCIE_CAP_PTR Register (Offset = 1034h) [reset = 40h]: [0]

11.14.5.5.24 PCIE_INT_PIN Register (Offset = 103Ch) [reset = 1FFh]

The Interrupt Pin Register (PCIE_INT_PIN) is shown in Figure 11-1243 and described in Table 11-3006.

Table 11-3005. PCIE_INT_PIN Instances

Instance	Physical Address
PCIE	2180 103Ch

Figure 11-1243. PCIE_INT_PIN Register

31	30	29	28	27	26	25	24
MAX_LATENCY							
R-0h							
23	22	21	20	19	18	17	16
MIN_GRANT							
R-0h							
15	14	13	12	11	10	9	8
INT_PIN							
R-1h							
7	6	5	4	3	2	1	0
INT_LINE							
R/W-FFh							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3006. PCIE_INT_PIN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	MAX_LATENCY	R	0h	Not applicable to PCI Express.
23-16	MIN_GRANT	R	0h	Not applicable to PCI Express.
15-8	INT_PIN	R	1h	Interrupt Pin. It identifies the legacy interrupt message that the device uses. For single function configuration, the core only uses INTA. This register is writable through internal bus interface. • 0 = Legacy interrupt is not being used • 1h = INTA • 2h = INTB • 3h = INTC • 4h = INTD • Others = Reserved
7-0	INT_LINE	R/W	FFh	Interrupt Line. Value is system software specified.

Table 11-3007. Register Call Summary for PCIE_INT_PIN

Configuration Type 0 Registers
- Register Summary: [0]
- PCIE_INT_PIN Register (Offset = 103Ch) [reset = 1FFh]: [0]

11.14.5.6 Configuration Type 1 Registers

11.14.5.6.1 Register Summary

Table 11-3008. PCIe Instances

Instance	Base Address
PCIE	2180 0000h

Table 11-3009. Configuration Type 1 Registers

Offset	Acronym	Register Name	PCIE Physical Address	Section
100Ch	PCIE _BIST_HEADER	BIST, Header Type, Latency Time and Cache Line Size Register	2180 100Ch	Section 11.14.5.6.2
1010h	PCIE _BAR0	Base Address Register 0	2180 1010h	Section 11.14.5.6.3
1010h	PCIE _BAR0_MASK	Register	2180 1010h	Section 11.14.5.6.4
1014h	PCIE _BAR1	Base Address Register 1	2180 1014h	Section 11.14.5.6.5
1014h	PCIE _BAR1_MASK	Register	2180 1014h	Section 11.14.5.6.6
1014h	PCIE _BAR1 (64 bit PCIE _BAR0)	Base Address Register 1 (64 bit PCIE _BAR0)	2180 1014h	Section 11.14.5.6.7
1014h	PCIE _BAR1_MASK (64 bit PCIE _BAR0)	Register1 (64 bit PCIE _BAR0)	2180 1014h	Section 11.14.5.6.8
1018h	PCIE_BUSNUM	Latency Timer and Bus Number Register	2180 1018h	Section 11.14.5.6.9
101Ch	PCIE_SECSTAT	Secondary Status and IO Space Register	2180 101Ch	Section 11.14.5.6.10
1020h	PCIE_MEMSPACE	Memory Limit and Base Register	2180 1020h	Section 11.14.5.6.11
1024h	PCIE_PREFETCH_MEM	Prefetchable Memory Limit and Base Register	2180 1024h	Section 11.14.5.6.12
1028h	PCIE_PREFETCH_BASE	Prefetchable Memory Base Upper 32 bits register	2180 1028h	Section 11.14.5.6.13
102Ch	PCIE_PREFETCH_LIMIT	Prefetchable Limit Upper 32 bits register	2180 102Ch	Section 11.14.5.6.14
1030h	PCIE_IOSPACE	IO Base and Limit Upper 16 bits register	2180 1030h	Section 11.14.5.6.15
1034h	PCIE _CAP_PTR	Capabilities Pointer	2180 1034h	Section 11.14.5.6.16
1038h	PCIE _EXPNSN_ROM	Expansion ROM Base Address	2180 1038h	Section 11.14.5.6.17
103Ch	PCIE_BRIDGE_INT	Bridge Control and Interrupt Register	2180 103Ch	Section 11.14.5.6.18

11.14.5.6.2 PCIE_BIST_HEADER Register (Offset = 100Ch) [reset = 10000h]

The BIST, Header Type, Latency Time and Cache Line Size Register (PCIE_BIST_HEADER) is shown in Figure 11-1244 and described in Table 11-3011.

Table 11-3010. PCIE_BIST_HEADER Instances

Instance	Physical Address
PCIE	2180 100Ch

Figure 11-1244. PCIE_BIST_HEADER Register

31	30	29	28	27	26	25	24
BIST_CAP	START_BIST	RESERVED		COMP_CODE			
R-0h	R-0h	R-0h		R-0h			
23	22	21	20	19	18	17	16
MULFUN_DEV	HDR_TYPE						
R-0h	R-1h						
15	14	13	12	11	10	9	8
LAT_TMR							
R-0h							
7	6	5	4	3	2	1	0
CACHELN_SIZE							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3011. PCIE_BIST_HEADER Register Field Descriptions

Bit	Field	Type	Reset	Description
31	BIST_CAP	R	0h	Returns a 1 for BIST capability and 0 otherwise. Not supported by PCIESS.
30	START_BIST	R	0h	Write a one to start BIST. Not supported by PCIESS.
29-28	RESERVED	R	0h	Reads return 0 and writes have no effect.
27-24	COMP_CODE	R	0h	Completion Code. Not supported by PCIESS.
23	MULFUN_DEV	R	0h	Returns 1 if it is a multi-function device. Writable from internal bus interface.
22-16	HDR_TYPE	R	1h	Configuration Header Format. • 0 = EP mode • 1 = RC mode
15-8	LAT_TMR	R	0h	Not applicable in PCIe
7-0	CACHELN_SIZE	R	0h	Not applicable in PCIe

Table 11-3012. Register Call Summary for PCIE_BIST_HEADER

Configuration Type 0 Registers • PCIE_BIST_HEADER Register (Offset = 100Ch) [reset = 0h]: [0] • Register Summary: [0]
Configuration Type 1 Registers • PCIE_BIST_HEADER Register (Offset = 100Ch) [reset = 10000h]: [0]

11.14.5.6.3 PCIE_BAR0 Register (Offset = 1010h) [reset = 0h]

The Base Address Register 0 (PCIE_BAR0) is shown in Figure 11-1245 and described in Table 11-3014.

Table 11-3013. PCIE_BAR0 Instances

Instance	Physical Address
PCIE	2180 1010h

Figure 11-1245. PCIE_BAR0 Register

31	30	29	28	27	26	25	24
BASE_ADDR							
R-0h							
23	22	21	20	19	18	17	16
BASE_ADDR							
R-0h							
15	14	13	12	11	10	9	8
BASE_ADDR							
R-0h							
7	6	5	4	3	2	1	0
BASE_ADDR				PREFETCH	TYPE		MEM_SPACE
R-0h				R-0h	R-0h		R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3014. PCIE_BAR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	BASE_ADDR	R	0h	Base Address. Actual writable bits are determined by Register
3	PREFETCH	R	0h	For memory BARs, it indicates whether the region is prefetchable. Writable from internal bus interface. • 0 = Non-prefetchable. • 1= Prefetchable. For I/O Bars, it is used as second least significant bit (LSB) of the base address. Writable from internal bus interface.
2-1	TYPE	R	0h	For memory BARs, they determine the BAR type. Writable from internal bus interface. • 0 = 32-bit BAR. • 2h = 64-bit BAR. • Others = Reserved • For I/O BARs, bit 2 is the least significant bit (LSB) of the base address and bit 1 is 0. Writable from internal bus interface.
0	MEM_SPACE	R	0h	Writable from internal bus interface. • 0 = Memory BAR. • 1 = I/O BAR.

Table 11-3015. Register Call Summary for PCIE_BAR0

Configuration Type 0 Registers • PCIE_BAR0 Register (Offset = 1010h) [reset = 0h]: [0] • Register Summary: [0][1][2][3][4] • PCIE_BAR1 (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0][1][2] • PCIE_BAR0_MASK Register (Offset = 1010h) [reset = 0h]: [0][1][2][3][4] • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0][1][2]
Application Registers • PCIE_MSI_IRQ Register (Offset = 54h) [reset = 0h]: [0]

Table 11-3015. Register Call Summary for PCIE_BAR0 (continued)

PCIe SS Registers • Application Registers: [0]
PCIe SS Functional Description • PHY Loopback: [0] • PCIE_IB_BAR0 Exception for In-Bound Address Translation: [0][1][2][3][4] • Inbound Address Translation: [0][1][2][3][4][5][6][7][8] • MSI Interrupt Generation in EP Mode: [0] • Interrupt Reception in EP Mode: [0] • Address Space 0: [0] • BAR Mask Registers: [0][1][2][3][4][5][6]
Configuration Type 1 Registers • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0][1][2] • PCIE_BAR0 Register (Offset = 1010h) [reset = 0h]: [0] • PCIE_BAR0_MASK Register (Offset = 1010h) [reset = 0h]: [0][1][2][3][4] • PCIE_BAR1 (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0][1][2]

11.14.5.6.4 PCIE_BAR0_MASK Register (Offset = 1010h) [reset = 0h]

The Register(PCIE_BAR0_MASK) is shown in Figure 11-1246 and described in Table 11-3017. The offset is 1010h (same as PCIE_BAR0, but requires DBI_CS2 bit set in PCIE_CMD_STATUS register, see for details).

Table 11-3016. PCIE_BAR0_MASK Instances

Instance	Physical Address
PCIE	2180 1010h

Figure 11-1246. PCIE_BAR0_MASK Register

31	30	29	28	27	26	25	24
BAR_MASK							
W-0h							
23	22	21	20	19	18	17	16
BAR_MASK							
W-0h							
15	14	13	12	11	10	9	8
BAR_MASK							
W-0h							
7	6	5	4	3	2	1	0
BAR_MASK							BAR_ENABLED
W-0h							W-0h

LEGEND: W = Write Only; *-n* = value after reset

Table 11-3017. PCIE_BAR0_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	BAR_MASK	W	0h	Indicates which PCIE_BAR0 bits to mask (non-writeable) from host, which determines the size of the BAR. The bits are writeable only, not readable.
0	BAR_ENABLED	W	0h	PCIE_BAR0 Enable. The bit is writeable only, not readable. • 0 = PCIE_BAR0 is disabled. • 1 = PCIE_BAR0 is enabled.

Table 11-3018. Register Call Summary for PCIE_BAR0_MASK

Configuration Type 0 Registers • Register Summary: [0] • PCIE_BAR0_MASK Register (Offset = 1010h) [reset = 0h]: [0]
PCIe SS Functional Description • BAR Mask Registers: [0]
Configuration Type 1 Registers • PCIE_BAR0_MASK Register (Offset = 1010h) [reset = 0h]: [0]

11.14.5.6.5 PCIE_BAR1 Register (Offset = 1014h) [reset = 8h]

The Base Address Register 1 (PCIE_BAR1) is shown in Figure 11-1247 and described in Table 11-3020.

Table 11-3019. PCIE_BAR1 Instances

Instance	Physical Address
PCIE	2180 1014h

Figure 11-1247. PCIE_BAR1 Register

31	30	29	28	27	26	25	24
BASE_ADDR							
R-0h							
23	22	21	20	19	18	17	16
BASE_ADDR							
R-0h							
15	14	13	12	11	10	9	8
BASE_ADDR							
R-0h							
7	6	5	4	3	2	1	0
BASE_ADDR				PREFETCH	TYPE		MEM_SPACE
R-0h				R-1h	R-0h		R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3020. PCIE_BAR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	BASE_ADDR	R	0h	Base Address. Actual writable bits are determined by Register
3	PREFETCH	R	1h	For memory BARs, it indicates whether the region is prefetchable. Writable from internal bus interface. • 0 = Non-prefetchable. • 1= Prefetchable. For I/O Bars, it is used as second least significant bit (LSB) of the base address. Writable from internal bus interface.
2-1	TYPE	R	0h	For memory BARs, they determine the BAR type. Writable from internal bus interface. • 0 = 32-bit BAR. • 2h = 64-bit BAR. • Others = Reserved. For I/O BARs, bit 2 is the least significant bit (LSB) of the base address and bit 1 is 0. Writable from internal bus interface.
0	MEM_SPACE	R	0h	Writable from internal bus interface. • 0 = Memory BAR. • 1 = I/O BAR.

Table 11-3021. Register Call Summary for PCIE_BAR1

Configuration Type 0 Registers
- PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0][1][2][3][4]
- Register Summary: [0][1][2][3][4][5]
- PCIE_BAR1 Register (Offset = 1014h) [reset = 8h]: [0]
- PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0]
- PCIE_BAR1 (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0]

Table 11-3021. Register Call Summary for PCIE_BAR1 (continued)

PCIe SS Functional Description • Using PCIE_IB_BAR1 Value As Start Address: [0][1][2][3][4][5][6] • PHY Loopback: [0] • Mapping Multiple Non-Contiguous Memory Ranges To One Region: [0][1] • Inbound Address Translation: [0][1][2][3][4][5][6][7]
Configuration Type 1 Registers • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0][1][2][3][4] • PCIE_BAR1 (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR1 Register (Offset = 1014h) [reset = 8h]: [0]

11.14.5.6.6 PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]

The Register (PCIE_BAR1_MASK) is shown in Figure 11-1248 and described in Table 11-3023. The offset is 1014h (same as PCIE_BAR1, but requires DBI_CS2 bit set in PCIE_CMD_STATUS register, see for details).

Table 11-3022. PCIE_BAR1_MASK Instances

Instance	Physical Address
PCIE	2180 1014h

Figure 11-1248. PCIE_BAR1_MASK Register

31	30	29	28	27	26	25	24
BAR_MASK							
W-0h							
23	22	21	20	19	18	17	16
BAR_MASK							
W-0h							
15	14	13	12	11	10	9	8
BAR_MASK							
W-0h							
7	6	5	4	3	2	1	0
BAR_MASK							BAR_ENABLED
W-0h							W-0h

LEGEND: W = Write Only; *-n* = value after reset

Table 11-3023. PCIE_BAR1_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	BAR_MASK	W	0h	Indicates which PCIE_BAR1 bits to mask (non-writeable) from host, which determines the size of the BAR. The bits are writeable only, not readable.
0	BAR_ENABLED	W	0h	PCIE_BAR1 Enable. The bit is writeable only, not readable. • 0 = PCIE_BAR1 is disabled. • 1 = PCIE_BAR1 is enabled.

Table 11-3024. Register Call Summary for PCIE_BAR1_MASK

Configuration Type 0 Registers • PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0] • Register Summary: [0][1] • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0]
Configuration Type 1 Registers • PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0]

11.14.5.6.7 PCIE_BAR1 (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]

The Base Address Register 1 (PCIE_BAR1) (64bit PCIE_BAR0) is shown in Figure 11-1249 and described in Table 11-3026.

Table 11-3025. PCIE_BAR1 (64 bit BAR0) Instances

Instance	Physical Address
PCIE	2180 1014h

Figure 11-1249. PCIE_BAR1 (64 bit BAR0) Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3026. PCIE_BAR1 (64 bit BAR0) Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BASE_ADDR	R	0h	Upper 32 bits of PCIE_BAR0 address if PCIE_BAR0 is a 64-bit BAR. Based on the configuration of Register, not all bits may be writable.

Table 11-3027. Register Call Summary for PCIE_BAR1

Configuration Type 0 Registers • PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0][1][2][3][4] • Register Summary: [0][1][2][3][4][5] • PCIE_BAR1 Register (Offset = 1014h) [reset = 8h]: [0] • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR1 (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0]
PCIe SS Functional Description • Using PCIE_IB_BAR1 Value As Start Address: [0][1][2][3][4][5][6] • PHY Loopback: [0] • Mapping Multiple Non-Contiguous Memory Ranges To One Region: [0][1] • Inbound Address Translation: [0][1][2][3][4][5][6][7]
Configuration Type 1 Registers • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0][1][2][3][4] • PCIE_BAR1 (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR1 Register (Offset = 1014h) [reset = 8h]: [0]

11.14.5.6.8 PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]

The Register(PCIE_BAR1_MASK) (64bit PCIE_BAR0) is shown in Figure 11-1250 and described in Table 11-3029. The offset is 1014h (same as PCIE_BAR1, but requires DBI_CS2 bit set in PCIE_CMD_STATUS register, see for details).

Table 11-3028. PCIE_BAR1_MASK (64 bit BAR0) Instances

Instance	Physical Address
PCIE	2180 1014h

Figure 11-1250. PCIE_BAR1_MASK (64 bit BAR0) Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BAR_MASK																															
W-0h																															

LEGEND: W = Write Only; *-n* = value after reset

Table 11-3029. PCIE_BAR1_MASK (64 bit BAR0) Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BAR_MASK	W	0h	Upper 32 bits of PCIE_BAR0 mask value if PCIE_BAR0 is a 64-bit BAR. Indicates which BAR bits to mask (non-writeable) from host, which determines the size of the BAR. The bits are writeable only, not readable.

Table 11-3030. Register Call Summary for PCIE_BAR1_MASK

Configuration Type 0 Registers • PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0] • Register Summary: [0][1] • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0]
Configuration Type 1 Registers • PCIE_BAR1_MASK Register (Offset = 1014h) [reset = 0h]: [0] • PCIE_BAR1_MASK (64 bit BAR0) Register (Offset = 1014h) [reset = 0h]: [0]

11.14.5.6.9 PCIE_BUSNUM Register (Offset = 1018h) [reset = 0h]

The Latency Timer and Bus Number Register (PCIE_BUSNUM) is shown in Figure 11-1251 and described in Table 11-3032.

Table 11-3031. PCIE_BUSNUM Instances

Instance	Physical Address
PCIE	2180 1018h

Figure 11-1251. PCIE_BUSNUM Register

31	30	29	28	27	26	25	24
SEC_LAT_TMR							
R-0h							
23	22	21	20	19	18	17	16
SUB_BUS_NUM							
R/W-0h							
15	14	13	12	11	10	9	8
SEC_BUS_NUM							
R/W-0h							
7	6	5	4	3	2	1	0
PRI_BUS_NUM							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3032. PCIE_BUSNUM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	SEC_LAT_TMR	R	0h	Secondary Latency Timer. Not applicable in PCI Express
23-16	SUB_BUS_NUM	R/W	0h	Subordinate Bus Number. This is highest bus number on downstream interface.
15-8	SEC_BUS_NUM	R/W	0h	Secondary Bus Number. It is typically 1h for RC.
7-0	PRI_BUS_NUM	R/W	0h	Primary Bus Number. It is 0 for RC and nonzero for switch devices only.

Table 11-3033. Register Call Summary for PCIE_BUSNUM

Configuration Type 1 Registers

- PCIE_BUSNUM Register (Offset = 1018h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.6.10 PCIE_SECSTAT Register (Offset = 101Ch) [reset = 0h]

The Secondary Status and IO Space Register (PCIE_SECSTAT) is shown in Figure 11-1252 and described in Table 11-3035.

Table 11-3034. PCIE_SECSTAT Instances

Instance	Physical Address
PCIE	2180 101Ch

Figure 11-1252. PCIE_SECSTAT Register

31	30	29	28	27	26	25	24
DTCT_PERROR	RX_SYS_ERROR	RX_MST_ABORT	RX_TGT_ABORT	TX_TGT_ABORT	RESERVED		MST_DPERR
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R-0h		R/W1C-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
IO_LIMIT				RESERVED			IO_LIMIT_ADDR
R/W-0h				R-0h			R-0h
7	6	5	4	3	2	1	0
IO_BASE				RESERVED			IO_BASE_ADDR
R/W-0h				R-0h			R-0h

LEGEND: R = Read Only; R/W = Read/Write; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-3035. PCIE_SECSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DTCT_PERROR	R/W1C	0h	Detected Parity Error. Set if the function receives a poisoned TLP.
30	RX_SYS_ERROR	R/W1C	0h	Received System Error. Set if the function receives an ERR_FATAL or ERR_NONFATAL message.
29	RX_MST_ABORT	R/W1C	0h	Received Master Abort. Set if the function receives a completion with unsupported request completion status.
28	RX_TGT_ABORT	R/W1C	0h	Received Target Abort. Set if the function receives a completion with completer abort completion status.
27	TX_TGT_ABORT	R/W1C	0h	Signaled Target Abort. Set if the function completes a posted or non-posted request as a completer abort error.
26-25	RESERVED	R	0h	Reads return 0 and writes have no effect.
24	MST_DPERR	R/W1C	0h	Master Data Parity Error. Set if the parity error enable bit is set in the Bridge Control Register and either the condition that the requester receives a poisoned completion or the condition that the requester poisons a write request is true.
23-16	RESERVED	R	0h	Reads return 0 and writes have no effect.
15-12	IO_LIMIT	R/W	0h	Upper 4 bits of 16bit IO Space Limit Address.
11-9	RESERVED	R	0h	Reads return 0 and writes have no effect.
8	IO_LIMIT_ADDR	R	0h	Indicates addressing for IO Limit Address. Writable from internal bus interface. • 0 = 16-bit IO addressing. • 1 = 32-bit IO addressing.
7-4	IO_BASE	R/W	0h	Upper 4 bits of 16bit IO Space Base Address.
3-1	RESERVED	R	0h	Reads return 0 and writes have no effect.

Table 11-3035. PCIE_SECSTAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	IO_BASE_ADDR	R	0h	Indicates addressing for the IO Base Address. Writable from internal bus interface. • 0 = 16-bit IO addressing. • 1 = 32-bit IO addressing.

Table 11-3036. Register Call Summary for PCIE_SECSTAT

Configuration Type 1 Registers
• PCIE_SECSTAT Register (Offset = 101Ch) [reset = 0h]: [0] • Register Summary: [0]

11.14.5.6.11 PCIE_MEMSPACE Register (Offset = 1020h) [reset = 0h]

The Memory Limit and Base Register (PCIE_MEMSPACE) is shown in Figure 11-1253 and described in Table 11-3038.

Table 11-3037. PCIE_MEMSPACE Instances

Instance	Physical Address
PCIE	2180 1020h

Figure 11-1253. PCIE_MEMSPACE Register

31	30	29	28	27	26	25	24
MEM_LIMIT							
R/W-0h							
23	22	21	20	19	18	17	16
MEM_LIMIT				RESERVED			
R/W-0h				R-0h			
15	14	13	12	11	10	9	8
MEM_BASE							
R/W-0h							
7	6	5	4	3	2	1	0
MEM_BASE				RESERVED			
R/W-0h				R-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3038. PCIE_MEMSPACE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	MEM_LIMIT	R/W	0h	Upper 12 bits of 32bit Memory Limit Address.
19-16	RESERVED	R	0h	Reads return 0 and writes have no effect.
15-4	MEM_BASE	R/W	0h	Upper 12 bits of 32bit Memory Base Address.
3-0	RESERVED	R	0h	Reads return 0 and writes have no effect.

Table 11-3039. Register Call Summary for PCIE_MEMSPACE

PCIe SS Functional Description • Use of Base Address Registers (BARs): [0]
Configuration Type 1 Registers • PCIE_MEMSPACE Register (Offset = 1020h) [reset = 0h]: [0] • Register Summary: [0]

11.14.5.6.12 PCIE_PREFETCH_MEM Register (Offset = 1024h) [reset = 0h]

The Prefetchable Memory Limit and Base Register (PCIE_PREFETCH_MEM) is shown in Figure 11-1254 and described in Table 11-3041.

Table 11-3040. PCIE_PREFETCH_MEM Instances

Instance	Physical Address
PCIE	2180 1024h

Figure 11-1254. PCIE_PREFETCH_MEM Register

31	30	29	28	27	26	25	24
PREFETCH_LIMIT							
R/W-0h							
23	22	21	20	19	18	17	16
PREFETCH_LIMIT				RESERVED			PRE_LIMIT_ADDR
R/W-0h				R-0h			R-0h
15	14	13	12	11	10	9	8
PREFETCH_BASE							
R-0h							
7	6	5	4	3	2	1	0
PREFETCH_BASE				RESERVED			PRE_BASE_ADDR
R-0h				R-0h			R-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3041. PCIE_PREFETCH_MEM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	PREFETCH_LIMIT	R/W	0h	Upper 12 bits of 32bit prefetchable memory limit address (end address).
19-17	RESERVED	R	0h	Reads return 0 and writes have no effect.
16	PRE_LIMIT_ADDR	R	0h	Indicates addressing for prefetchable memory limit address (end address). Writable from internal bus interface. • 0 = 32-bit memory addressing • 1 = 64-bit memory addressing
15-4	PREFETCH_BASE	R	0h	Upper 12 bits of 32bit prefetchable memory base address (start address).
3-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	PRE_BASE_ADDR	R	0h	Indicates addressing for the prefetchable memory base address (start address). Writable from internal bus interface. • 0 = 32-bit memory addressing • 1 = 64-bit memory addressing

Table 11-3042. Register Call Summary for PCIE_PREFETCH_MEM

PCIe SS Functional Description • Use of Base Address Registers (BARs): [0]
Configuration Type 1 Registers • PCIE_PREFETCH_MEM Register (Offset = 1024h) [reset = 0h]: [0] • Register Summary: [0]

11.14.5.6.13 PCIE_PREFETCH_BASE Register (Offset = 1028h) [reset = 0h]

The Prefetchable Memory Base Upper 32 bits Register (PCIE_PREFETCH_BASE) is shown in Figure 11-1255 and described in Table 11-3044.

Table 11-3043. PCIE_PREFETCH_BASE Instances

Instance	Physical Address
PCIE	2180 1028h

Figure 11-1255. PCIE_PREFETCH_BASE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PREFETCH_BASE																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3044. PCIE_PREFETCH_BASE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PREFETCH_BASE	R/W	0h	Upper 32 bits of Prefetchable Memory Base Address. Used with 64bit prefetchable memory addressing only.

Table 11-3045. Register Call Summary for PCIE_PREFETCH_BASE

PCIe SS Functional Description • Use of Base Address Registers (BARs): [0]
Configuration Type 1 Registers • PCIE_PREFETCH_BASE Register (Offset = 1028h) [reset = 0h]: [0] • Register Summary: [0]

11.14.5.6.14 PCIE_PREFETCH_LIMIT Register (Offset = 102Ch) [reset = 0h]

The Prefetchable Limit Upper 32 bits Register (PCIE_PREFETCH_LIMIT) is shown in Figure 11-1256 and described in Table 11-3047.

Table 11-3046. PCIE_PREFETCH_LIMIT Instances

Instance	Physical Address
PCIE	2180 102Ch

Figure 11-1256. PCIE_PREFETCH_LIMIT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PREFETCH_LIMIT																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3047. PCIE_PREFETCH_LIMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PREFETCH_LIMIT	R/W	0h	Upper 32 bits of Prefetchable Memory Limit Address. Used with 64 bit prefetchable memory addressing only.

Table 11-3048. Register Call Summary for PCIE_PREFETCH_LIMIT

PCIe SS Functional Description • Use of Base Address Registers (BARs): [0]
Configuration Type 1 Registers • PCIE_PREFETCH_LIMIT Register (Offset = 102Ch) [reset = 0h]: [0] • Register Summary: [0]

11.14.5.6.15 PCIE_IOSPACE Register (Offset = 1030h) [reset = 0h]

The IO Base and Limit Upper 16 bits Register (PCIE_IOSPACE) is shown in Figure 11-1257 and described in Table 11-3050.

Table 11-3049. PCIE_IOSPACE Instances

Instance	Physical Address
PCIE	2180 1030h

Figure 11-1257. PCIE_IOSPACE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IOBASE																IOLIMIT															
R/W-0h																R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3050. PCIE_IOSPACE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	IOBASE	R/W	0h	Upper 16 bits of IO Base Address. Used with 32 bit IO space addressing only.
15-0	IOLIMIT	R/W	0h	Upper 16 bits of IO Limit Address. Used with 32 bit IO space addressing only.

Table 11-3051. Register Call Summary for PCIE_IOSPACE

PCIe SS Functional Description • Use of Base Address Registers (BARs): [0]
Configuration Type 1 Registers • PCIE_IOSPACE Register (Offset = 1030h) [reset = 0h]: [0] • Register Summary: [0]

11.14.5.6.16 PCIE_CAP_PTR Register (Offset = 1034h) [reset = 40h]

The Capabilities Pointer (PCIE_CAP_PTR) is shown in Figure 11-1258 and described in Table 11-3053.

Table 11-3052. PCIE_CAP_PTR Instances

Instance	Physical Address
PCIE	2180 1034h

Figure 11-1258. PCIE_CAP_PTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CAP_PTR							
R-0h																								R-40h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3053. PCIE_CAP_PTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reads return 0 and writes have no effect.
7-0	CAP_PTR	R	40h	First Capability Pointer. By default, it points to Power Management Capability structure. Writable from internal bus interface.

Table 11-3054. Register Call Summary for PCIE_CAP_PTR

Configuration Type 0 Registers • Register Summary: [0] • PCIE_CAP_PTR Register (Offset = 1034h) [reset = 40h]: [0]
Configuration Type 1 Registers • PCIE_CAP_PTR Register (Offset = 1034h) [reset = 40h]: [0]

11.14.5.6.17 PCIE_EXPNSN_ROM Register (Offset = 1038h) [reset = 0h]

The Expansion ROM Base Address (PCIE_EXPNSN_ROM) is shown in Figure 11-1259 and described in Table 11-3056.

Table 11-3055. PCIE_EXPNSN_ROM Instances

Instance	Physical Address
PCIE	2180 1038h

Figure 11-1259. PCIE_EXPNSN_ROM Register

31	30	29	28	27	26	25	24
EXP_ROM_BASE_ADDR							
R-0h							
23	22	21	20	19	18	17	16
EXP_ROM_BASE_ADDR							
R-0h							
15	14	13	12	11	10	9	8
EXP_ROM_BASE_ADDR					RESERVED		
R-0h					R-0h		
7	6	5	4	3	2	1	0
RESERVED							EXP_ROM_EN
R-0h							R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3056. PCIE_EXPNSN_ROM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	EXP_ROM_BASE_ADDR	R	0h	Address of Expansion ROM
10-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	EXP_ROM_EN	R	0h	Expansion ROM Enable

Table 11-3057. Register Call Summary for PCIE_EXPNSN_ROM

Configuration Type 0 Registers • PCIE_EXPNSN_ROM Register (Offset = 1030h) [reset = 0h]: [0] • Register Summary: [0]
Configuration Type 1 Registers • PCIE_EXPNSN_ROM Register (Offset = 1038h) [reset = 0h]: [0]

11.14.5.6.18 PCIE_BRIDGE_INT Register (Offset = 103Ch) [reset = 1FFh]

The Bridge Control and Interrupt Register (PCIE_BRIDGE_INT) is shown in Figure 11-1260 and described in Table 11-3059.

Table 11-3058. PCIE_BRIDGE_INT Instances

Instance	Physical Address
PCIE	2180 103Ch

Figure 11-1260. PCIE_BRIDGE_INT Register

31	30	29	28	27	26	25	24
RESERVED				SERREN_STATUS	TIMER_STATUS	SEC_TIMER	PRI_TIMER
R-0h				R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
B2B_EN	SEC_BUS_RST	MST_ABORT_MODE	VGA_DECODE	VGA_EN	ISA_EN	SERR_EN	PERR_RESP_EN
R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
INT_PIN							
R-1h							
7	6	5	4	3	2	1	0
INT_LINE							
R/W-FFh							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3059. PCIE_BRIDGE_INT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reads return 0 and writes have no effect.
27	SERREN_STATUS	R	0h	Discard Timer SERR Enable Status. Not Applicable to PCI Express. Hardwired to 0.
26	TIMER_STATUS	R	0h	Discard Timer Status. Not applicable to PCI Express. Hardwired to 0.
25	SEC_TIMER	R	0h	Secondary Discard Timer. Not applicable to PCI Express. Hardwired to 0.
24	PRI_TIMER	R	0h	Primary Discard Timer. Not applicable to PCI Express. Hardwired to 0.
23	B2B_EN	R	0h	Fast Back to Back Transactions Enable. Not applicable to PCI Express. Hardwired to 0.
22	SEC_BUS_RST	R/W	0h	Secondary Bus Reset.
21	MST_ABORT_MODE	R	0h	Master Abort Mode. Not applicable to PCI Express. Hardwired to 0.
20	VGA_DECODE	R/W	0h	VGA 16 bit Decode
19	VGA_EN	R/W	0h	VGA Enable
18	ISA_EN	R/W	0h	ISA Enable
17	SERR_EN	R/W	0h	SERR Enable. Set to enable forwarding of ERR_COR, ERR_NONFATAL and ERR_FATAL messages.
16	PERR_RESP_EN	R/W	0h	Parity Error Response Enable. This bit controls the logging of poisoned TLPs in the Master Data Parity Error bit in the Secondary Status Register.

Table 11-3059. PCIE_BRIDGE_INT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-8	INT_PIN	R	1h	Interrupt Pin. It identifies the legacy interrupt message that the device uses. For single function configuration, the core only uses INTA. This register is writable through internal bus interface. • 0 = Legacy interrupt is not being used • 1h = INTA • 2h = INTB • 3h = INTC • 4h = INTD • Others = Reserved.
7-0	INT_LINE	R/W	FFh	Interrupt Line. Value is system software specified.

Table 11-3060. Register Call Summary for PCIE_BRIDGE_INT

Configuration Type 1 Registers
- PCIE_BRIDGE_INT Register (Offset = 103Ch) [reset = 1FFh]: [0]
- Register Summary: [0]

11.14.5.7 Power Management Capability Registers

11.14.5.7.1 Register Summary

Table 11-3061. PCIe Instances

Instance	Base Address
PCIE	2180 0000h

Table 11-3062. Power Management Capability Registers

Offset	Acronym	Register Name	PCIE Physical Address	Section
1040h	PCIE_PMCAP	Power Management Capability Register	2180 1040h	Section 11.14.5.7.2
1044h	PCIE_PM_CTL_STAT	Power Management Control and Status Register	2180 1044h	Section 11.14.5.7.3

11.14.5.7.2 PCIE_PMCAP Register (Offset = 1040h) [reset = 35001h]

The Power Management Capability Register (PCIE_PMCAP) is shown in Figure 11-1261 and described in Table 11-3064.

Table 11-3063. PCIE_PMCAP Instances

Instance	Physical Address
PCIE	2180 1040h

Figure 11-1261. PCIE_PMCAP Register

31	30	29	28	27	26	25	24
PME_SUPP_N					D2_SUPP_N	D1_SUPP_N	AUX_CURR_N
R-0h					R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
AUX_CURR_N		DSI_N	RESERVED	PME_CLK	PME_SPEC_VER		
R-0h		R-0h	R-0h	R-0h	R-3h		
15	14	13	12	11	10	9	8
PM_NEXT_PTR							
R-50h							
7	6	5	4	3	2	1	0
PM_CAP_ID							
R-1h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3064. PCIE_PMCAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	PME_SUPP_N	R	0h	PME Support. Identifies the power states from which generates PME Messages. A value of 0 for any bit indicates that the device (or function) is not capable of generating PME Messages while in that power state. Writable from internal bus interface. • Bit 31 = If set, PME Messages can be generated from D3cold. • Bit 30 = If set, PME Messages can be generated from D3hot. • Bit 29 = If set, PME Messages can be generated from D2. • Bit 28 = If set, PME Messages can be generated from D1. • Bit 27 = If set, PME Messages can be generated from D0.
26	D2_SUPP_N	R	0h	D2 Support. Writable from internal bus interface.
25	D1_SUPP_N	R	0h	D1 Support. Writable from internal bus interface.
24-22	AUX_CURR_N	R	0h	Auxiliary Current. Writable from internal bus interface.
21	DSI_N	R	0h	Device Specific Initialization. Writable from internal bus interface.
20	RESERVED	R	0h	Reads return 0 and writes have no effect.
19	PME_CLK	R	0h	PME Clock. Hardwired to zero.
18-16	PME_SPEC_VER	R	3h	Power Management Specification Version. Writable from internal bus interface.
15-8	PM_NEXT_PTR	R	50h	Next capability pointer. By default, it points to Message Signaled Interrupt structure.Writable from internal bus interface.
7-0	PM_CAP_ID	R	1h	Power Management Capability ID.

Table 11-3065. Register Call Summary for PCIE_PMCAP

Power Management Capability Registers
- Register Summary: [0]
- PCIE_PMCAP Register (Offset = 1040h) [reset = 35001h]: [0]

11.14.5.7.3 PCIE_PM_CTL_STAT Register (Offset = 1044h) [reset = 0h]

The Power Management Control and Status Register (PCIE_PM_CTL_STAT) is shown in Figure 11-1262 and described in Table 11-3067.

Table 11-3066. PCIE_PM_CTL_STAT Instances

Instance	Physical Address
PCIE	2180 1044h

Figure 11-1262. PCIE_PM_CTL_STAT Register

31	30	29	28	27	26	25	24
DATA_REG							
R-0h							
23	22	21	20	19	18	17	16
CLK_CTRL_EN	B2_B3_SUPPORT	RESERVED					
R-0h	R-0h	R-0h					
15	14	13	12	11	10	9	8
PME_STATUS	DATA_SCALE			DATA_SELECT			PME_EN
R/W1C-0h	R-0h			R-0h			R/W-0h
7	6	5	4	3	2	1	0
RESERVED				NO_SOFT_RST	RESERVED	PWR_STATE	
R-0h				R-0h	R-0h	R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-3067. PCIE_PM_CTL_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DATA_REG	R	0h	Data register for additional information. Not supported.
23	CLK_CTRL_EN	R	0h	Bus Power/Clock Control Enable. Hardwired to zero.
22	B2_B3_SUPPORT	R	0h	B2 and B3 support. Hardwired to zero.
21-16	RESERVED	R	0h	Reads return 0 and writes have no effect.
15	PME_STATUS	R/W1C	0h	PME Status. Indicates if a previously enabled PME event occurred or not.
14-13	DATA_SCALE	R	0h	Data Scale. Not supported.
12-9	DATA_SELECT	R	0h	Data Select. Not supported.
8	PME_EN	R/W	0h	PME Enable. Value of 1 indicates device is enabled to generate PME. Writable from internal bus interface.
7-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3	NO_SOFT_RST	R	0h	No Soft Reset. It is set to disable reset during a transition from D3 to D0. Writable from internal bus interface.
2	RESERVED	R	0h	Reads return 0 and writes have no effect.
1-0	PWR_STATE	R/W	0h	Power State. Controls the device power state. Writes are ignored if the state is not supported. Writable from internal bus interface. • 0 = D0 power state • 1h = D1 power state • 2h = D2 power state • 3h = D3 power states

Table 11-3068. Register Call Summary for PCIE_PM_CTL_STAT

Power Management Capability Registers
• PCIE_PM_CTL_STAT Register (Offset = 1044h) [reset = 0h]: [0] • Register Summary: [0]

11.14.5.8 Message Signaled Interrupts Registers

11.14.5.8.1 Register Summary

Table 11-3069. PCIe Instances

Instance	Base Address
PCIE	2180 0000h

Table 11-3070. Message Signaled Interrupts Registers

Offset	Acronym	Register Name	PCIE Physical Address	Section
1050h	PCIE_MSI_CAP	MSI Capabilities Register	2180 1050h	Section 11.14.5.8.2
1054h	PCIE_MSI_LOW32	MSI Lower 32 Bits register	2180 1054h	Section 11.14.5.8.3
1058h	PCIE_MSI_UP32	MSI Upper 32 Bits register	2180 1058h	Section 11.14.5.8.4
105Ch	PCIE_MSI_DATA	MSI Data Register (Offset is 0x1058 if 64-bit addressing not enabled)	2180 105Ch	Section 11.14.5.8.5

11.14.5.8.2 PCIE_MSI_CAP Register (Offset = 1050h) [reset = 807005h]

The MSI Capabilities Register (PCIE_MSI_CAP) is shown in Figure 11-1263 and described in Table 11-3072.

Table 11-3071. PCIE_MSI_CAP Instances

Instance	Physical Address
PCIE	2180 1050h

Figure 11-1263. PCIE_MSI_CAP Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
64BIT_EN	MULT_MSG_EN			MULT_MSG_CAP			MSI_EN
R-1h	R/W-0h			R-0h			R/W-0h
15	14	13	12	11	10	9	8
NEXT_CAP							
R-70h							
7	6	5	4	3	2	1	0
CAP_ID							
R-5h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3072. PCIE_MSI_CAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reads return 0 and writes have no effect.
23	64BIT_EN	R	1h	64-bit addressing enabled. Writable from internal bus interface.
22-20	MULT_MSG_EN	R/W	0h	Multiple message enabled. Indicates that multiple message mode is enabled by software. Number of messages enabled must not be greater than multiple message capable value. • 0 = 1 • 1h = 2 • 2h = 4 • 3h = 8 • 4h = 16 • 5h = 32 • Others = Reserved
19-17	MULT_MSG_CAP	R	0h	Multiple message capable. Writable from internal bus interface. • 0 = 1 • 1h = 2 • 2h = 4 • 3h = 8 • 4h = 16 • 5h = 32 • Others = Reserved
16	MSI_EN	R/W	0h	MSI Enabled. When set, INTx must be disabled.
15-8	NEXT_CAP	R	70h	Next capability pointer. By default, it points to PCI Express Capabilities structure. Writable from internal bus interface.
7-0	CAP_ID	R	5h	MSI Capability ID.

Table 11-3073. Register Call Summary for PCIE_MSI_CAP

PCIe SS Functional Description • MSI Interrupt Generation in EP Mode: [0][1]
Message Signaled Interrupts Registers • PCIE_MSI_CAP Register (Offset = 1050h) [reset = 807005h]: [0] • Register Summary: [0]

11.14.5.8.3 PCIE_MSI_LOW32 Register (Offset = 1054h) [reset = 0h]

The MSI Lower 32 Bits Register (PCIE_MSI_LOW32) is shown in Figure 11-1264 and described in Table 11-3075.

Table 11-3074. PCIE_MSI_LOW32 Instances

Instance	Physical Address
PCIE	2180 1054h

Figure 11-1264. PCIE_MSI_LOW32 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOWP32_ADDR																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3075. PCIE_MSI_LOW32 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LOW32_ADDR	R/W	0h	Lower 32 bits address

Table 11-3076. Register Call Summary for PCIE_MSI_LOW32

PCIe SS Functional Description • MSI Interrupt Generation in EP Mode: [0][1]
Message Signaled Interrupts Registers • Register Summary: [0] • PCIE_MSI_LOW32 Register (Offset = 1054h) [reset = 0h]: [0]

11.14.5.8.4 PCIE_MSI_UP32 Register (Offset = 1058h) [reset = 0h]

The MSI Upper 32 Bits Register (PCIE_MSI_UP32) is shown in Figure 11-1265 and described in Table 11-3078.

Table 11-3077. PCIE_MSI_UP32 Instances

Instance	Physical Address
PCIE	2180 1058h

Figure 11-1265. PCIE_MSI_UP32 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UP32_ADDR																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3078. PCIE_MSI_UP32 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UP32_ADDR	R/W	0h	Upper 32 bits address

Table 11-3079. Register Call Summary for PCIE_MSI_UP32

PCIe SS Functional Description • MSI Interrupt Generation in EP Mode: [0][1]
Message Signaled Interrupts Registers • Register Summary: [0] • PCIE_MSI_UP32 Register (Offset = 1058h) [reset = 0h]: [0]

11.14.5.8.5 PCIE_MSI_DATA Register (Offset = 105Ch) [reset = 0h]

The MSI Data Register (PCIE_MSI_DATA) is shown in Figure 11-1266 and described in Table 11-3081. The offset is 105Ch (the offset is 1058h if 64-bit addressing not enabled).

Table 11-3080. PCIE_MSI_DATA Instances

Instance	Physical Address
PCIE	2180 105Ch

Figure 11-1266. PCIE_MSI_DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																MSI_DATA															
R-0h																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3081. PCIE_MSI_DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reads return 0 and writes have no effect.
15-0	MSI_DATA	R/W	0h	MSI Data

Table 11-3082. Register Call Summary for PCIE_MSI_DATA

PCIe SS Functional Description • MSI Interrupt Generation in EP Mode: [0][1]
Message Signaled Interrupts Registers • Register Summary: [0] • PCIE_MSI_DATA Register (Offset = 105Ch) [reset = 0h]: [0]

11.14.5.9 PCI Express Capabilities Registers

11.14.5.9.1 Register Summary

Table 11-3083. PCIe Instances

Instance	Base Address
PCIE	2180 0000h

Table 11-3084. PCI Express Capabilities Registers

Offset	Acronym	Register Name	PCIE Physical Address	Section
1070h	PCIE_CAP	PCI Express Capabilities Register	2180 1070h	Section 11.14.5.9.2
1074h	PCIE_DEVICE_CAP	Device Capabilities Register	2180 1074h	Section 11.14.5.9.3
1078h	PCIE_DEV_STAT_CTRL	Device Status and Control Register	2180 1078h	Section 11.14.5.9.4
107Ch	PCIE_LINK_CAP	Link Capabilities Register	2180 107Ch	Section 11.14.5.9.5
1080h	PCIE_LINK_STAT_CTRL	Link Status and Control Register	2180 1080h	Section 11.14.5.9.6
1084h	PCIE_SLOT_CAP	Slot Capabilities Register (RC Mode only)	2180 1084h	Section 11.14.5.9.7
1088h	PCIE_SLOT_STAT_CTRL	Slot Status and Control Register (RC Mode only)	2180 1088h	Section 11.14.5.9.8
108Ch	PCIE_ROOT_CTRL_CAP	Root Control and Capabilities Register (RC Mode only)	2180 108Ch	Section 11.14.5.9.9
1090h	PCIE_ROOT_STATUS	Root Status and Control Register (RC Mode only)	2180 1090h	Section 11.14.5.9.10
1094h	PCIE_DEV_CAP2	Device Capabilities 2 Register	2180 1094h	Section 11.14.5.9.11
1098h	PCIE_DEV_STAT_CTRL2	Device Status and Control Register 2	2180 1098h	Section 11.14.5.9.12
10A0h	PCIE_LINK_CTRL2	Link Control Register 2	2180 10A0h	Section 11.14.5.9.13

11.14.5.9.2 PCIE_CAP Register (Offset = 1070h) [reset = 420002h]

The PCI Express Capabilities Register (PCIE_CAP) is shown in Figure 11-1267 and described in Table 11-3086.

Table 11-3085. PCIE_CAP Instances

Instance	Physical Address
PCIE	2180 1070h

Figure 11-1267. PCIE_CAP Register

31	30	29	28	27	26	25	24
RESERVED				INT_MSG			SLT_IMPL_N
R-0h				R-0h			R-0h
23	22	21	20	19	18	17	16
DPORT_TYPE				PCIE_CAP			
R-see				R-2h			
15	14	13	12	11	10	9	8
NEXT_CAP							
R-0h							
7	6	5	4	3	2	1	0
CAP_ID							
R-2h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3086. PCIE_CAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reads return 0 and writes have no effect.
29-25	INT_MSG	R	0h	Interrupt Message Number. Updated by hardware and writable through internal bus Interface.
24	SLT_IMPL_N	R	0h	Slot Implemented. Writable from internal bus interface.
23-20	DPORT_TYPE	R	4h	Device Port Type. • 0 = EP type • 4h = RC type • Others = Reserved
19-16	PCIE_CAP	R	2h	PCI Express Capability Version
15-8	NEXT_CAP	R	0h	Next capability pointer. Writable from internal bus interface.
7-0	CAP_ID	R	2h	PCIe Capability ID.

Table 11-3087. Register Call Summary for PCIE_CAP

PCI Express Capabilities Registers
- Register Summary: [0]
- PCIE_CAP Register (Offset = 1070h) [reset = 420002h]: [0][1][2]

11.14.5.9.3 PCIE_DEVICE_CAP Register (Offset = 1074h) [reset = 8001h]

The Device Capabilities Register (PCIE_DEVICE_CAP) is shown in Figure 11-1268 and described in Table 11-3089.

Table 11-3088. PCIE_DEVICE_CAP Instances

Instance	Physical Address
PCIE	2180 1074h

Figure 11-1268. PCIE_DEVICE_CAP Register

31	30	29	28	27	26	25	24
RESERVED				PWR_LIMIT_SCALE		PWR_LIMIT_VALUE	
R-0h				R-0h		R-0h	
23	22	21	20	19	18	17	16
PWR_LIMIT_VALUE						RESERVED	
R-0h						R-0h	
15	14	13	12	11	10	9	8
ERR_RPT	RESERVED			L1_LATENCY			L0_LATENCY
R-1h	R-0h			R-see			R-see
7	6	5	4	3	2	1	0
L0_LATENCY		EXT_TAG_FLD	PHANTOM_FLD		MAX_PAYLD_SZ		
R-see		R-0h	R-0h		R-1h		

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3089. PCIE_DEVICE_CAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reads return 0 and writes have no effect.
27-26	PWR_LIMIT_SCALE	R	0h	Captured Slot Power Limit Scale. For upstream ports (EP ports) only.
25-18	PWR_LIMIT_VALUE	R	0h	Captured Slot Power Limit Value. For upstream ports (EP ports) only.
17-16	RESERVED	R	0h	Reads return 0 and writes have no effect.
15	ERR_RPT	R	1h	Role-based Error Reporting. Writable from internal bus interface.
14-12	RESERVED	R	0h	Reads return 0 and writes have no effect.
11-9	L1_LATENCY	R	0h	Endpoint L1 Acceptable Latency. Must be • 0 in RC mode • 3h for EP mode.
8-6	L0_LATENCY	R	0h	Endpoint L0s Acceptable Latency. Must be • 0 in RC mode • 4h for EP mode.
5	EXT_TAG_FLD	R	0h	Extended Tag Field Supported. Writable from internal interface but should not be as the hardware is not capable.
4-3	PHANTOM_FLD	R	0h	Phantom Field Supported. Writable from internal bus interface.
2-0	MAX_PAYLD_SZ	R	1h	Maximum Payload size supported. Writable from internal bus interface.

Table 11-3090. Register Call Summary for PCIE_DEVICE_CAP

PCI Express Capabilities Registers
- PCIE_DEVICE_CAP Register (Offset = 1074h) [reset = 8001h]: [0]
- Register Summary: [0]

11.14.5.9.4 PCIE_DEV_STAT_CTRL Register (Offset = 1078h) [reset = 2810h]

The Device Status and Control Register (PCIE_DEV_STAT_CTRL) is shown in Figure 11-1269 and described in Table 11-3092.

Table 11-3091. PCIE_DEV_STAT_CTRL Instances

Instance	Physical Address
PCIE	2180 1078h

Figure 11-1269. PCIE_DEV_STAT_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED		TPEND	AUX_PWR	UNSUP_RQ_DET	FATAL_ERR	NFATAL_ERR	CORR_ERR
R-0h		R-0h	R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
15	14	13	12	11	10	9	8
RESERVED	MAX_REQ_SZ			NO_SNOOP	AUX_PWR_PM_EN	PHANTOM_EN	XTAG_FIELD_EN
R-0h	R/W-2h			R/W-1h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MAX_PAYLD			RELAXED	UNSUP_REQ_RP	FATAL_ERR_RP	NFATAL_ERR_RP	CORR_ERR_RP
R/W-0h			R/W-1h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-3092. PCIE_DEV_STAT_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reads return 0 and writes have no effect.
21	TPEND	R	0h	Transaction Pending
20	AUX_PWR	R	0h	Auxiliary Power Detected
19	UNSUP_RQ_DET	R/W1C	0h	Unsupported Request Detected
18	FATAL_ERR	R/W1C	0h	Fatal Error Detected
17	NFATAL_ERR	R/W1C	0h	Non-fatal Error Detected
16	CORR_ERR	R/W1C	0h	Correctable Error Detected
15	RESERVED	R	0h	Reads return 0 and writes have no effect.
14-12	MAX_REQ_SZ	R/W	2h	Maximum Read Request Size
11	NO_SNOOP	R/W	1h	Enable no snoop
10	AUX_PWR_PM_EN	R/W	0h	AUX Power PM Enable
9	PHANTOM_EN	R/W	0h	Phantom Function Enable
8	XTAG_FIELD_EN	R/W	0h	Extended Tag Field Enable
7-5	MAX_PAYLD	R/W	0h	Maximum Payload Size
4	RELAXED	R/W	1h	Enable Relaxed Ordering
3	UNSUP_REQ_RP	R/W	0h	Enable Unsupported Request Reporting
2	FATAL_ERR_RP	R/W	0h	Fatal Error Reporting Enable
1	NFATAL_ERR_RP	R/W	0h	Non-fatal Error Reporting Enable
0	CORR_ERR_RP	R/W	0h	Correctable Error Reporting Enable

Table 11-3093. Register Call Summary for PCIE_DEV_STAT_CTRL

PCIe SS Functional Description • PCIe Baseline Error Handling: [0][1][2] • Error Reporting: [0]
PCI Express Capabilities Registers • PCIE_DEV_STAT_CTRL Register (Offset = 1078h) [reset = 2810h]: [0] • Register Summary: [0]

11.14.5.9.5 PCIE_LINK_CAP Register (Offset = 107Ch) [reset = 35422h]

The Link Capabilities Register (PCIE_LINK_CAP) is shown in Figure 11-1270 and described in Table 11-3095.

Table 11-3094. PCIE_LINK_CAP Instances

Instance	Physical Address
PCIE	2180 107Ch

Figure 11-1270. PCIE_LINK_CAP Register

31	30	29	28	27	26	25	24
PORT_NUM							
R-0h							
23	22	21	20	19	18	17	16
RESERVED		BW_NOTIFY_CAP	DLL_REP_CAP	DOWN_ERR_REP_CAP	CLK_PWR_MGMT	L1_EXIT_LAT	
R-0h		R-0h	R-0h	R-0h	R-0h	R-6h	
15	14	13	12	11	10	9	8
L1_EXIT_LAT	LOS_EXIT_LAT			AS_LINK_PM		MAX_LINK_WIDTH	
R-6h	R-5h			R-1h		R-2h	
7	6	5	4	3	2	1	0
MAX_LINK_WIDTH				MAX_LINK_SPEED			
R-2h				R-2h			

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3095. PCIE_LINK_CAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	PORT_NUM	R	0h	Port Number. Writable from internal bus interface.
23-22	RESERVED	R	0h	Reads return 0 and writes have no effect.
21	BW_NOTIFY_CAP	R	0h	Link Bandwidth Notification Capable. • 0 = For upstream ports (EP ports) • 1 = For downstream ports (RC ports)
20	DLL_REP_CAP	R	0h	Data Link Layer Active Reporting Capable. • 0 = For upstream ports (EP ports) • 1 = For downstream ports (RC ports)
19	DOWN_ERR_REP_CAP	R	0h	Surprise Down Error Reporting Capable. Not supported. Always zero.
18	CLK_PWR_MGMT	R	0h	Clock Power Management. Writable from internal bus interface. For upstream ports (EP Ports), a value of 1h in this bit indicates that the component tolerates the removal of any reference clock(s) in the L1 and L2/L3 Ready Link states. A value of 0 indicates the reference clock(s) must not be removed in these Link states. For downstream ports (RC Ports), this bit is always 0.
17-15	L1_EXIT_LAT	R	6h	L1 Exit Latency when common clock is used. Writable from internal bus interface. • 0 = Less than 64 ns • 1h = 64 ns to less than 128 ns • 2h = 128 ns to less than 256 ns • 3h = 256 ns to less than 512 ns • 4h = 512 ns to less than 1 µs • 5h = 1 µs to less than 2 µs • 6h = 2 µs to less than 4 µs • 7h = More than 4 µs

Table 11-3095. PCIE_LINK_CAP Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14-12	LOS_EXIT_LAT	R	5h	L0s Exit Latency. Writable from internal bus interface. • 0 = Less than 64 ns • 1h = 64 ns to less than 128 ns • 2h = 128 ns to less than 256 ns • 3h = 256 ns to less than 512 ns • 4h = 512 ns to less than 1 µs • 5h = 1 µs to less than 2 µs • 6h = 2 µs to less than 4 µs • 7h = More than 4 µs
11-10	AS_LINK_PM	R	1h	Active State Link Power Management Support. Writable from internal bus interface. • 1h = L0s entry supported. • 3h = L0s and L1 supported. • Others = Reserved.
9-4	MAX_LINK_WIDTH	R	2h	Maximum Link Width (×N – corresponding to N Lanes). Writable from internal bus interface. • 1h = ×1 • 2h = ×2 • Others = Reserved.
3-0	MAX_LINK_SPEED	R	2h	Maximum Link Speed. Writable from internal bus interface. • 1h = 2.5GT/s Link speed supported. • 2h = 5.0 GT/s and 2.5 GT/s Link speeds supported. • Others = Reserved.

Table 11-3096. Register Call Summary for PCIE_LINK_CAP

PCIe SS Functional Description • L0s State: [0][1]
PCI Express Capabilities Registers • Register Summary: [0] • PCIE_LINK_CAP Register (Offset = 107Ch) [reset = 35422h]: [0]

11.14.5.9.6 PCIE_LINK_STAT_CTRL Register (Offset = 1080h) [reset = 10110008h]

The Link Status and Control Register (PCIE_LINK_STAT_CTRL) is shown in Figure 11-1271 and described in Table 11-3098.

Table 11-3097. PCIE_LINK_STAT_CTRL Instances

Instance	Physical Address
PCIE	2180 1080h

Figure 11-1271. PCIE_LINK_STAT_CTRL Register

31	30	29	28	27	26	25	24
LINK_BW_STATUS	LINK_BW_MGMT_STATUS	DLL_ACTIVE	SLOT_CLK_CFG	LINK_TRAINING	RESERVED	NEGOTIATED_LINK_WD	
R/W1C-0h	R/W1C-0h	R-0h	R-1h	R-0h	R-0h	R-1h	
23	22	21	20	19	18	17	16
NEGOTIATED_LINK_WD				LINK_SPEED			
R-1h				R-1h			
15	14	13	12	11	10	9	8
RESERVED				LINK_BW_INT_EN	LINK_BW_MGMT_INT_EN	HW_AUTO_WIDTH_DIS	CLK_PWR_MGMT_EN
R-0h				R-0h	R-0h	R-0h	R/W-0h
7	6	5	4	3	2	1	0
EXT_SYNC	COMMON_CLK_CFG	RETRAIN_LINK	LINK_DISABLE	RCB	RESERVED	ACTIVE_LINK_PM	
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-1h	R-0h	R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-3098. PCIE_LINK_STAT_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	LINK_BW_STATUS	R/W1C	0h	Link Autonomous Bandwidth Status. This bit is Set by hardware to indicate that hardware has autonomously changed Link speed or width, without the Port transitioning through DL_Down status, for reasons other than to attempt to correct unreliable Link operation. This bit must be set if the Physical Layer reports a speed or width change was initiated by the downstream component that was indicated as an autonomous change. Not applicable and reserved for EP.
30	LINK_BW_MGMT_STATUS	R/W1C	0h	Link Bandwidth Management Status. This bit is Set by hardware to indicate that either of the following has occurred without the Port transitioning through DL_Down status: • A Link retraining has completed following a write of 1b to the Retrain Link bit • Hardware has changed Link speed or width to attempt to correct unreliable Link operation, either through an LTSSM timeout or a higher level process. This bit must be set if the Physical Layer reports a speed or width change was initiated by the downstream component that was not indicated as an autonomous change. Not applicable and reserved for EP.
29	DLL_ACTIVE	R	0h	Data Link Layer Active This bit indicates the status of the Data Link Control and Management State Machine. It returns a 1 to indicate the DL_Active state, 0 otherwise. Not applicable and reserved for EP.

Table 11-3098. PCIE_LINK_STAT_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
28	SLOT_CLK_CFG	R	1h	Slot Clock Configuration. Writable from internal bus interface. This bit indicates that the component uses the same physical reference clock that the platform provides on the connector.
27	LINK_TRAINING	R	0h	Link Training. Not applicable to EP.
26	RESERVED	R	0h	Reserved
25-20	NEGOTIATED_LINK_WD	R	1h	Negotiated Link Width. Set automatically by hardware after link initialization.
19-16	LINK_SPEED	R	1h	Link Speed. Set automatically by hardware after link initialization.
15-12	RESERVED	R	0h	Reads return 0 and writes have no effect.
11	LINK_BW_INT_EN	R	0h	Link Autonomous Bandwidth Interrupt Enable. Not applicable and is reserved for EP.
10	LINK_BW_MGMT_INT_EN	R	0h	Link Bandwidth Management Interrupt Enable. Not applicable and is reserved for EP.
9	HW_AUTO_WIDTH_DIS	R	0h	Hardware Autonomous Width Disable. Not supported and hardwired to zero.
8	CLK_PWR_MGMT_EN	R/W	0h	Enable Clock Power Management.
7	EXT_SYNC	R/W	0h	Extended Synchronization.
6	COMMON_CLK_CFG	R/W	0h	Common Clock Configuration. • 0 = Indicates that this device and the device at the opposite end of the link are operating with separate reference clock sources. • 1 = Indicates that this device and the device at the opposite end of the link are operating with a common clock source.
5	RETRAIN_LINK	R/W	0h	Retrain Link. Not applicable and reserved for EP.
4	LINK_DISABLE	R/W	0h	This bit disables the link by directing the LTSSM to the Disabled state when set. Not applicable and is reserved for EP.
3	RCB	R	1h	Read Completion Boundary. Writable via internal bus interface for RC. • 0 = 64 bytes • 1 = 128 bytes
2	RESERVED	R	0h	Reads return 0 and writes have no effect.
1-0	ACTIVE_LINK_PM	R/W	0h	Active State Link Power Management Control • 0 = Disabled. • 1h = L0s entry enabled. • 2h = L1 entry enabled. • 3h = L0s and L1 entry enabled.

Table 11-3099. Register Call Summary for PCIE_LINK_STAT_CTRL

PCIe SS Functional Description • PCIe Baseline Error Handling: [0][1][2] • ASPM L1 State: [0][1] • Clock Control: [0] • Loopback Master: [0]
PCI Express Capabilities Registers • PCIE_LINK_STAT_CTRL Register (Offset = 1080h) [reset = 10110008h]: [0] • Register Summary: [0]

11.14.5.9.7 PCIE_SLOT_CAP Register (Offset = 1084h) [reset = 40h]

The Slot Capabilities Register (PCIE_SLOT_CAP) (RC Mode only) is shown in Figure 11-1272 and described in Table 11-3101. This register is for RC mode only.

Table 11-3100. PCIE_SLOT_CAP Instances

Instance	Physical Address
PCIE	2180 1084h

Figure 11-1272. PCIE_SLOT_CAP Register

31	30	29	28	27	26	25	24
SLOT_NUM							
R-0h							
23	22	21	20	19	18	17	16
SLOT_NUM					CMD_COMP_SUPP	EML_PRESENT	PWR_LMT_SCALE
R-0h					R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
PWR_LMT_SCALE	PWR_LMT_VALUE						
R-0h	R-0h						
7	6	5	4	3	2	1	0
PWR_LMT_VALUE	HP_CAP	HP_SURPRISE	PWR_IND	ATTN_IND	MRL_SENSOR	PWR_CTL	ATTN_BUTTON
R-0h	R-1h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3101. PCIE_SLOT_CAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	SLOT_NUM	R	0h	Physical Slot Number. Writable from internal bus interface.
18	CMD_COMP_SUPP	R	0h	No Command Complete Support. Writable from internal bus interface. When Set, this bit indicates that this slot does not generate software notification when an issued command is completed by the Hot-Plug Controller.
17	EML_PRESENT	R	0h	Electromechanical Interlock Present. Writable from internal bus interface. When Set, this bit indicates that an Electromechanical Interlock is implemented on the chassis for this slot.
16-15	PWR_LMT_SCALE	R	0h	Slot Power Limit Scale. Writable from internal bus interface.
14-7	PWR_LMT_VALUE	R	0h	Slow Power Limit Value. Writable from internal bus interface.
6	HP_CAP	R	1h	Hot Plug Capable. Writable from internal bus interface.
5	HP_SURPRISE	R	0h	Hot Plug Surprise. Writable from internal bus interface.
4	PWR_IND	R	0h	Power Indicator Present. Writable from internal bus interface.
3	ATTN_IND	R	0h	Attention Indicator Present. Writable from internal bus interface.
2	MRL_SENSOR	R	0h	MRL Sensor Present. Writable from internal bus interface.
1	PWR_CTL	R	0h	Power Controller Present. Writable from internal bus interface. If there is no power controller, software must ensure that system power is up before reading Presence Detect state.
0	ATTN_BUTTON	R	0h	Attention Indicator Present. Writable from internal bus interface.

Table 11-3102. Register Call Summary for PCIE_SLOT_CAP

PCI Express Capabilities Registers
• PCIE_SLOT_CAP Register (Offset = 1084h) [reset = 40h]: [0] • Register Summary: [0]

11.14.5.9.8 PCIE_SLOT_STAT_CTRL Register (Offset = 1088h) [reset = 4003C0h]

The Slot Status and Control Register (PCIE_SLOT_STAT_CTRL) is shown in Figure 11-1273 and described in Table 11-3104. This register is for RC mode only.

Table 11-3103. PCIE_SLOT_STAT_CTRL Instances

Instance	Physical Address
PCIE	2180 1088h

Figure 11-1273. PCIE_SLOT_STAT_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							DLL_STATE
R-0h							R/W1C-0h
23	22	21	20	19	18	17	16
EM_LOCK	PRESENCE_DET	MRL_STATE	CMD_COMLETE	PRESENCE_CHG	MRL_CHANGE	PWR_FAULT	ATTN_PRESSED
R-0h	R-1h	R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
15	14	13	12	11	10	9	8
RESERVED			DLL_CHG_EN	EM_LOCK_CTL	PM_CTL	PM_IND_CTL	
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-3h	
7	6	5	4	3	2	1	0
ATTN_IND_CTL		HP_INT_EN	CMD_CMP_INT_EN	PRS_DET_CHG_EN	MRL_CHG_EN	PWR_FLT_DET_EN	ATTN_BUTT_EN
R/W-3h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-3104. PCIE_SLOT_STAT_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reads return 0 and writes have no effect.
24	DLL_STATE	R/W1C	0h	Data Link Layer State Changed
23	EM_LOCK	R	0h	Electromechanical Lock Status
22	PRESENCE_DET	R	1h	Presence Detect State
21	MRL_STATE	R	0h	MRL Sensor State
20	CMD_COMLETE	R/W1C	0h	Command Completed
19	PRESENCE_CHG	R/W1C	0h	Presence Detect Changed
18	MRL_CHANGE	R/W1C	0h	MRL Sensor Changed
17	PWR_FAULT	R/W1C	0h	Power Fault Detected
16	ATTN_PRESSED	R/W1C	0h	Attention Button Pressed.
15-13	RESERVED	R	0h	Reads return 0 and writes have no effect.
12	DLL_CHG_EN	R/W	0h	Data Link Layer State Changed Enable.
11	EM_LOCK_CTL	R/W	0h	Electromechanical Interlock Control.
10	PM_CTL	R/W	0h	Power Controller Control
9-8	PM_IND_CTL	R/W	3h	Power Indicator Control
7-6	ATTN_IND_CTL	R/W	3h	Attention Indicator Control.
5	HP_INT_EN	R/W	0h	Hot Plug Interrupt Enable.
4	CMD_CMP_INT_EN	R/W	0h	Command Completed Interrupt Enable.
3	PRS_DET_CHG_EN	R/W	0h	Presence Detect Changed Enable.
2	MRL_CHG_EN	R/W	0h	MRL Sensor Changed Enable.
1	PWR_FLT_DET_EN	R/W	0h	Power Fault Detected Enable.
0	ATTN_BUTT_EN	R/W	0h	Attention Button Pressed Enable.

Table 11-3105. Register Call Summary for PCIE_SLOT_STAT_CTRL

PCI Express Capabilities Registers
• Register Summary: [0]
• PCIE_SLOT_STAT_CTRL Register (Offset = 1088h) [reset = 4003C0h]: [0]

11.14.5.9.9 PCIE_ROOT_CTRL_CAP Register (Offset = 108Ch) [reset = 0h]

The Root Control and Capabilities Register (PCIE_ROOT_CTRL_CAP) is shown in Figure 11-1274 and described in Table 11-3107. This register is for RC mode only.

Table 11-3106. PCIE_ROOT_CTRL_CAP Instances

Instance	Physical Address
PCIE	2180 108Ch

Figure 11-1274. PCIE_ROOT_CTRL_CAP Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							CRS_SW
R-0h							R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			CRS_SW_EN	PME_INT_EN	SERR_FATAL_ERR	SERR_NFATAL_ERR	SERR_EN
R-0h			R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3107. PCIE_ROOT_CTRL_CAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reads return 0 and writes have no effect.
16	CRS_SW	R	0h	CRS Software Visibility. Not supported and set to 0.
15-5	RESERVED	R	0h	Reads return 0 and writes have no effect.
4	CRS_SW_EN	R	0h	CRS Software Visibility Enable. Not supported and set to 0x0.
3	PME_INT_EN	R/W	0h	PME Interrupt Enable
2	SERR_FATAL_ERR	R/W	0h	System Error on Fatal Error Enable
1	SERR_NFATAL_ERR	R/W	0h	System Error on Non-fatal Error Enable
0	SERR_EN	R/W	0h	System Error on Correctable Error Enable

Table 11-3108. Register Call Summary for PCIE_ROOT_CTRL_CAP

PCIe SS Functional Description • PCIe Baseline Error Handling: [0]
PCI Express Capabilities Registers • Register Summary: [0] • PCIE_ROOT_CTRL_CAP Register (Offset = 108Ch) [reset = 0h]: [0]

11.14.5.9.10 PCIE_ROOT_STATUS Register (Offset = 1090h) [reset = 0h]

The Root Status and Control Register (PCIE_ROOT_STATUS) is shown in Figure 11-1275 and described in Table 11-3110. This register is for RC mode only.

Table 11-3109. PCIE_ROOT_STATUS Instances

Instance	Physical Address
PCIE	2180 1090h

Figure 11-1275. PCIE_ROOT_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						PME_PEND	PME_STATUS
R-0h						R-0h	R/W1C-0h
15	14	13	12	11	10	9	8
PME_REQ_ID							
R-0h							
7	6	5	4	3	2	1	0
PME_REQ_ID							
R-0h							

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-3110. PCIE_ROOT_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reads return 0 and writes have no effect.
17	PME_PEND	R	0h	Indicates that another PME is pending when the PME Status bit is Set.
16	PME_STATUS	R/W1C	0h	Indicates that PME was asserted by the PME Requester.
15-0	PME_REQ_ID	R	0h	ID of the last PME Requester. This field is only valid when the PME Status bit is Set.

Table 11-3111. Register Call Summary for PCIE_ROOT_STATUS

PCI Express Capabilities Registers
- Register Summary: [0]
- PCIE_ROOT_STATUS Register (Offset = 1090h) [reset = 0h]: [0]

11.14.5.9.11 PCIE_DEV_CAP2 Register (Offset = 1094h) [reset = 1Fh]

The Device Capabilities 2 Register (PCIE_DEV_CAP2) is shown in Figure 11-1276 and described in Table 11-3113.

Table 11-3112. PCIE_DEV_CAP2 Instances

Instance	Physical Address
PCIE	2180 1094h

Figure 11-1276. PCIE_DEV_CAP2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			CMPL_TO_DIS_SUPP	CMPL_TO_EN			
R-0h			R-1h	R-Fh			

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3113. PCIE_DEV_CAP2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reads return 0 and writes have no effect.
4	CMPL_TO_DIS_SUPP	R	1h	Completion timeout disable supported
3-0	CMPL_TO_EN	R	Fh	Completion timeout ranges supported. Applicable to RC/EP that issue requests on own behalf.

Table 11-3114. Register Call Summary for PCIE_DEV_CAP2

PCI Express Capabilities Registers
- Register Summary: [0]
- PCIE_DEV_CAP2 Register (Offset = 1094h) [reset = 1Fh]: [0]

11.14.5.9.12 PCIE_DEV_STAT_CTRL2 Register (Offset = 1098h) [reset = 0h]

The Device Status and Control Register 2 (PCIE_DEV_STAT_CTRL2) is shown in Figure 11-1277 and described in Table 11-3116.

Table 11-3115. PCIE_DEV_STAT_CTRL2 Instances

Instance	Physical Address
PCIE	2180 1098h

Figure 11-1277. PCIE_DEV_STAT_CTRL2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			CMPL_TO_DIS	CMPL_TO			
R-0h			R/W-0h	R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3116. PCIE_DEV_STAT_CTRL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reads return 0 and writes have no effect.
4	CMPL_TO_DIS	R/W	0h	Completion timeout disable
3-0	CMPL_TO	R/W	0h	Completion timeout value. It is strongly recommended that the Completion Timeout mechanism not expire in less than 10 ms. • 0 = Default range: 50 s to 50 ms. • 1h = 50 s to 100 s. • 2h = 1 ms to 10 ms. • 5h = 16 ms to 55 ms. • 6h = 65 ms to 210 ms. • 9h = 260 ms to 900 ms. • Ah = 1 s to 3.5 s. • Dh = 4 s to 13 s. • Eh = 17 s to 64 s. • Others = Reserved.

Table 11-3117. Register Call Summary for PCIE_DEV_STAT_CTRL2

PCI Express Capabilities Registers
- Register Summary: [0]
- PCIE_DEV_STAT_CTRL2 Register (Offset = 1098h) [reset = 0h]: [0]

11.14.5.9.13 PCIE_LINK_CTRL2 Register (Offset = 10A0h) [reset = 10002h]

The Link Control Register 2 (PCIE_LINK_CTRL2) is shown in Figure 11-1278 and described in Table 11-3119.

Table 11-3118. PCIE_LINK_CTRL2 Instances

Instance	Physical Address
PCIE	2180 10A0h

Figure 11-1278. PCIE_LINK_CTRL2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							DE_EMPH
R-0h							R-1h
15	14	13	12	11	10	9	8
RESERVED			POLL_DEEMPH	CMPL_SOS	ENTR_MOD_COMPL	TX_MARGIN	
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	
7	6	5	4	3	2	1	0
TX_MARGIN	SEL_DEEMPH	HW_AUTO_SPEED_DIS	ENTR_COMPL	TGT_SPEED			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-2h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3119. PCIE_LINK_CTRL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reads return 0 and writes have no effect.
16	DE_EMPH	R	1h	Current De-emphasis level • 0 = -6 dB • 1 = -3.5 dB
15-13	RESERVED	R	0h	Reads return 0 and writes have no effect.
12	POLL_DEEMPH	R/W	0h	De-emphasis level in polling-compliance state This bit sets the de-emphasis level in Polling Compliance state if the entry occurred due to the Enter Compliance bit being 1. • 0 = -6 dB • 1 = -3.5 dB
11	CMPL_SOS	R/W	0h	Compliance SOS. When this bit is set to 1, the LTSSM is required to send SKP Ordered Sets periodically in between the modified compliance patterns.
10	ENTR_MOD_COMPL	R/W	0h	Enter modified compliance. When this bit is set to 1, the device transmits Modified Compliance Pattern if the LTSSM enters Polling Compliance substate.
9-7	TX_MARGIN	R/W	0h	Value of non-de-emphasized voltage level at transmitter pins.
6	SEL_DEEMPH	R/W	0h	Selectable De-emphasis. When the Link is operating at 5.0 GT/s speed, this bit selects the level of de-emphasis for an upstream component. When the Link is operating at 2.5 GT/s speed, the setting of this bit has no effect. • 0 = -6 dB • 1 = -3.5 dB

Table 11-3119. PCIE_LINK_CTRL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	HW_AUTO_SPEED_DIS	R/W	0h	Hardware Autonomous Speed Disable. • 0 = Enable hardware to change the link speed. • 1 = Disables hardware from changing the Link speed for device specific reasons other than attempting to correct unreliable Link operation by reducing Link speed.
4	ENTR_COMPL	R/W	0h	Enter Compliance. Software is permitted to force a Link to enter Compliance mode at the speed indicated in the Target Link Speed field by setting this bit to 1 in both components on a Link and then initiating a hot reset on the Link.
3-0	TGT_SPEED	R/W	2h	Target Link Speed. • 1h = 2.5 GT/s Target Link Speed. • 2h = 5.0 GT/s Target Link Speed. • Others = Reserved.

Table 11-3120. Register Call Summary for PCIE_LINK_CTRL2

PCI Express Capabilities Registers
- PCIE_LINK_CTRL2 Register (Offset = 10A0h) [reset = 10002h]: [0]
- Register Summary: [0]

11.14.5.10 PCI Express Extended Capabilities Registers

11.14.5.10.1 Register Summary

Table 11-3121. PCIe Instances

Instance	Base Address
PCIE	2180 0000h

Table 11-3122. PCI Express Extended Capabilities Registers

Offset	Acronym	Register Name	PCIE Physical Address	Section
1100h	PCIE_EXTCAP	PCI Express Extended Capabilities Header	2180 1100h	Section 11.14.5.10.2
1104h	PCIE_UNCERR	PCI Express Uncorrectable Error Status Register	2180 1104h	Section 11.14.5.10.3
1108h	PCIE_UNCERR_MASK	PCI Express Uncorrectable Error Mask Register	2180 1108h	Section 11.14.5.10.4
110Ch	PCIE_UNCERR_SVRTY	PCI Express Uncorrectable Error Severity Register	2180 110Ch	Section 11.14.5.10.5
1110h	PCIE_CERR	PCI Express Correctable Error Status Register	2180 1110h	Section 11.14.5.10.6
1114h	PCIE_CERR_MASK	PCI Express Correctable Error Mask Register	2180 1114h	Section 11.14.5.10.7
1118h	PCIE_ACCR	PCI Express Advanced Capabilities and Control Register	2180 1118h	Section 11.14.5.10.8
111Ch	PCIE_HDR_LOG0	Header Log Register 0	2180 111Ch	Section 11.14.5.10.9
1120h	PCIE_HDR_LOG1	Header Log Register 1	2180 1120h	Section 11.14.5.10.10
1124h	PCIE_HDR_LOG2	Header Log Register 2	2180 1124h	Section 11.14.5.10.11
1128h	PCIE_HDR_LOG3	Header Log Register 3	2180 1128h	Section 11.14.5.10.12
112Ch	PCIE_RC_ERR_CMD	Root Error Command Register	2180 112Ch	Section 11.14.5.10.13
1130h	PCIE_RC_ERR_ST	Root Error Status Register	2180 1130h	Section 11.14.5.10.14
1134h	PCIE_ERR_SRC_ID	Error Source Identification Register	2180 1134h	Section 11.14.5.10.15

11.14.5.10.2 PCIE_EXTCAP Register (Offset = 1100h) [reset = 10001h]

The PCI Express Extended Capabilities Header (PCIE_EXTCAP) is shown in Figure 11-1279 and described in Table 11-3124.

Table 11-3123. PCIE_EXTCAP Instances

Instance	Physical Address
PCIE	2180 1100h

Figure 11-1279. PCIE_EXTCAP Register

31	30	29	28	27	26	25	24
NEXT_CAP							
R-0h							
23	22	21	20	19	18	17	16
NEXT_CAP				EXT_CAP_VER			
R-0h				R-1h			
15	14	13	12	11	10	9	8
EXT_CAP_ID							
R-1h							
7	6	5	4	3	2	1	0
EXT_CAP_ID							
R-1h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3124. PCIE_EXTCAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	NEXT_CAP	R	0h	Next Capability Pointer
19-16	EXT_CAP_VER	R	1h	Extended Capability Version
15-0	EXT_CAP_ID	R	1h	PCIe Extended Capability ID

Table 11-3125. Register Call Summary for PCIE_EXTCAP

PCI Express Extended Capabilities Registers

- PCIE_EXTCAP Register (Offset = 1100h) [reset = 10001h]: [0]
- Register Summary: [0]

11.14.5.10.3 PCIE_UNCERR Register (Offset = 1104h) [reset = 0h]

The PCI Express Uncorrectable Error Status Register (PCIE_UNCERR) is shown in Figure 11-1280 and described in Table 11-3127.

Table 11-3126. PCIE_UNCERR Instances

Instance	Physical Address
PCIE	2180 1104h

Figure 11-1280. PCIE_UNCERR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED			UR_ERR_ST	ECRC_ERR_ST	MTLP_ERR_ST	RCVR_OF_ST	UCMP_ST
R-0h			R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
15	14	13	12	11	10	9	8
CMPL_ABRT_ST	CMPL_TMOT_ST	FCP_ERR_ST	PSND_TLP_ST	RESERVED			
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R-0h			
7	6	5	4	3	2	1	0
RESERVED		SRPS_DN_ST	DLP_ERR_ST	RESERVED			
R-0h		R-0h	R/W1C-0h	R-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-3127. PCIE_UNCERR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-21	RESERVED	R	0h	Reads return 0 and writes have no effect.
20	UR_ERR_ST	R/W1C	0h	Unsupported Request Error Status
19	ECRC_ERR_ST	R/W1C	0h	ECRC Error Status
18	MTLP_ERR_ST	R/W1C	0h	Malformed TLP Status
17	RCVR_OF_ST	R/W1C	0h	Receiver Overflow Status
16	UCMP_ST	R/W1C	0h	Unexpected Completion Status
15	CMPL_ABRT_ST	R/W1C	0h	Completer Abort Status
14	CMPL_TMOT_ST	R/W1C	0h	Completion Timeout Status
13	FCP_ERR_ST	R/W1C	0h	Flow Control Protocol Error Status
12	PSND_TLP_ST	R/W1C	0h	Poisoned TLP Status
11-6	RESERVED	R	0h	Reads return 0 and writes have no effect.
5	SRPS_DN_ST	R	0h	Surprise Down Error Status. Not supported.
4	DLP_ERR_ST	R/W1C	0h	Data Link Protocol Error Status
3-0	RESERVED	R	0h	Reads return 0 and writes have no effect.

Table 11-3128. Register Call Summary for PCIE_UNCERR

PCIe SS Functional Description • PCIe Advanced Error Reporting: [0][1][2]
PCI Express Extended Capabilities Registers • Register Summary: [0] • PCIE_UNCERR Register (Offset = 1104h) [reset = 0h]: [0]

11.14.5.10.4 PCIE_UNCERR_MASK Register (Offset = 1108h) [reset = 0h]

The PCI Express Uncorrectable Error Mask Register (PCIE_UNCERR_MASK) is s Figure 11-1281 and described in Table 11-3130.

Table 11-3129. PCIE_UNCERR_MASK Instances

Instance	Physical Address
PCIE	2180 1108h

Figure 11-1281. PCIE_UNCERR_MASK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED			UR_ERR_MSK	ECRC_ERR_MSK	MTLP_ERR_MSK	RCVR_OF_MSK	UCMP_MSK
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
CMPL_ABRT_MSK	CMPL_TMOT_MSK	FCP_ERR_MSK	PSND_TLP_MSK	RESERVED			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h			
7	6	5	4	3	2	1	0
RESERVED		SRPS_DN_MSK	DLP_ERR_MSK	RESERVED			
R-0h		R-0h	R-0h	R/W-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3130. PCIE_UNCERR_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-21	RESERVED	R	0h	Reads return 0 and writes have no effect.
20	UR_ERR_MSK	R/W	0h	Unsupported Request Error Mask
19	ECRC_ERR_MSK	R/W	0h	ECRC Error Mask
18	MTLP_ERR_MSK	R/W	0h	Malformed TLP Mask
17	RCVR_OF_MSK	R/W	0h	Receiver Overflow Mask
16	UCMP_MSK	R/W	0h	Unexpected Completion Mask
15	CMPL_ABRT_MSK	R/W	0h	Completer Abort Mask
14	CMPL_TMOT_MSK	R/W	0h	Completion Timeout Mask
13	FCP_ERR_MSK	R/W	0h	Flow Control Protocol Error Mask
12	PSND_TLP_MSK	R/W	0h	Poisoned TLP Mask
11-6	RESERVED	R	0h	Reads return 0 and writes have no effect.
5	SRPS_DN_MSK	R	0h	Surprise Down Error Mask. Not supported.
4	DLP_ERR_MSK	R/W	0h	Data Link Protocol Error Mask
3-0	RESERVED	R	0h	Reads return 0 and writes have no effect.

Table 11-3131. Register Call Summary for PCIE_UNCERR_MASK

PCIe SS Functional Description • PCIe Advanced Error Reporting: [0]
PCI Express Extended Capabilities Registers • PCIE_UNCERR_MASK Register (Offset = 1108h) [reset = 0h]: [0] • Register Summary: [0]

11.14.5.10.5 PCIE_UNCERR_SVRTY Register (Offset = 110Ch) [reset = 62030h]

The PCI Express Uncorrectable Error Severity Register (PCIE_UNCERR_SVRTY) is shown in Figure 11-1282 and described in Table 11-3133.

Table 11-3132. PCIE_UNCERR_SVRTY Instances

Instance	Physical Address
PCIE	2180 110Ch

Figure 11-1282. PCIE_UNCERR_SVRTY Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED			UR_ERR_SVRTY	ECRC_ERR_SVRTY	MTLP_ERR_SVRTY	RCVR_OF_SVRTY	UCMP_SVRTY
R-0h			R/W-0h	R/W-0h	R/W-1h	R/W-1h	R/W-0h
15	14	13	12	11	10	9	8
CMPL_ABRT_SVRTY	CMPL_TMOT_SVRTY	FCP_ERR_SVRTY	PSND_TLP_SVRTY	RESERVED			
R/W-0h	R/W-0h	R/W-1h	R/W-0h	R-0h			
7	6	5	4	3	2	1	0
RESERVED		SRPS_DN_SVRTY	DLP_ERR_SVRTY	RESERVED			
R-0h		R-1h	R/W-1h	R-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3133. PCIE_UNCERR_SVRTY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-21	RESERVED	R	0h	Reads return 0 and writes have no effect.
20	UR_ERR_SVRTY	R/W	0h	Unsupported Request Error Severity • 0 = Non-fatal • 1 = Fatal
19	ECRC_ERR_SVRTY	R/W	0h	ECRC Error Severity • 0 = Non-fatal • 1 = Fatal
18	MTLP_ERR_SVRTY	R/W	1h	Malformed TLP Severity • 0 = Non-fatal • 1 = Fatal
17	RCVR_OF_SVRTY	R/W	1h	Receiver Overflow Severity • 0 = Non-fatal • 1 = Fatal
16	UCMP_SVRTY	R/W	0h	Unexpected Completion Severity • 0 = Non-fatal • 1 = Fatal
15	CMPL_ABRT_SVRTY	R/W	0h	Completer Abort Severity • 0 = Non-fatal • 1 = Fatal
14	CMPL_TMOT_SVRTY	R/W	0h	Completion Timeout Severity • 0 = Non-fatal • 1 = Fatal

Table 11-3133. PCIE_UNCERR_SVRTY Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	FCP_ERR_SVRTY	R/W	1h	Flow Control Protocol Error Severity • 0 = Non-fatal • 1 = Fatal
12	PSND_TLP_SVRTY	R/W	0h	Poisoned TLP Severity • 0 = Non-fatal • 1 = Fatal
11-6	RESERVED	R	0h	Reads return 0 and writes have no effect.
5	SRPS_DN_SVRTY	R	1h	Surprise Down Error Severity. Not supported.
4	DLP_ERR_SVRTY	R/W	1h	Data Link Protocol Error Severity • 0 = Non-fatal • 1 = Fatal
3-0	RESERVED	R	0h	Reads return 0 and writes have no effect.

Table 11-3134. Register Call Summary for PCIE_UNCERR_SVRTY

PCIe SS Functional Description • PCIe Advanced Error Reporting: [0]
PCI Express Extended Capabilities Registers • Register Summary: [0] • PCIE_UNCERR_SVRTY Register (Offset = 110Ch) [reset = 62030h]: [0]

11.14.5.10.6 PCIE_CERR Register (Offset = 1110h) [reset = 0h]

The PCI Express Correctable Error Status Register (PCIE_CERR) is shown in Figure 11-1283 and described in Table 11-3136.

Table 11-3135. PCIE_CERR Instances

Instance	Physical Address
PCIE	2180 1110h

Figure 11-1283. PCIE_CERR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		ADV_NFERR_ST	RPLY_TMR_ST	RESERVED			RPLT_RO_ST
R-0h		R/W1C-0h	R/W1C-0h	R-0h			R/W1C-0h
7	6	5	4	3	2	1	0
BAD_DLLP_ST	BAD_TLP_ST	RESERVED					RCVR_ERR_ST
R/W1C-0h	R/W1C-0h	R-0h					R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-3136. PCIE_CERR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reads return 0 and writes have no effect.
13	ADV_NFERR_ST	R/W1C	0h	Advisory Non-Fatal Error Status
12	RPLY_TMR_ST	R/W1C	0h	Replay Timer Timeout Status
11-9	RESERVED	R	0h	Reads return 0 and writes have no effect.
8	RPLT_RO_ST	R/W1C	0h	REPLAY_NUM Rollover Status
7	BAD_DLLP_ST	R/W1C	0h	Bad DLLP Status
6	BAD_TLP_ST	R/W1C	0h	Bad TLP Status
5-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	RCVR_ERR_ST	R/W1C	0h	Receiver Error Status

Table 11-3137. Register Call Summary for PCIE_CERR

PCIe SS Functional Description • PCIe Advanced Error Reporting: [0]
PCI Express Extended Capabilities Registers • PCIE_CERR Register (Offset = 1110h) [reset = 0h]: [0] • Register Summary: [0]

11.14.5.10.7 PCIE_CERR_MASK Register (Offset = 1114h) [reset = 2000h]

The PCI Express Correctable Error Mask Register (PCIE_CERR_MASK) is shown in Figure 11-1284 and described in Table 11-3139.

Table 11-3138. PCIE_CERR_MASK Instances

Instance	Physical Address
PCIE	2180 1114h

Figure 11-1284. PCIE_CERR_MASK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		ADV_NFERR_MSK	RPLY_TMR_MSK	RESERVED			RPLT_RO_MSK
R-0h		R/W-1h	R/W-0h	R-0h			R/W-0h
7	6	5	4	3	2	1	0
BAD_DLLP_MSK	BAD_TLP_MSK	RESERVED					RCVR_ERR_MSK
R/W-0h	R/W-0h	R-0h					R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3139. PCIE_CERR_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reads return 0 and writes have no effect.
13	ADV_NFERR_MSK	R/W	1h	Advisory Non Fatal Error Mask This bit is Set by default to enable compatibility with software that does not comprehend Role-Based Error Reporting.
12	RPLY_TMR_MSK	R/W	0h	Reply Timer Timeout Mask
11-9	RESERVED	R	0h	Reads return 0 and writes have no effect.
8	RPLT_RO_MSK	R/W	0h	REPLAY_NUM Rollover Mask
7	BAD_DLLP_MSK	R/W	0h	Bad DLLP Mask
6	BAD_TLP_MSK	R/W	0h	Bad TLP Mask
5-1	RESERVED	R	0h	Reads return 0 and writes have no effect.
0	RCVR_ERR_MSK	R/W	0h	Receiver Error Mask

Table 11-3140. Register Call Summary for PCIE_CERR_MASK

PCIe SS Functional Description • PCIe Advanced Error Reporting: [0]
PCI Express Extended Capabilities Registers • Register Summary: [0] • PCIE_CERR_MASK Register (Offset = 1114h) [reset = 2000h]: [0]

11.14.5.10.8 PCIE_ACCR Register (Offset = 1118h) [reset = A0h]

The PCI Express Advanced Error Capabilities and Control Register (PCIE_ACCR) is shown in Figure 11-1285 and described in Table 11-3142.

Table 11-3141. PCIE_ACCR Instances

Instance	Physical Address
PCIE	2180 1118h

Figure 11-1285. PCIE_ACCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							ECRC_CHK_EN
R-0h							R/W-0h
7	6	5	4	3	2	1	0
ECRC_CHK_CAP	ECRC_GEN_EN	ECRC_GEN_CAP	FRST_ERR_PTR				
R-1h	R/W-0h	R-1h	R-0h				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3142. PCIE_ACCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reads return 0 and writes have no effect.
8	ECRC_CHK_EN	R/W	0h	ECRC Check Enable
7	ECRC_CHK_CAP	R	1h	ECRC Check Capable
6	ECRC_GEN_EN	R/W	0h	ECRC Generation Enable • 0 = ECRC generation is disabled • 1 = ECRC generation is enabled
5	ECRC_GEN_CAP	R	1h	ECRC Generation Capability
4-0	FRST_ERR_PTR	R	0h	First Error Pointer The First Error Pointer is a field that identifies the bit position of the first error reported in the Uncorrectable Error Status register.

Table 11-3143. Register Call Summary for PCIE_ACCR

PCIe SS Functional Description • PCIe Advanced Error Reporting: [0][1]
PCI Express Extended Capabilities Registers • Register Summary: [0] • PCIE_ACCR Register (Offset = 1118h) [reset = A0h]: [0]

11.14.5.10.9 PCIE_HDR_LOG0 Register (Offset = 111Ch) [reset = 0h]

The Header Log Register 0 (PCIE_HDR_LOG0) is shown in Figure 11-1286 and described in Table 11-3145.

Table 11-3144. PCIE_HDR_LOG0 Instances

Instance	Physical Address
PCIE	2180 111Ch

Figure 11-1286. PCIE_HDR_LOG0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HDR_DW0																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3145. PCIE_HDR_LOG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HDR_DW0	R	0h	First DWORD of Header for a detected error.

Table 11-3146. Register Call Summary for PCIE_HDR_LOG0

PCIe SS Functional Description • PCIe Advanced Error Reporting: [0]
PCI Express Extended Capabilities Registers • Register Summary: [0] • PCIE_HDR_LOG0 Register (Offset = 111Ch) [reset = 0h]: [0]

11.14.5.10.10 PCIE_HDR_LOG1 Register (Offset = 1120h) [reset = 0h]

The Header Log Register 1 (PCIE_HDR_LOG1) is shown in Figure 11-1287 and described in Table 11-3148.

Table 11-3147. PCIE_HDR_LOG1 Instances

Instance	Physical Address
PCIE	2180 1120h

Figure 11-1287. PCIE_HDR_LOG1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HDR_DW1																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3148. PCIE_HDR_LOG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HDR_DW1	R	0h	Second DWORD of header for a detected error.

Table 11-3149. Register Call Summary for PCIE_HDR_LOG1

PCI Express Extended Capabilities Registers
- Register Summary: [0]
- PCIE_HDR_LOG1 Register (Offset = 1120h) [reset = 0h]: [0]

11.14.5.10.11 PCIE_HDR_LOG2 Register (Offset = 1124h) [reset = 0h]

The Header Log Register 2 (PCIE_HDR_LOG2) is shown in Figure 11-1288 and described in Table 11-3151.

Table 11-3150. PCIE_HDR_LOG2 Instances

Instance	Physical Address
PCIE	2180 1124h

Figure 11-1288. PCIE_HDR_LOG2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HDR_DW2																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3151. PCIE_HDR_LOG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HDR_DW2	R	0h	Third DWORD of header for a detected error.

Table 11-3152. Register Call Summary for PCIE_HDR_LOG2

PCI Express Extended Capabilities Registers
- PCIE_HDR_LOG2 Register (Offset = 1124h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.10.12 PCIE_HDR_LOG3 Register (Offset = 1128h) [reset = 0h]

The Header Log Register 3 (PCIE_HDR_LOG3) is shown in Figure 11-1289 and described in Table 11-3154.

Table 11-3153. PCIE_HDR_LOG3 Instances

Instance	Physical Address
PCIE	2180 1128h

Figure 11-1289. PCIE_HDR_LOG3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HDR_DW3																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3154. PCIE_HDR_LOG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HDR_DW3	R	0h	Fourth DWORD of Header for a detected error.

Table 11-3155. Register Call Summary for PCIE_HDR_LOG3

PCIe SS Functional Description • PCIe Advanced Error Reporting: [0]
PCI Express Extended Capabilities Registers • PCIE_HDR_LOG3 Register (Offset = 1128h) [reset = 0h]: [0] • Register Summary: [0]

11.14.5.10.13 PCIE_RC_ERR_CMD Register (Offset = 112Ch) [reset = 0h]

The Root Error Command Register (PCIE_RC_ERR_CMD) is shown in Figure 11-1290 and described in Table 11-3157.

Table 11-3156. PCIE_RC_ERR_CMD Instances

Instance	Physical Address
PCIE	2180 112Ch

Figure 11-1290. PCIE_RC_ERR_CMD Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					FERR_RPT_EN	NFERR_RPT_EN	CERR_RPT_EN
R-0h					R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3157. PCIE_RC_ERR_CMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reads return 0 and writes have no effect.
2	FERR_RPT_EN	R/W	0h	Fatal Error Reporting Enable. • 0 = Error reporting is disabled • 1 = Error reporting is enabled
1	NFERR_RPT_EN	R/W	0h	Nonfatal Error Reporting Enable. • 0 = Error reporting is disabled • 1 = Error reporting is enabled
0	CERR_RPT_EN	R/W	0h	Correctable Error Reporting Enable. • 0 = Error reporting is disabled • 1 = Error reporting is enabled

Table 11-3158. Register Call Summary for PCIE_RC_ERR_CMD

PCIe SS Functional Description • Advanced Error Reporting Interrupt: [0] • PCIe Advanced Error Reporting: [0]
PCI Express Extended Capabilities Registers • Register Summary: [0] • PCIE_RC_ERR_CMD Register (Offset = 112Ch) [reset = 0h]: [0]

11.14.5.10.14 PCIE_RC_ERR_ST Register (Offset = 1130h) [reset = 0h]

The Root Error Status Register (PCIE_RC_ERR_ST) is shown in Figure 11-1291 and described in Table 11-3160.

Table 11-3159. PCIE_RC_ERR_ST Instances

Instance	Physical Address
PCIE	2180 1130h

Figure 11-1291. PCIE_RC_ERR_ST Register

31	30	29	28	27	26	25	24
AER_INT_MSG					RESERVED		
R-0h					R-0h		
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	FERR_RCV	NFERR	UNCOR_FATAL	MULT_FNF	ERR_FNF	MULT_COR	CORR_ERR
R-0h	R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; *-n* = value after reset

Table 11-3160. PCIE_RC_ERR_ST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	AER_INT_MSG	R	0h	AER Interrupt Message Number. Writable through internal bus interface.
26-7	RESERVED	R	0h	Reads return 0 and writes have no effect.
6	FERR_RCV	R	0h	Fatal Error Messages Received.
5	NFERR	R/W1C	0h	Non-Fatal Error Messages Received.
4	UNCOR_FATAL	R/W1C	0h	First Uncorrectable Fatal Received.
3	MULT_FNF	R/W1C	0h	Multiple Uncorrectable Error (ERR_FATAL/NONFATAL) Received.
2	ERR_FNF	R/W1C	0h	Uncorrectable Error (ERR_FATAL/NONFATAL) Received.
1	MULT_COR	R/W1C	0h	Multiple Correctable Error (ERR_COR) Received.
0	CORR_ERR	R/W1C	0h	Correctable Error (ERR_COR) Received.

Table 11-3161. Register Call Summary for PCIE_RC_ERR_ST

PCIe SS Functional Description • PCIe Advanced Error Reporting: [0]
PCI Express Extended Capabilities Registers • Register Summary: [0] • PCIE_RC_ERR_ST Register (Offset = 1130h) [reset = 0h]: [0]

11.14.5.10.15 PCIE_ERR_SRC_ID Register (Offset = 1134h) [reset = 0h]

The Error Source Identification Register (PCIE_ERR_SRC_ID) is shown in and described in .

Table 11-3162. PCIE_ERR_SRC_ID Instances

Instance	Physical Address
PCIE	2180 1134h

Figure 11-1292. PCIE_ERR_SRC_ID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FNF_SRC_ID																CORR_SRC_ID															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3163. PCIE_ERR_SRC_ID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	FNF_SRC_ID	R	0h	Fatal or Non-Fatal error source identification
15-0	CORR_SRC_ID	R	0h	Correctable error source identification

Table 11-3164. Register Call Summary for PCIE_ERR_SRC_ID

PCIe SS Functional Description • PCIe Advanced Error Reporting: [0]
PCI Express Extended Capabilities Registers • Register Summary: [0] • PCIE_ERR_SRC_ID Register (Offset = 1134h) [reset = 0h]: [0]

11.14.5.11 Port Logic Registers

11.14.5.11.1 Register Summary

Table 11-3165. PCIe Instances

Instance	Base Address
PCIE	2180 0000h

Table 11-3166. Port Logic Registers

Offset	Acronym	Register Name	PCIE Physical Address	Section
1700h	PCIE_PL_ACKTIMER	Ack Latency Time and Replay Timer	2180 1700h	Section 11.14.5.11.2
1704h	PCIE_PL_OMSG	Other Message Register	2180 1704h	Section 11.14.5.11.3
1708h	PCIE_PL_FORCE_LINK	Port Force Link Register	2180 1708h	Section 11.14.5.11.4
170Ch	PCIE_ACK_FREQ	Ack Frequency Register	2180 170Ch	Section 11.14.5.11.5
1710h	PCIE_PL_LINK_CTRL	Port Link Control Register	2180 1710h	Section 11.14.5.11.6
1714h	PCIE_LANE_SKEW	Lane Skew Register	2180 1714h	Section 11.14.5.11.7
1718h	PCIE_SYM_NUM	Symbol Number Register	2180 1718h	Section 11.14.5.11.8
171Ch	PCIE_SYMTIMER_FLTMASK	Symbol Timer and Filter Mask Register	2180 171Ch	Section 11.14.5.11.9
1720h	PCIE_FLT_MASK2	Filter Mask Two register	2180 1720h	Section 11.14.5.11.10
1728h	PCIE_DEBUG0	Debug 0 register	2180 1728h	Section 11.14.5.11.11
172Ch	PCIE_DEBUG1	Debug 1 register	2180 172Ch	Section 11.14.5.11.12
180Ch	PCIE_PL_GEN2	Gen2 Register	2180 180Ch	Section 11.14.5.11.13

11.14.5.11.2 PCIE_PL_ACKTIMER Register (Offset = 1700h) [reset = C00040h]

The Ack Latency Time and Replay Timer (PCIE_PL_ACKTIMER) is shown in Figure 11-1293 and described in Table 11-3168.

Table 11-3167. PCIE_PL_ACKTIMER Instances

Instance	Physical Address
PCIE	2180 1700h

Figure 11-1293. PCIE_PL_ACKTIMER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RPLY_LIMT																RND_TRP_LMT															
R/W-C0h																R/W-40h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3168. PCIE_PL_ACKTIMER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RPLY_LIMT	R/W	C0h	Replay Time Limit. The replay timer expires when it reaches this limit.
15-0	RND_TRP_LMT	R/W	40h	Round Trip Latency Time Limit. The Ack/Nak latency timer expires when it reaches this limit.

Table 11-3169. Register Call Summary for PCIE_PL_ACKTIMER

Port Logic Registers
- Register Summary: [0]
- PCIE_PL_ACKTIMER Register (Offset = 1700h) [reset = C00040h]: [0]

11.14.5.11.3 PCIE_PL_OMSG Register (Offset = 1704h) [reset = FFFFFFFFh]

The Other Message Register (PCIE_PL_OMSG) is shown in Figure 11-1294 and described in Table 11-3171.

Table 11-3170. PCIE_PL_OMSG Instances

Instance	Physical Address
PCIE	2180 1704h

Figure 11-1294. PCIE_PL_OMSG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OMSG																															
R/W-FFFFFFFFh																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3171. PCIE_PL_OMSG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	OMSG	R/W	FFFFFFFFh	Other Message Register. It can be used to send a specific PCI Express message in which case this register is programmed with the payload and bit 0 of Port Link Control Register is set to transmit the message.

Table 11-3172. Register Call Summary for PCIE_PL_OMSG

Port Logic Registers

- PCIE_PL_OMSG Register (Offset = 1704h) [reset = FFFFFFFFh]: [0]
- Register Summary: [0]
- PCIE_PL_LINK_CTRL Register (Offset = 1710h) [reset = 30120h]: [0]

11.14.5.11.4 PCIE_PL_FORCE_LINK Register (Offset = 1708h) [reset = 7000004h]

The Port Force Link Register (PCIE_PL_FORCE_LINK) is shown in Figure 11-1295 and described in Table 11-3174.

Table 11-3173. PCIE_PL_FORCE_LINK Instances

Instance	Physical Address
PCIE	2180 1708h

Figure 11-1295. PCIE_PL_FORCE_LINK Register

31	30	29	28	27	26	25	24
LPE_CNT							
R/W-7h							
23	22	21	20	19	18	17	16
RESERVED		LINK_STATE					
R-0h		R/W-0h					
15	14	13	12	11	10	9	8
FORCE_LINK	RESERVED						
W1S-0h	R-0h						
7	6	5	4	3	2	1	0
LINK_NUM							
R/W-4h							

LEGEND: R = Read Only; R/W = Read/Write; W1S = Write 1 to Set Bit; -*n* = value after reset

Table 11-3174. PCIE_PL_FORCE_LINK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	LPE_CNT	R/W	7h	Low Power Entrance Count
23-22	RESERVED	R	0h	Reads return 0 and writes have no effect.
21-16	LINK_STATE	R/W	0h	Link State. The link state that the PCIe will be forced to when FORCE_LINK field is set. Please see for LTSSM states encoded values.
15	FORCE_LINK	W1S	0h	Force Link. Forces the link to the state specified by the LINK_STATE field. The Force Link pulse will trigger link re-negotiation.
14-8	RESERVED	R	0h	Reads return 0 and writes have no effect.
7-0	LINK_NUM	R/W	4h	Link Number. Not used for EP.

Table 11-3175. Register Call Summary for PCIE_PL_FORCE_LINK

Port Logic Registers • PCIE_PL_FORCE_LINK Register (Offset = 1708h) [reset = 7000004h]: [0] • Register Summary: [0]
PCIe SS Functional Description • PHY Loopback: [0][1][2][3] • Loopback Master: [0][1]

11.14.5.11.5 PCIE_ACK_FREQ Register (Offset = 170Ch) [reset = 1B0F6400h]

The Ack Frequency Register (PCIE_ACK_FREQ) is shown in Figure 11-1296 and described in Table 11-3177.

Table 11-3176. PCIE_ACK_FREQ Instances

Instance	Physical Address
PCIE	2180 170Ch

Figure 11-1296. PCIE_ACK_FREQ Register

31	30	29	28	27	26	25	24
RESERVED	ASPM_L1	L1_ENTRY_LATENCY			L0S_ENTRY_LATENCY		
R-0h	R/W-0h	R/W-3h			R/W-3h		
23	22	21	20	19	18	17	16
COMM_NFTS							
R/W-Fh							
15	14	13	12	11	10	9	8
NFTS							
R/W-64h							
7	6	5	4	3	2	1	0
ACK_FREQ							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3177. PCIE_ACK_FREQ Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reads return 0 and writes have no effect.
30	ASPM_L1	R/W	0h	Set to allow entering ASPM L1 even when link partner did not to L0s. When cleared, the ASPM L1 state is entered only after idle period during which both RX and TX are in L0s.
29-27	L1_ENTRY_LATENCY	R/W	3h	L1 entrance latency. The latency is set to 2^L1_ENTRY_LATENCY microseconds with the max being 64 microseconds. • 0 = 1 µs • 1h = 2 µs • 2h = 4 µs • 3h = 8 µs • 4h = 16 µs • 5h = 32 µs • 6h or 7h = 64 µs
26-24	L0S_ENTRY_LATENCY	R/W	3h	L0s entrance latency. The latency is set to L0S_ENTRY_LATENCY+1 microseconds. Maximum is 7 microseconds. • 0 = 1 µs • 1h = 2 µs • 2h = 3 µs • 3h = 4 µs • 4h = 5 µs • 5h = 6 µs • 6h or 7h = 7 µs
23-16	COMM_NFTS	R/W	Fh	Number of fast training sequences when common clock is used and when transitioning from L0s to L0.
15-8	NFTS	R/W	64h	Number of fast training sequences to be transmitted when transitioning from L0s to L0. Value of 0is not supported.

Table 11-3177. PCIE_ACK_FREQ Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-0	ACK_FREQ	R/W	0h	Ack Frequency. The value range is 0x0-0xFF. A value of 0 in the ACK Frequency field indicates that this ACK frequency control feature is disabled. A value of N (N>0) indicates that the module will ack N TLPs received by sending an ACK DLLP.

Table 11-3178. Register Call Summary for PCIE_ACK_FREQ

Port Logic Registers • PCIE_ACK_FREQ Register (Offset = 170Ch) [reset = 1B0F6400h]: [0] • Register Summary: [0]
PCIe SS Functional Description • L0s State: [0] • ASPM L1 State: [0]

11.14.5.11.6 PCIE_PL_LINK_CTRL Register (Offset = 1710h) [reset = 30120h]

The Port Link Control Register (PCIE_PL_LINK_CTRL) is shown in Figure 11-1297 and described in Table 11-3180.

Table 11-3179. PCIE_PL_LINK_CTRL Instances

Instance	Physical Address
PCIE	2180 1710h

Figure 11-1297. PCIE_PL_LINK_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED		LNK_MODE					
R-0h		R/W-3h					
15	14	13	12	11	10	9	8
RESERVED				LINK_RATE			
R-0h				R/W-1h			
7	6	5	4	3	2	1	0
FLNK_MODE	RESERVED	DLL_EN	RESERVED	RST_ASRT	LPBK_EN	SCRM_DIS	OMSG_REQ
R/W-0h	R-0h	R/W-1h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3180. PCIE_PL_LINK_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reads return 0 and writes have no effect.
21-16	LNK_MODE	R/W	3h	Link Mode Enable. (×N – corresponding to N Lanes). • 1h = ×1 • 3h = ×2 • 7h = ×4 • Fh = ×8 • 1Fh = ×16 • 3Fh = ×32 • Others = Reserved
15-12	RESERVED	R	0h	Reads return 0 and writes have no effect.
11-8	LINK_RATE	R/W	1h	Default link rate. For 2.5 GT/s it is 0x1. This register does not affect any functionality.
7	FLNK_MODE	R/W	0h	Fast link mode. Set all internal timers to fast mode for simulation purposes.
6	RESERVED	R	0h	Reads return 0 and writes have no effect.
5	DLL_EN	R/W	1h	DLL Link Enable. Enable link initialization.
4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3	RST_ASRT	R/W	0h	Reset assert. Triggers a recovery and forces the LTSSM to the Hot Reset state. Downstream ports (RC ports) only.
2	LPBK_EN	R/W	0h	Loopback Enable. Turn on loopback.
1	SCRM_DIS	R/W	0h	Scramble Disable. Turn off data scrambling.
0	OMSG_REQ	R/W	0h	Other message request. Set to transmit the message contained in the Other Message Register (PCIE_PL_OMSG).

Table 11-3181. Register Call Summary for PCIE_PL_LINK_CTRL

Port Logic Registers • Register Summary: [0] • PCIE_PL_LINK_CTRL Register (Offset = 1710h) [reset = 30120h]: [0]
PCIe SS Functional Description • Loopback Master: [0][1]

11.14.5.11.7 PCIE_LANE_SKEW Register (Offset = 1714h) [reset = 0h]

The Lane Skew Register (PCIE_LANE_SKEW) is shown in Figure 11-1298 and described in Table 11-3183.

Table 11-3182. PCIE_LANE_SKEW Instances

Instance	Physical Address
PCIE	2180 1714h

Figure 11-1298. PCIE_LANE_SKEW Register

31	30	29	28	27	26	25	24
L2L_DESKEW	RESERVED					ACK_DISABLE	FC_DISABLE
R/W-0h	R-0h					R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
LANE_SKEW							
R/W-0h							
15	14	13	12	11	10	9	8
LANE_SKEW							
R/W-0h							
7	6	5	4	3	2	1	0
LANE_SKEW							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3183. PCIE_LANE_SKEW Register Field Descriptions

Bit	Field	Type	Reset	Description
31	L2L_DESKEW	R/W	0h	Disable Lane to Lane Deskew.
30-26	RESERVED	R	0h	Reads return 0 and writes have no effect.
25	ACK_DISABLE	R/W	0h	Disable Ack and Nak DLLP transmission.
24	FC_DISABLE	R/W	0h	Flow Control Disable. Set to disable transmission of Flow Control DLLPs.
23-0	LANE_SKEW	R/W	0h	Insert Lane Skew for Transmit. The value is in units of one symbol time. Thus a value 0x02 will force a skew of two symbol times for that lane. Max allowed is 5 symbol times. This 24 bit field is used for programming skew for eight lanes with three bits per lane.

Table 11-3184. Register Call Summary for PCIE_LANE_SKEW

Port Logic Registers

- PCIE_LANE_SKEW Register (Offset = 1714h) [reset = 0h]: [0]
- Register Summary: [0]

11.14.5.11.8 PCIE_SYM_NUM Register (Offset = 1718h) [reset = 103AAh]

The Symbol Number Register (PCIE_SYM_NUM) is shown in Figure 11-1299 and described in Table 11-3186.

Table 11-3185. PCIE_SYM_NUM Instances

Instance	Physical Address
PCIE	2180 1718h

Figure 11-1299. PCIE_SYM_NUM Register

31	30	29	28	27	26	25	24
MAX_FUNC			FCWATCH_TIMER				
R/W-0h			R/W-0h				
23	22	21	20	19	18	17	16
ACK_LATENCY_TIMER					REPLAY_TIMER		
R/W-0h					R/W-4h		
15	14	13	12	11	10	9	8
REPLAY_TIMER		RESERVED			SKP_COUNT		
R/W-4h		R-0h			R/W-3h		
7	6	5	4	3	2	1	0
NUM_TS2_SYMBOLS				TS_COUNT			
R/W-Ah				R/W-Ah			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3186. PCIE_SYM_NUM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	MAX_FUNC	R/W	0h	Configuration requests targeted at function numbers above this value will result in UR response.
28-24	FCWATCH_TIMER	R/W	0h	Timer Modifier for Flow Control Watchdog Timer. Increases the timer value for Flow Control watchdog timer in increments of 16 clock cycles.
23-19	ACK_LATENCY_TIMER	R/W	0h	Timer Modifier for Ack/Nak Latency Timer. Increases the timer value for the Ack/Nak latency timer in increments of 64 clock cycles.
18-14	REPLAY_TIMER	R/W	4h	Timer Modifier for Replay Timer. Increases the timer value for the replay timer in increments of 64 clock cycles.
13-11	RESERVED	R	0h	Reads return 0 and writes have no effect.
10-8	SKP_COUNT	R/W	3h	Number of SKP Symbols.
7-4	NUM_TS2_SYMBOLS	R/W	Ah	Number of TS2 Symbols. This field does not affect any functionality.
3-0	TS_COUNT	R/W	Ah	Number of TS Symbols. Set the number of TS identifier symbols that are sent in TS1 and TS2 ordered sets.

Table 11-3187. Register Call Summary for PCIE_SYM_NUM

Port Logic Registers
- Register Summary: [0]
- PCIE_SYM_NUM Register (Offset = 1718h) [reset = 103AAh]: [0]

11.14.5.11.9 PCIE_SYMTIMER_FLTMASK Register (Offset = 171Ch) [reset = 500h]

The Symbol Timer and Filter Mask Register (PCIE_SYMTIMER_FLTMASK) is shown in Figure 11-1300 and described in Table 11-3189.

Table 11-3188. PCIE_SYMTIMER_FLTMASK Instances

Instance	Physical Address
PCIE	2180 171Ch

Figure 11-1300. PCIE_SYMTIMER_FLTMASK Register

31	30	29	28	27	26	25	24
F1_CFG_DROP	F1_IO_DROP	F1_MSG_DROP	F1_CPL_ECRC_DROP	F1_ECRC_DROP	F1_CPL_LEN_TEST	F1_CPL_ATTR_TEST	F1_CPL_TC_TEST
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
F1_CPL_FUNC_TEST	F1_CPL_REQID_TEST	F1_CPL_TAGERR_TEST	F1_LOCKED_RD_AS_UR	F1_CFG1_RE_AS_US	F1_UR_OUT_OF_BAR	F1_UR_POISON	F1_UR_FUN_MISMATCH
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
FC_WDOG_DISABLE	RESERVED				SKP_VALUE		
R/W-0h	R-0h				R/W-500h		
7	6	5	4	3	2	1	0
SKP_VALUE							
R/W-500h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3189. PCIE_SYMTIMER_FLTMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	F1_CFG_DROP	R/W	0h	Set to allow CFG TLPs on RC • 0 = Not allow CFG transaction being received on RC. • 1 = Allow CFG transaction being received on RC.
30	F1_IO_DROP	R/W	0h	Set to allow IO TLPs on RC • 0 = Not allow IO transaction being received on RC. • 1 = Allow IO transaction being received on RC.
29	F1_MSG_DROP	R/W	0h	Set to allow MSG TLPs on RC • 0 = Not allow MSG transaction being received on RC. • 1 = Allow MSG transaction being received on RC.
28	F1_CPL_ECRC_DROP	R/W	0h	Set to allow Completion TLPs with ECRC to pass up • 0 = Discard completion TLPs with ECRC errors. • 1 = Allow completion TLPs with ECRC errors to be passed up.
27	F1_ECRC_DROP	R/W	0h	Set to allow TLPs with ECRC error to pass up • 0 = Discard TLPs with ECRC errors. • 1 = Allow TLPs with ECRC errors to be passed up.
26	F1_CPL_LEN_TEST	R/W	0h	Set to mask length match for received completion TLPs • 0 = Enforce length match for received completion TLPs. • 1 = Mask length match for received completion TLPs.
25	F1_CPL_ATTR_TEST	R/W	0h	Set to mask attribute match on received completion TLPs • 0 = Enforce attribute match for received completion TLPs. • 1 = Mask attribute match on received completion TLPs.
24	F1_CPL_TC_TEST	R/W	0h	Set to mask traffic match on received completion TLPs • 0 = Enforce traffic class match for received completion TLPs. • 1 = Mask traffic class match on received completion TLPs.

Table 11-3189. PCIE_SYMTIMER_FLTMASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	F1_CPL_FUNC_TEST	R/W	0h	Set to mask function match on received completion TLPs • 0 = Enforce function match for received completion TLPs. • 1 = Mask function match for received completion TLPs.
22	F1_CPL_REQID_TEST	R/W	0h	Set to mask request ID match on received completion TLPs • 0 = Enforce request ID match for received completion TLPs. • 1 = Mask request ID match for received completion TLPs.
21	F1_CPL_TAGERR_TEST	R/W	0h	Set to mask tag error rules for received completion TLPs • 0 = Enforce tag error rules for received completion TLPs. • 1 = Mask tag error rules for received completion TLPs.
20	F1_LOCKED_RD_AS_UR	R/W	0h	Set to treat locked read TLPs as supported for EP, UR for RC • 0 = Treat locked read TLPs as UR for EP, supported for RC. • 1 = Treat locked read TLPs as supported for EP, UR for RC.
19	F1_CFG1_RE_AS_US	R/W	0h	Set to treat type 1 CFG TLPs as supported for EP, UR for RC • 0 = Treat type 1 CFG TLPs as UR for EP and supported for RC. • 1 = Treat type 1 CFG TLPs as supported for EP and UR for RC.
18	F1_UR_OUT_OF_BAR	R/W	0h	Set to treat out-of-BAR TLPs as supported requests • 0 = Treat out-of-BAR TLPs as UR. • 1 = Treat out-of-BAR TLPs as supported requests.
17	F1_UR_POISON	R/W	0h	Set to treat poisoned TLPs as supported requests • 0 = Treat poisoned TLPs as UR. • 1 = Treat poisoned TLPs as supported requests.
16	F1_UR_FUN_MISMATCH	R/W	0h	Set to treat function mismatched TLPs as supported requests • 0 = Treat function mismatched TLPs as UR. • 1 = Treat function mismatched TLPs as supported requests.
15	FC_WDOG_DISABLE	R/W	0h	Set to disable FC watchdog timer • 0 = Enable Flow Control watchdog timer. • 1 = Disable Flow Control watchdog timer.
14-11	RESERVED	R	0h	Reads return 0 and writes have no effect.
10-0	SKP_VALUE	R/W	500h	Number of symbol times to wait between transmitting SKP ordered sets. For example, for a setting of 1536 decimal, the wait will be for 1537 symbol times.

Table 11-3190. Register Call Summary for PCIE_SYMTIMER_FLTMASK

Port Logic Registers
- Register Summary: [0]
- PCIE_SYMTIMER_FLTMASK Register (Offset = 171Ch) [reset = 500h]: [0]

11.14.5.11.10 PCIE_FLT_MASK2 Register (Offset = 1720h) [reset = 0h]

The Filter Mask Register 2 (PCIE_FLT_MASK2) is shown in Figure 11-1301 and described in Table 11-3192.

Table 11-3191. PCIE_FLT_MASK2 Instances

Instance	Physical Address
PCIE	2180 1720h

Figure 11-1301. PCIE_FLT_MASK2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				FLUSH_REQ	DLLP_ABORT	VMSG1_DROP	VMSG0_DROP
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3192. PCIE_FLT_MASK2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reads return 0 and writes have no effect.
3	FLUSH_REQ	R/W	0h	Set to enable the filter to handle flush request • 0 = Disable the filter to handle flush request. • 1 = Enable the filter to handle flush request.
2	DLLP_ABORT	R/W	0h	Set to disable DLLP abort for unexpected CPL • 0 = Enable DLLP abort for unexpected CPL. • 1 = Disable DLLP abort for unexpected CPL.
1	VMSG1_DROP	R/W	0h	Set to disable dropping of Vendor MSG Type 1 • 0 = Enable dropping of Vendor MSG Type 1. It will be passed to internal bus interface. • 1 = Disable dropping of Vendor MSG Type 1.
0	VMSG0_DROP	R/W	0h	Set to disable dropping of Vendor MSG Type 0 with UR reporting • 0 = Enable dropping of Vendor MSG Type 0 with UR error reporting. It will be passed to internal bus interface. • 1 = Disable dropping of Vendor MSG Type 0 with UR error reporting.

Table 11-3193. Register Call Summary for PCIE_FLT_MASK2

Port Logic Registers
- Register Summary: [0]
- PCIE_FLT_MASK2 Register (Offset = 1720h) [reset = 0h]: [0]

11.14.5.11.11 PCIE_DEBUG0 Register (Offset = 1728h) [reset = 0h]

The Debug 0 Register (PCIE_DEBUG0) is shown in Figure 11-1302 and described in Table 11-3195.

Table 11-3194. PCIE_DEBUG0 Instances

Instance	Physical Address
PCIE	2180 1728h

Figure 11-1302. PCIE_DEBUG0 Register

31	30	29	28	27	26	25	24
TS_LINK_CTRL				TS_LANE_K237	TS_LINK_K237	RCVD_IDLE0	RCVD_IDLE1
R-0h				R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
PIPE_TXDATA							
R-0h							
15	14	13	12	11	10	9	8
PIPE_TXDATA							
R-0h							
7	6	5	4	3	2	1	0
PIPE_TXDATAK		TXB_SKIP_TX	LTSSM_STATE				
R-0h		R-0h	R-0h				

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3195. PCIE_DEBUG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	TS_LINK_CTRL	R	0h	Link control bits advertised by link partner.
27	TS_LANE_K237	R	0h	Currently receiving k237 (PAD) in place of lane number.
26	TS_LINK_K237	R	0h	Currently receiving k237 (PAD) in place of link number.
25	RCVD_IDLE0	R	0h	Receiver is receiving logical idle.
24	RCVD_IDLE1	R	0h	2nd symbol is also idle (16bit PHY interface only).
23-8	PIPE_TXDATA	R	0h	PIPE Transmit data. Reset value is zero but changes at every clock after that.
7-6	PIPE_TXDATAK	R	0h	PIPE transmit K indication.
5	TXB_SKIP_TX	R	0h	A skip ordered set has been transmitted.
4-0	LTSSM_STATE	R	0h	LTSSM current state. Please see for the names of the LTSSM states corresponding to the encoded values.

Table 11-3196. Register Call Summary for PCIE_DEBUG0

Port Logic Registers • Register Summary: [0] • PCIE_DEBUG0 Register (Offset = 1728h) [reset = 0h]: [0]
PCIe SS Functional Description • Initialization Sequence: [0] • Initialization Sequence: [0]

11.14.5.11.12 PCIE_DEBUG1 Register (Offset = 172Ch) [reset = 8200000h]

The Debug 1 Register (PCIE_DEBUG1) is shown in Figure 11-1303 and described in Table 11-3198.

Table 11-3197. PCIE_DEBUG1 Instances

Instance	Physical Address
PCIE	2180 172Ch

Figure 11-1303. PCIE_DEBUG1 Register

31	30	29	28	27	26	25	24
SCRAMBLER_DISABLE	LINK_DISABLE	LINK_IN_TRAINING	RCVR_REVRS_POL_EN	TRAINING_RST_N	RESERVED		
R-0h	R-0h	R-0h	R-0h	R-1h	R-0h		
23	22	21	20	19	18	17	16
RESERVED	PIPE_TXDETECTRX_LB	PIPE_TXELECIDLE	PIPE_TXCOMPLIANCE	APP_INIT_RST	RESERVED		
R-0h	R-0h	R-0h	R-1h	R-0h	R-0h		
15	14	13	12	11	10	9	8
RMLH_TS_LINK_NUM							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			XMLH_LINK_UP	RMLH_INSKIP_RCV	RMLH_TS1_RCVD	RMLH_TS2_RCVD	RMLH_RCVD_LANE_REV
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3198. PCIE_DEBUG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SCRAMBLER_DISABLE	R	0h	Scrambling disabled for the link.
30	LINK_DISABLE	R	0h	LTSSM in DISABLE state. Link inoperable.
29	LINK_IN_TRAINING	R	0h	LTSSM performing link training.
28	RCVR_REVRS_POL_EN	R	0h	LTSSM testing for polarity reversal.
27	TRAINING_RST_N	R	1h	LTSSM-negotiated link reset.
26-23	RESERVED	R	0h	Reads return 0 and writes have no effect.
22	PIPE_TXDETECTRX_LB	R	0h	PIPE receiver detect/loopback request.
21	PIPE_TXELECIDLE	R	1h	PIPE transmit electrical idle request.
20	PIPE_TXCOMPLIANCE	R	0h	PIPE transmit compliance request.
19	APP_INIT_RST	R	0h	Application request to initiate training reset.
18-16	RESERVED	R	0h	Reads return 0 and writes have no effect.
15-8	RMLH_TS_LINK_NUM	R	0h	Link number advertised/confirmed by link partner.
7-5	RESERVED	R	0h	Reads return 0 and writes have no effect.
4	XMLH_LINK_UP	R	0h	LTSSM reports PHY link up.
3	RMLH_INSKIP_RCV	R	0h	Receiver reports skip reception.
2	RMLH_TS1_RCVD	R	0h	TS1 training sequence received (pulse).
1	RMLH_TS2_RCVD	R	0h	TS2 training sequence received (pulse).
0	RMLH_RCVD_LANE_REV	R	0h	Receiver detected lane reversal.

Table 11-3199. Register Call Summary for PCIE_DEBUG1

Port Logic Registers
• Register Summary: [0] • PCIE_DEBUG1 Register (Offset = 172Ch) [reset = 8200000h]: [0]

11.14.5.11.13 PCIE_PL_GEN2 Register (Offset = 180Ch) [reset = 20Fh]

The Gen2 Register (PCIE_PL_GEN2) is shown in Figure 11-1304 and described in Table 11-3201.

Table 11-3200. PCIE_PL_GEN2 Instances

Instance	Physical Address
PCIE	2180 180Ch

Figure 11-1304. PCIE_PL_GEN2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED			DEEMPH	CFG_TX_CMPL	CFG_TX_SWING	DIR_SPD	LN_EN
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-2h
15	14	13	12	11	10	9	8
LN_EN							
R/W-2h							
7	6	5	4	3	2	1	0
NUM_FTS							
R/W-Fh							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3201. PCIE_PL_GEN2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-21	RESERVED	R	0h	Reads return 0 and writes have no effect.
20	DEEMPH	R/W	0h	Set de-emphasis level for upstream ports (EP ports).
19	CFG_TX_CMPL	R/W	0h	Configure TX compliance receive bit. When set to 1, signals LTSSM to transmit TS ordered sets with the compliance receive bit assert (equal to 1).
18	CFG_TX_SWING	R/W	0h	Configure PHY TX Swing. Indicates the voltage level the PHY should drive. • 0 = Full Swing • 1 = Low Swing
17	DIR_SPD	R/W	0h	Directed Speed Change. • 0 = Indicates to the LTSSM not to initiate a speed change to Gen2 after the link is initialized at Gen1 speed. • 1 = Indicates to the LTSSM to initiate a speed change to Gen2 after the link is initialized at Gen1 speed.
16-8	LN_EN	R/W	2h	Lane Enable. • 1h = ×1 • Others = Reserved.
7-0	NUM_FTS	R/W	Fh	Number of fast training sequences.

Table 11-3202. Register Call Summary for PCIE_PL_GEN2

Port Logic Registers • Register Summary: [0] • PCIE_PL_GEN2 Register (Offset = 180Ch) [reset = 20Fh]: [0]
PCIe SS Functional Description • Reference Clock Multiplication: [0]

11.15 Quad Serial Peripheral Interface (QSPI)

This section describes the Quad Serial Peripheral Interface (QSPI™) module for the device.

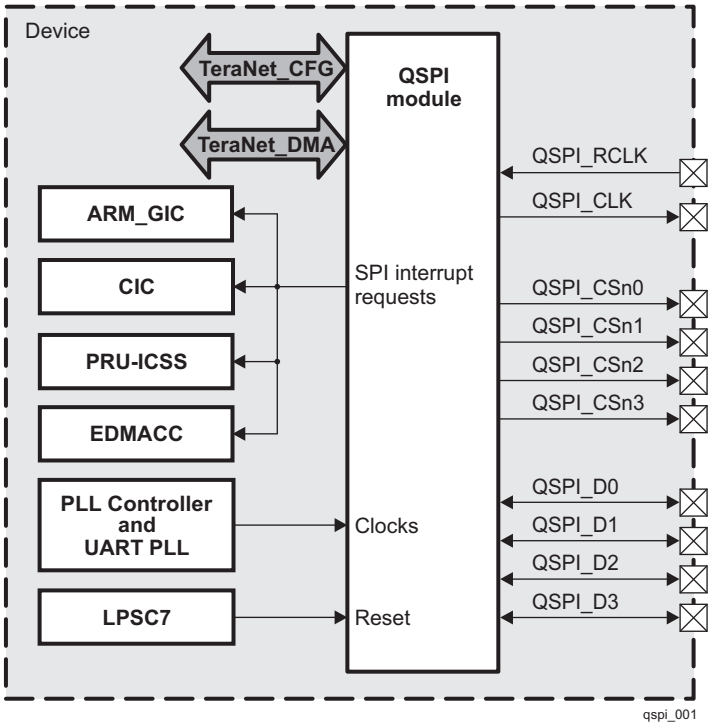
NOTE: This section contains information that is Copyright ©2010-2015 Cadence Design Systems, Inc. All rights reserved.

11.15.1 QSPI Overview

The Quad Serial Peripheral Interface (QSPI™) module (see Figure 11-1305) is a kind of Serial Peripheral Interface (SPI) module which allows single, dual or quad read and write access to external flash devices. This module has a memory mapped register interface, which provides a direct memory interface for accessing data from external flash devices, simplifying software requirements.

The QSPI module is used to transfer data, either in a memory mapped direct mode (for example a processor wishing to execute code directly from external flash memory), or in an indirect mode where the module is set-up to silently perform some requested operation, signaling its completion via interrupts or status registers. For indirect operations, data is transferred between system memory and external flash memory via an internal SRAM which is loaded for writes and unloaded for reads by a device master at low latency system speeds. Interrupts or status registers are used to identify the specific times at which this SRAM should be accessed using user programmable configuration registers.

Figure 11-1305. QSPI Module



The QSPI module has the following features:

- Memory mapped 'direct' mode of operation for performing flash data transfers and executing code from flash memory.
- Software triggered 'indirect' mode of operation for performing low latency and non-processor intensive flash data transfers.
- Local SRAM to reduce bus overhead and buffer flash data during indirect transfers.
- Set of software accessible flash control registers to perform any flash command, including data transfers up to 8-bytes at a time.
- Supports any device clock frequency, including frequencies of 96 MHz (QSPI mode 0 only).

- Supports XIP (Execute in Place), also referred to as continuous mode.
- Supports single, dual or quad I/O instructions.
- Supports 16/32/64 byte cacheline wrap accesses.
- Supports ECC for its internal SRAM buffer.
- Programmable device sizes.
- Programmable write protected regions to block system writes from taking effect.
- Programmable delays between transactions.
- Legacy mode allowing software direct access to low level transmit and receive FIFOs bypassing the higher layer processes.
- Independent reference clock to decouple bus clock from SPI clock – allows slow system clocks.
- Serial clock with programmable polarity.
- Programmable baud rate generator to generate QSPI clocks.
- Features included to improve high speed read data capture mechanism.
- Option to use adapted clocks to further improve read data capturing.
- Programmable interrupt generation.
- Up to four external chip selects.
- Supports Little-endian operation only.

The following features are not supported:

- DMA peripheral interface.
- High speed performance (greater than 50 MHz) for QSPI modes 1, 2 and 3.
- Continuous addressing mode for each of the connected devices and auto-detection of boundaries between them.

NOTE: The QSPI module does not provide any busy or idle status. Software needs to ensure that the QSPI module is idle before clocks can be shut off. Otherwise a lockup condition may occur.

11.15.2 QSPI Environment

The QSPI module is primarily intended for fast booting from Quad-SPI flash memories. Figure 11-1306 shows a typical connection of the QSPI module to an external Quad-SPI flash memory.

Figure 11-1306. QSPI Connected to an External Quad-SPI Flash Memory

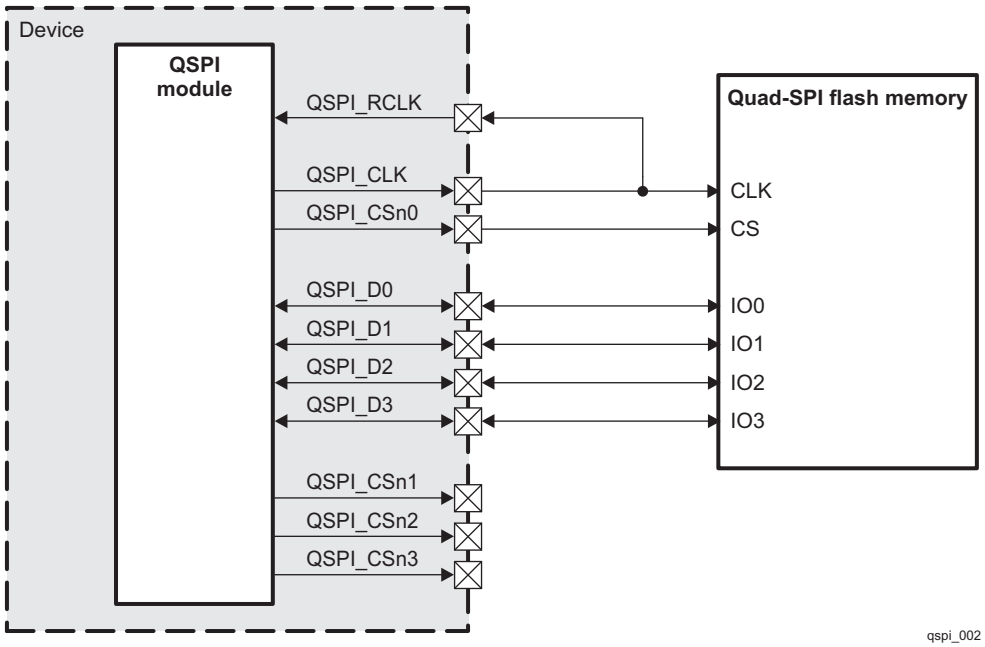


Table 11-3203 lists and describes the QSPI I/O signals.

Table 11-3203. QSPI I/O Signals

Signal Name	I/O[1]	Description
QSPI_D0	IO	QSPI data input/output 0
QSPI_D1	IO	QSPI data input/output 1
QSPI_D2	IO	QSPI data input/output 2
QSPI_D3	IO	QSPI data input/output 3
QSPI_CSn0	O	External flash device chip select 0
QSPI_CSn1	O	External flash device chip select 1
QSPI_CSn2	O	External flash device chip select 2
QSPI_CSn3	O	External flash device chip select 3
QSPI_CLK	O	QSPI clock output for the external flash device
QSPI_RCLK	I	QSPI return clock input (looped back clock of QSPI_CLK)[2]

(1) I = Input; O = Output

(2) The clock input signal is a looped back version of QSPI_CLK and facilitates easier timing closure at higher speeds. The loop back has to be at board level in order to support higher QSPI speeds.

11.15.3 QSPI Integration

This section describes the module integration in the device, including information about clocks, resets, and hardware requests.

Figure 11-1307 shows the QSPI module integration.

Figure 11-1307. QSPI Integration

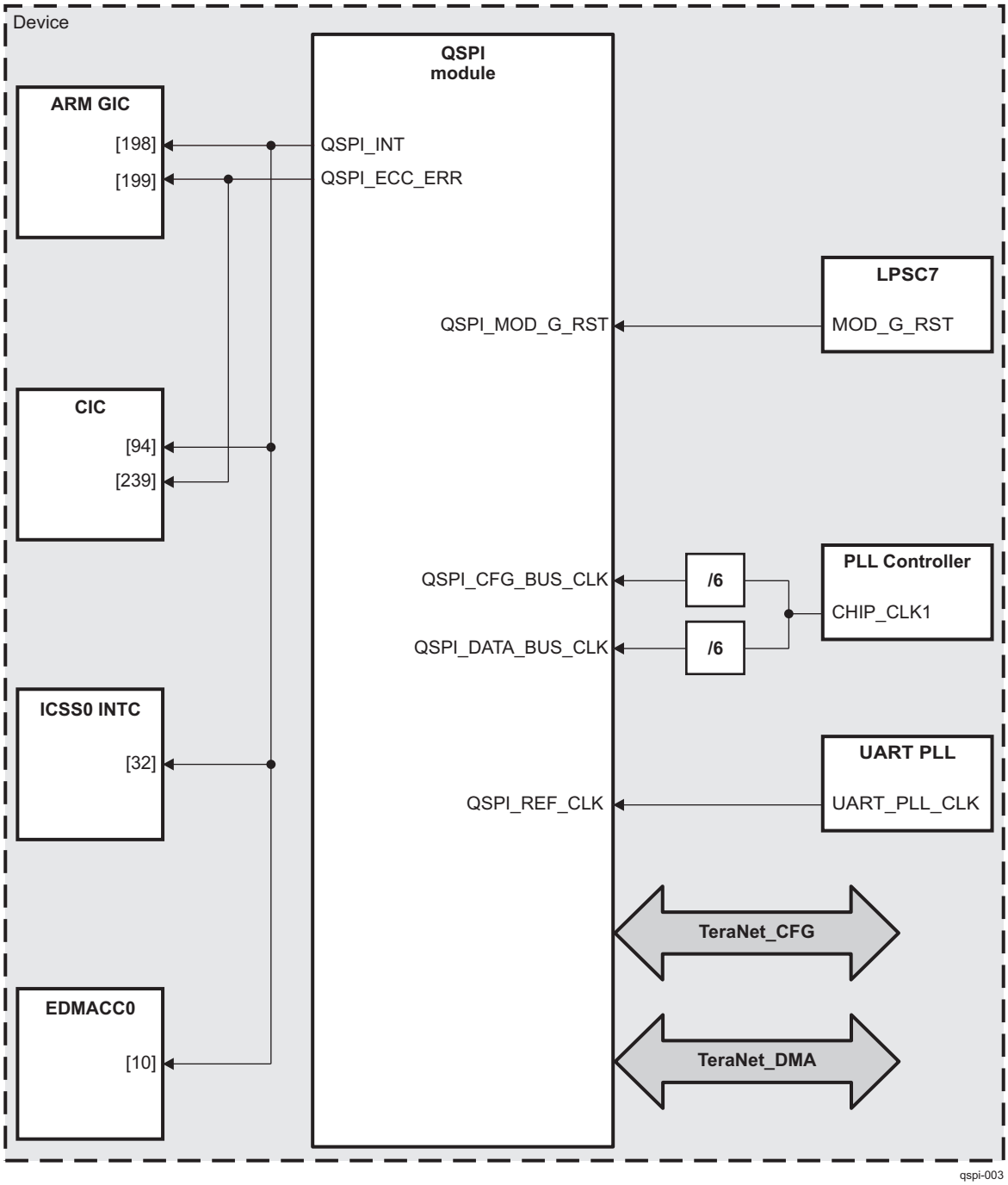


Table 11-3204 through Table 11-3206 summarize the integration of the module in the device.

Table 11-3204. QSPI Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
QSPI	PD5	LPSC7	TeraNet_DMA TeraNet_CFG

Table 11-3205. QSPI Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
QSPI	QSPI_REF_CLK	UART_PLL_CLK	UART PLL	QSPI Functional Reference Clock
	QSPI_DATA_BUS_CLK	CHIP_CLK1 / 6	PLL Controller	QSPI Data Bus Clock
	QSPI_CFG_BUS_CLK	CHIP_CLK1 / 6	PLL Controller	QSPI Config Bus Clock
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
QSPI	QSPI_MOD_G_RST	MOD_G_RST	LPSC7	Module Reset

Table 11-3206. QSPI Hardware Requests

Interrupt Requests						
Module Instance	**Event Name**	**Mapped To Input Event [Number]**				**Description**
		ARM GIC	**CIC**	**ICSS0 INTC**	**EDMACC0**	
QSPI	QSPI_INT	[198]	[94]	[32]	[10]	QSPI Interrupt Request
	QSPI_ECC_ERR	[199]	[239]	-	-	QSPI ECC Interrupt Request

11.15.3.1 QSPI Clock Domains

The QSPI module has three clock domains:

- Config bus clock domain.
- Data bus clock domain.
- SPI reference clock domain.

All of these clocks are asynchronous to each other. The first two clock domains correspond to the configuration and data busses respectively. The data bus clock (QSPI_DATA_BUS_CLK) is used to transfer data over the data bus between a master on the system interconnect and the QSPI module. The data bus clock also drives the internal QSPI SRAM. The config bus clock (QSPI_CFG_BUS_CLK) is used to access the QSPI configuration registers and for interrupt handling. The SPI reference clock (QSPI_REF_CLK) drives the SPI transmit and receive logic in the QSPI module. It is also used to generate the output SPI protocol clock (QSPI_CLK) and for oversampling of the input data (QSPI_RCLK). Using the QSPI_REF_CLK allows the QSPI module to decouple the frequency of the SPI flash device from the device system clocks, thereby providing more flexible clocking solution.

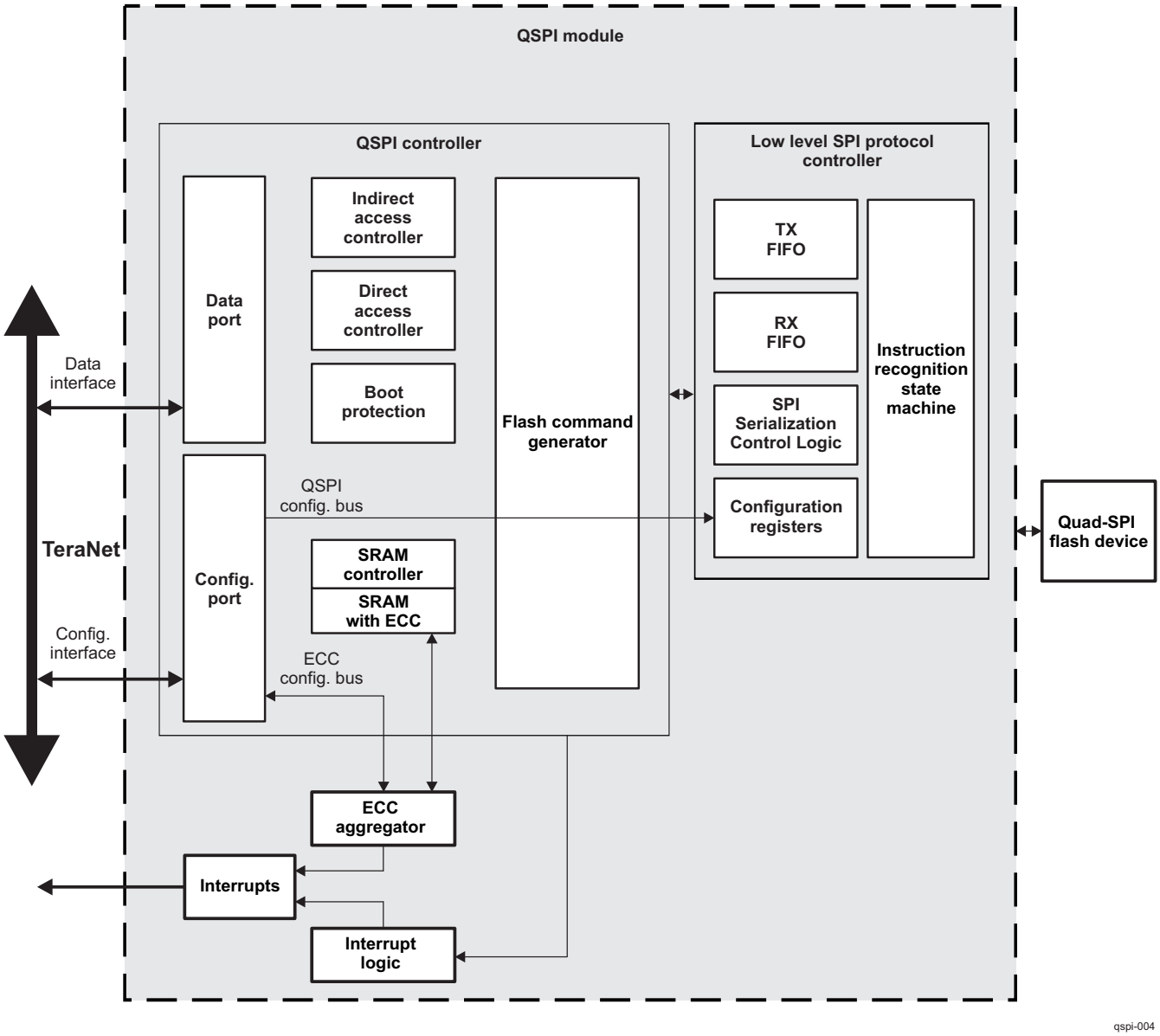
NOTE: There is no particular clock ratio requirement between QSPI_DATA_BUS_CLK and QSPI_CFG_BUS_CLK. But QSPI_REF_CLK has to be 2 times greater than QSPI_DATA_BUS_CLK and 4 times greater than QSPI_CLK.

11.15.4 QSPI Functional Description

11.15.4.1 QSPI Block Diagram

Figure 11-1308 shows the QSPI module block diagram.

Figure 11-1308. QSPI Block Diagram



The QSPI module is composed of two main blocks. The first one is the QSPI controller and the second one is the Low level SPI protocol controller.

The QSPI module has the following two slave ports:

- Data slave port intended for data transfer.
- Configuration slave port intended for accessing the programmable set of registers.

11.15.4.1.1 Data Slave Port

The data port is used for data transfer to external flash devices in direct and indirect mode of operation. It validates the incoming data accesses, responds to invalid requests, performs any required byte and half-word reordering, blocks writes violating the programmed write protection rules (only for direct access) and forwards the transfer request to either the direct access controller (DAC) or the indirect access controller (INDAC). The data interface bus is 32-bit wide. Therefore only byte, half-word and word accesses are permitted.

NOTE: Cache line wrap accesses over the data slave port should be word aligned.

Data slave port does't support cache line wrap bursts of 128 bytes.

11.15.4.1.2 Configuration Slave Port

The configuration port is used to configure the QSPI module and perform software controlled flash accesses using the QSPI_FLASH_CMD_CTRL_REG register (for more information refer to Section 11.15.4.10, *Software Triggered Instruction Generator (STIG)*). Depending on the address it routes the incoming interconnect transfer to the Low level SPI protocol controller or to the ECC aggregator. The configuration port is also used to interact with the QSPI configuration and SRAM ECC registers.

NOTE: The configuration port supports only 32-bit accesses. For single byte or half word manipulations software should perform read-modify-write operations.

11.15.4.2 QSPI Modes

The QSPI module supports four SPI modes. These modes are defined through the QSPI_CONFIG_REG[1] SEL_CLK_POL_FLD and QSPI_CONFIG_REG[2] SEL_CLK_PHASE_FLD bits. The SEL_CLK_POL_FLD bit defines the clock polarity and the SEL_CLK_PHASE_FLD bit defines the data launch and data capture relation to the QSPI_CLK edges. Table 11-3207 gives a brief description of these modes.

Table 11-3207. QSPI Modes

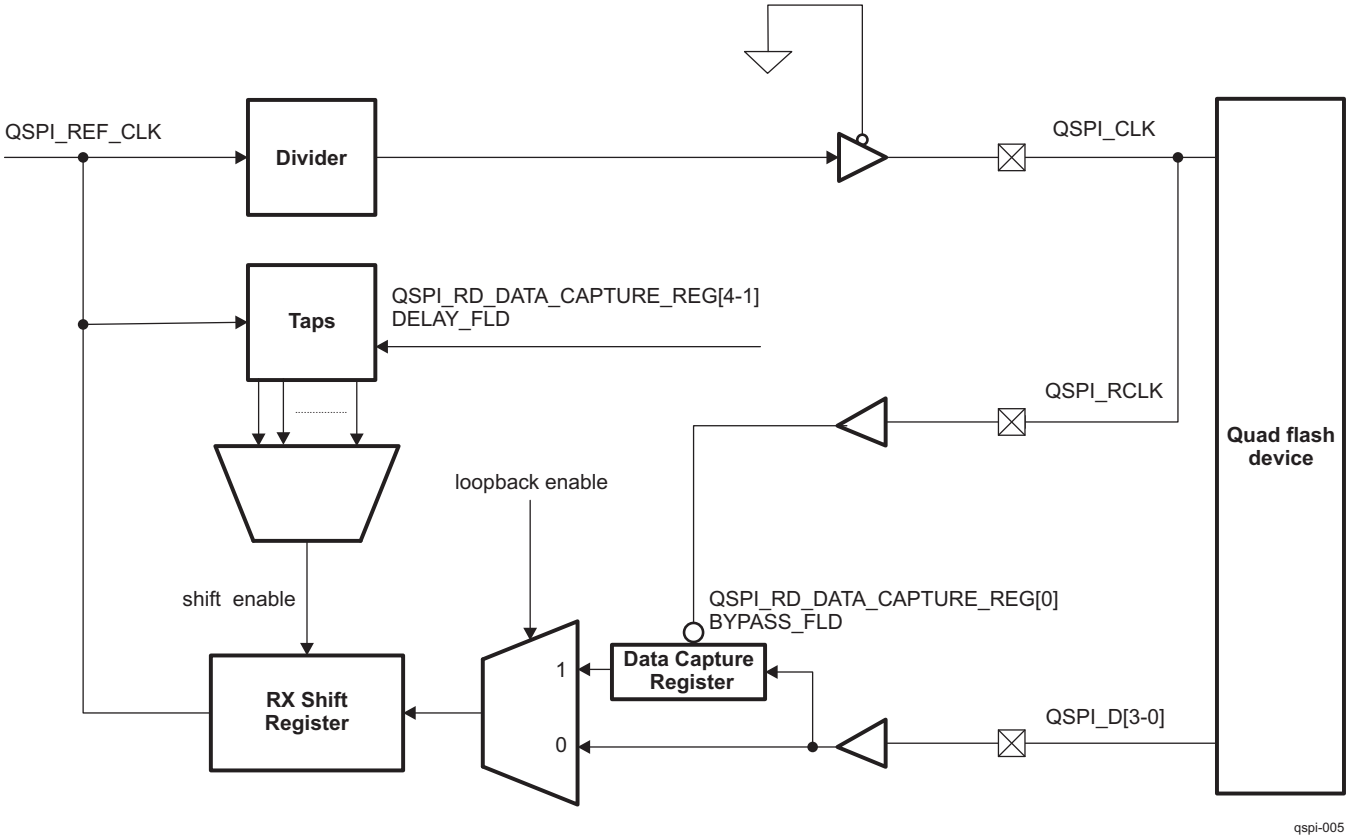
QSPI Mode	SEL_CLK_POL_FLD	SEL_CLK_PHASE_FLD	Description
0	0	0	Clock inactive state : low
			Data launch edge : clock falling edge
			Data capture edge : clock rising edge
1	0	1	Clock inactive state : low
			Data launch edge : clock rising edge
			Data capture edge : clock falling edge
2	1	0	Clock inactive state : high
			Data launch edge : clock rising edge
			Data capture edge : clock falling edge
3	1	1	Clock inactive state : high
			Data launch edge : clock falling edge
			Data capture edge : clock rising edge

The QSPI module has a loopback mode in which a clock, looped back at board level, is used for registering the input data. And the edge used is same as the launch edge (for more information see Section 11.15.4.2.1, *Read Data Capture*).

11.15.4.2.1 Read Data Capture

Figure 11-1309 shows the Read Data Capture Logic in the QSPI module.

Figure 11-1309. Read Data Capture Logic



The loopback mode is enabled by writing 0 to QSPI_RD_DATA_CAPTURE_REG[0] BYPASS_FLD bit. The taps are selected by programming QSPI_RD_DATA_CAPTURE_REG[4-1] DELAY_FLD field. The taps delay the read data capturing logic by the programmed number of QSPI_REF_CLK cycles.

There are three programmable features allowing software to tune the logic that captures the read data from the device. After POR, the adapted loopback clock circuit and the QSPI_DEV_DELAY_REG register line both wake in a disabled state. This should be valid for most applications, and can be used for device enumeration, where the device clock will be configured slowly. The QSPI_RD_DATA_CAPTURE_REG register provides the control for these features. The QSPI_RD_DATA_CAPTURE_REG[5] SAMPLE_EDGE_SEL_FLD bit selects the edge of the QSPI_REF_CLK signal, on which data outputs from flash memory are sampled. The QSPI_RD_DATA_CAPTURE_REG[4-1] DELAY_FLD field controls the additional number of read data capture cycles (this is the fast QSPI_REF_CLK signal, running at least x4 times of the device clock) that should be applied to the internal read data capture circuit. The large clock-to-out delay of the flash memory together with trace delays as well as other device delays may impose a maximum flash clock frequency which is less.

11.15.4.3 QSPI Memory Regions

A total of 256 B space is allocated to the configuration port and data port is 64 MB. Table 11-3208 shows QSPI memory map.

Table 11-3208. QSPI Memory Map

Address Range	Size	Description
2400 0000h to 27FF FFFFh	64 MB	Data register space
0294 0000h to 0294 00FFh	256 B	Configuration register space
0294 0100h to 0294 03FFh	768 B	Reserved
0294 0400h to 0294 07FFh	1 KB	SRAM ECC
0294 0800h to 0294 0FFFh	2 KB	Reserved

11.15.4.4 QSPI Interrupt Requests

The QSPI module generates two interrupts. The ECC interrupt (QSPI_ECC_ERR) is generated by the ECC aggregator. The other interrupt (QSPI_INT) is generated by the QSPI module.

Table 11-3209 lists the event flags and the corresponding mask bits of the sources which can cause interrupts.

Table 11-3209. QSPI Events

Event Flag	Event Mask	Description
QSPI_IRQ_STATUS_REG[0] MODE_M_FAIL_FLD	QSPI_IRQ_MASK_REG[0] MODE_M_FAIL_MASK_FLD	Event Flag and Event Mask for the QSPI Interrupts.
QSPI_IRQ_STATUS_REG[1] UNDERFLOW_DET_FLD	QSPI_IRQ_MASK_REG[1] UNDERFLOW_DET_MASK_FLD	
QSPI_IRQ_STATUS_REG[2] INDIRECT_OP_DONE_FLD	QSPI_IRQ_MASK_REG[2] INDIRECT_OP_DONE_MASK_FLD	
QSPI_IRQ_STATUS_REG[3] INDIRECT_READ_REJECT_FLD	QSPI_IRQ_MASK_REG[3] INDIRECT_READ_REJECT_MASK_FLD	
QSPI_IRQ_STATUS_REG[4] PROT_WR_ATTEMPT_FLD	QSPI_IRQ_MASK_REG[4] PROT_WR_ATTEMPT_MASK_FLD	
QSPI_IRQ_STATUS_REG[5] ILLEGAL_ACCESS_DET_FLD	QSPI_IRQ_MASK_REG[5] ILLEGAL_ACCESS_DET_MASK_FLD	
QSPI_IRQ_STATUS_REG[6] INDIRECT_XFER_LEVEL_BREACH_FLD	QSPI_IRQ_MASK_REG[6] INDIRECT_XFER_LEVEL_BREACH_MASK_FLD	
QSPI_IRQ_STATUS_REG[7] RECV_OVERFLOW_FLD	QSPI_IRQ_MASK_REG[7] RECV_OVERFLOW_MASK_FLD	
QSPI_IRQ_STATUS_REG[8] TX_FIFO_NOT_FULL_FLD	QSPI_IRQ_MASK_REG[8] TX_FIFO_NOT_FULL_MASK_FLD	
QSPI_IRQ_STATUS_REG[9] TX_FIFO_FULL_FLD	QSPI_IRQ_MASK_REG[9] TX_FIFO_FULL_MASK_FLD	
QSPI_IRQ_STATUS_REG[10] RX_FIFO_NOT_EMPTY_FLD	QSPI_IRQ_MASK_REG[10] RX_FIFO_NOT_EMPTY_MASK_FLD	
QSPI_IRQ_STATUS_REG[11] RX_FIFO_FULL_FLD	QSPI_IRQ_MASK_REG[11] RX_FIFO_FULL_MASK_FLD	
QSPI_IRQ_STATUS_REG[12] INDRD_SRAM_FULL_FLD	QSPI_IRQ_MASK_REG[12] INDRD_SRAM_FULL_MASK_FLD	

Table 11-3209. QSPI Events (continued)

Event Flag	Event Mask	Description
QSPI_ECC_INT_STATUS[31-0] BITMASK	QSPI_ECC_INT_ENABLE[31-0] BITMASK QSPI_ECC_INT_CLEAR[31-0] BITMASK	Event Flag and Event Mask for the ECC Interrupts.

11.15.4.5 Data Interface Address Remapping

The incoming data interface address, by default, maps directly to the address sent serially to the FLASH device. If the FLASH device has a 24-bit address, then the 24 LSB's of the data address will be forwarded. A remap feature is available to remap all incoming AHB addresses to ADDRESS + N, where N is the value stored in the QSPI_REMAP_ADDR_REG[31-0] VALUE_FLD bit field. It is enabled via the QSPI_CONFIG_REG[16] ENB_AHB_ADDR_REMAP_FLD bit. This feature could be used when software needs to move boot code to another FLASH region.

11.15.4.6 Write Protection

In order to protect the FLASH device, a software controlled write protection feature is supported. Any data write detected (by using DAC), pointing to an area of the FLASH that is protected, will be not permitted.

A programmable region of the FLASH device, defined as a number of FLASH 'blocks' starting from a particular block number can be protected. Three programmable registers are provided. The first QSPI_LOWER_WR_PROT_REG register defines the FLASH block that is located at the bottom of the region to be protected. The second QSPI_UPPER_WR_PROT_REG register defines the FLASH block that is located at the top of the region to be protected. The third QSPI_WR_PROT_CTRL_REG register is a control register consisting of 2 bits. The QSPI_WR_PROT_CTRL_REG[0] INV_FLD bit allows software to invert the region that is being protected, causing the programmed region to become the only areas of FLASH memory that is not protected from writes. The QSPI_WR_PROT_CTRL_REG[1] ENB_FLD bit is the write protection enable bit. When this bit is set to 0, the FLASH device is unprotected.

For implementation, the data interface must map the incoming address into its associated FLASH block. A block can be between 1 and 65 KB, programmed via the QSPI_DEV_SIZE_CONFIG_REG register.

NOTE: The write protection feature is supported for DAC mode only. For INDAC mode this feature is not supported.

11.15.4.7 Access Forwarding

For legal accesses, the data interface will forward all accesses to one of two access controllers - the direct access and the indirect access controllers. Assuming DAC has been enabled via the QSPI_CONFIG_REG[7] ENB_DIR_ACC_CTRL_FLD bit, then by default all accesses will be forwarded to this controller. Before any accesses can be forwarded to INDAC, it must first be configured by software. This process is fully explained in Section 11.15.4.9, *Indirect Controller (INDAC)*. If DAC is disabled, any incoming access that can't be forwarded to INDAC will be completed immediately with an error. If DAC is enabled, the same access will be forwarded and serviced by DAC.

11.15.4.8 Direct Access Controller (DAC)

Direct access refers to the operation where data interface accesses directly trigger a read or write to FLASH memory. It is memory mapped and can be used to both access and directly execute code from external FLASH memory. Any incoming access that is not recognized as being within the programmable indirect trigger region is assumed to be a direct access and will be serviced by the DAC. Note that accesses that use DAC do not use the embedded SRAM. The data transfer stops when read or write burst is carried out. The amount of wait states applied will be dependent on the latency through the controller. Latency is kept to a minimum when the use of XIP read instructions are enabled (see QSPI_CONFIG_REG[18] ENTER_XIP_MODE_IMM_FLD and QSPI_CONFIG_REG[17] ENTER_XIP_MODE_FLD bits).

11.15.4.9 Indirect Controller (INDAC)

11.15.4.9.1 Indirect Read Controller

The aim of the indirect mode of operation is to read significant numbers of bytes from FLASH memory without requiring a data interface access to trigger it. Instead indirect operations are controlled and triggered by software via specific control/configuration Indirect Read Transfer registers (for more information see the following registers: QSPI_INDIRECT_READ_XFER_CTRL_REG, QSPI_INDIRECT_READ_XFER_WATERMARK_REG, QSPI_INDIRECT_READ_XFER_START_REG, and QSPI_INDIRECT_READ_XFER_NUM_BYTES_REG). This block will communicate with an embedded Low level SPI protocol state machine module to perform an efficient and optimized FLASH read burst, placing the read data into the local SRAM module ready for fast and low latency delivery to any external TeraNet master.

By default, the indirect read controller is disabled. Before enabling it, software must configure how much data is required and the start address. The start address and total number of bytes to be fetched is defined in QSPI_INDIRECT_READ_XFER_START_REG and QSPI_INDIRECT_READ_XFER_NUM_BYTES_REG registers respectively. Up to two indirect operations can be programmed at any one time. The second operation can be triggered while the first is in progress. Supporting two indirect operations allows a short turnaround time between the completion of one indirect operation and the start of the second. For more information refer to Section 11.15.4.9.3, *Indirect Access Queuing*.

The total number of bytes to read in an indirect operation is not limited by the size of the SRAM. The size of SRAM will only limit the size of requests. In the case of SRAM overrun, the controller will back pressure FLASH reads until space becomes available in the SRAM. Back pressuring the reads on the SPI interface is handled by completing any current read burst, waiting until space in the SRAM becomes available and then issuing a new read burst at the address where the previous terminated burst ended.

An external master will be able to fetch the data that the controller has read from external FLASH memory by issuing data interface reads to the QSPI module. The address of the incoming read access must be in the range of Indirect trigger address programmed via the QSPI_IND_AHB_ADDR_TRIGGER_REG register to Indirect trigger address + 15. This allows a 16-beat burst to be applied starting from the Indirect trigger address. The smaller bursts are possible to handle effectively as well with this approach. Furthermore it is not strict requirement to push consecutive address sequence. Actual address just has to be in the Indirect Range to grant SRAM as source. Each read will cause the internal SRAM to be popped, thereby decoupling the incoming read access address from the FLASH address – that is not direct mapped. Therefore the Indirect trigger address does not have any relationship with the FLASH address. It is just to indicate that data should take the SRAM as source instead of the FLASH Memory array after triggering of any valid Indirect Read. The FLASH address for Indirect Read is taken from the QSPI_INDIRECT_READ_XFER_START_REG register. Assuming the requested data is present in the SRAM at the point the data interface access is received by the QSPI module, then the data will be fetched from the SRAM and the response to the read burst will be achieved with minimum latency. Once the data has been read from the SRAM, the QSPI module will free up the associated resource in it.

If a read access is received whose address is not within the range described above then that access will not be completed using the indirect controller. It will instead be serviced by the direct access controller.

If a read access is received whose address is within the range described above but the requested data is not immediately present in the SRAM then wait states will be applied until the data has been read from FLASH and pushed to the SRAM.

If a read burst is received whose access elements traverse the Indirect trigger range, then the accesses within the Indirect trigger range will be processed by the indirect controller and the rest will be taken by the direct access controller. This is likely to be a software configuration error.

The external master is only permitted to issue 32-bit data interface reads until the last word of an indirect transfer. This helps keep the SRAM control logic less complex. On the final read, the external master may issue a 16-bit (HalfWord) or byte access to complete the transfer. It is also permitted for the external master to always issue a 32-bit Word read on the last indirect access. The controller will pad the upper bits of the response with zero. The current expectation is that the SRAM will be kept fairly full while the read operation is carried out. The fill level of the SRAM is directly readable by software reading the QSPI_SRAM_FILL_REG register.

An indirect operation may be cancelled at any time by setting 1 to QSPI_INDIRECT_READ_XFER_CTRL_REG[1] CANCEL_FLD bit.

Any bus master should be allowed to initiate an indirect access. The QSPI module provide software access mechanism to the SRAM fill-level directly via the configuration registers and then decide for itself when the data should be fetched from the local SRAM. The fill level watermark register (see QSPI_INDIRECT_READ_XFER_WATERMARK_REG register) is provided. When the SRAM fill level passes this watermark, an interrupt is generated. If the watermark value is > 0, the watermark interrupt is also generated when the final byte of data has been read by the QSPI module and placed in the SRAM, even if the actual SRAM fill level has not risen above the watermark. This last feature is useful to avoid software tracking how much data has been read and resetting the watermark value for the last few bytes of an indirect read transfer.

Two further interrupt sources are provided to help understand the status of an indirect operation. Firstly, an interrupt is generated when an indirect operation has completed. Secondly, an interrupt is generated if an indirect read operation was requested but could not be accepted due to the fact 2 indirect operations have already been buffered by the QSPI module.

Setting the QSPI_INDIRECT_READ_XFER_CTRL_REG[0] START_FLD bit starts an indirect read operation. QSPI_INDIRECT_READ_XFER_CTRL_REG[2] RD_STATUS_FLD bit is available to check the status.

11.15.4.9.1.1 Indirect Read Transfer Process

The following sequence can be followed:

- 1. Set-up QSPI_CONFIG_REG register.
- 2. Set-up the indirect transfer's FLASH start address in the QSPI_INDIRECT_READ_XFER_START_REG register.
- 3. Set-up the number of bytes to be transferred in the QSPI_INDIRECT_READ_XFER_NUM_BYTES_REG register.
- 4. Set-up the indirect transfer's trigger address in the QSPI_IND_AHB_ADDR_TRIGGER_REG register.
- 5. If the watermark interrupt feature is to be used, set the SRAM fill level watermark register (QSPI_INDIRECT_READ_XFER_WATERMARK_REG register) which will cause an interrupt to be generated when the fill level increases beyond the watermark level. Setting the watermark can be useful indication to software when to read the next part of the indirect read transfer. Note that if the watermark is set to a value other than zero, the watermark interrupt will always trigger once the final byte of indirect transfer has been fetched and placed in the embedded SRAM, even if the watermark value is higher than the actual completed fill level.
- 6. Trigger Indirect Read access by setting the QSPI_INDIRECT_READ_XFER_CTRL_REG[0] START_FLD bit to 1.
- 7. If the watermark interrupt feature is to be used, wait for watermark interrupt. Else poll the SRAM fill level via the QSPI_SRAM_FILL_REG register to decide when sufficient data is in the SRAM to trigger data fetches.
- 8. Read the expected amount of data from SRAM. If there is still more data to fetch in order to complete the indirect read transfer, then loop back to step 7. Otherwise continue to step 9.
- 9. The completion status of the indirect read operation can be polled via the QSPI_INDIRECT_READ_XFER_CTRL_REG[5] IND_OPS_DONE_STATUS_FLD bit.
- 10. An Indirect Complete interrupt will be generated when the Indirect read operation has completed.

11.15.4.9.2 Indirect Write Controller

The aim of the indirect mode of operation is to perform bulk transfer of data from the processor into a FLASH memory in the most efficient manner. The fewest possible write cycles inside the FLASH device will be carried out for the indirect transfer, thus maximizing the life of the device. Indirect write operation can be thought of from a software perspective as the inverse of the indirect read. It is controlled and triggered by software via specific control/configuration Indirect Write Transfer registers (for more

information see the following registers: QSPI_INDIRECT_WRITE_XFER_CTRL_REG, QSPI_INDIRECT_WRITE_XFER_WATERMARK_REG, QSPI_INDIRECT_WRITE_XFER_START_REG, and QSPI_INDIRECT_WRITE_XFER_NUM_BYTES_REG). This block will await delivery of the write data via the external TeraNet master, placing it in the local SRAM before communicating with the existing legacy SPI core to perform an efficient and optimized FLASH write burst.

By default, the indirect write controller is disabled. Before enabling it, the software must configure how much data is required and the start address. The start address and total number of bytes to be written is defined in QSPI_INDIRECT_WRITE_XFER_START_REG and QSPI_INDIRECT_WRITE_XFER_NUM_BYTES_REG registers respectively. Up to two indirect operations can be programmed at any one time. The second operation can be triggered while the first is in progress. Supporting two indirect operations allows a short turnaround time between the completion of one indirect operation and the start of the second. The Indirect write queuing is very similar to indirect read queuing. For more information refer to Section 11.15.4.9.3, *Indirect Access Queuing*.

The total number of bytes to write in an indirect operation is not limited by the size of the SRAM. The size of SRAM will only limit the amount of data that can be accepted from the external TeraNet master. In the case of SRAM overrun, the controller will back pressure the data interface with wait states. Note the fill level of the SRAM is readable via programmable QSPI_SRAM_FILL_REG register and this can be used to avoid this situation.

An external master will provide the write data and will transfer this to the QSPI module by issuing data interface writes. The address of the incoming write access must be in the range of Indirect trigger address programmed via the QSPI_IND_AHB_ADDR_TRIGGER_REG register to Indirect trigger address + 15. This allows a 16-beat burst to be applied starting from the Indirect trigger address. The smaller bursts are possible to handle effectively as well with this approach. Furthermore it is not strict requirement to push consecutive address sequence. Actual address just has to be in the Indirect Range to grant SRAM as source. Each write will cause the internal SRAM to be popped, thereby decoupling the incoming write access address from the FLASH address – that is not direct mapped. Therefore Indirect trigger address does not have any relationship with FLASH address. It is just to indicate that data should take SRAM as source instead of FLASH Memory array after triggering of any valid Indirect Write. The FLASH address for Indirect Write is taken from the QSPI_INDIRECT_WRITE_XFER_START_REG register. Assuming the SRAM is not full at the point the data interface access is received by the QSPI module, then the data will be pushed to the SRAM with minimum latency.

If a write access is received whose address is not within the range described above then that access will not be completed using the indirect controller. It will instead be serviced by the direct access controller.

If a write access is received whose address is within the range described above but the SRAM is full then wait states will be applied until some or all of the data has been pushed from the SRAM to the FLASH.

If a write burst is received whose access elements traverse the Indirect trigger range, then the accesses within the Indirect trigger range will be processed by the indirect controller, and the rest will be taken by the direct access controller. This is likely to be a software configuration error.

The external master is only permitted to issue 32-bit data interface writes until the last word of an indirect transfer. This helps keep the SRAM control logic less complex. On the final write, the external master may issue a 32-bit word, 16-bit (half-word) or a byte access to complete the transfer. If the number of bytes to write is less than 4 on the last transfer, the master is still permitted to issue a 32-bit transfer. In these cases, the extra bytes are discarded by the controller.

When the SRAM holds a number of bytes equal to or greater than the size of a FLASH page (which itself is programmed into the QSPI module, with a default of 256 bytes) or when the SRAM holds all remaining bytes of the currently executing indirect transfer, the QSPI module will initiate a write burst to the flash command generator.

An indirect operation may be cancelled at any time by setting 1 to the QSPI_INDIRECT_WRITE_XFER_CTRL_REG[1] CANCEL_FLD bit.

Any bus master should be allowed to initiate an indirect access. The QSPI module provide software access mechanism to the SRAM fill-level directly via the configuration registers and then decide for itself when the data should be written to the local SRAM. The fill level watermark register (see QSPI_INDIRECT_WRITE_XFER_WATERMARK_REG register) is provided. When the SRAM fill level falls below this watermark, an interrupt is generated.

Two further interrupt sources are provided to help understand the status of an indirect operation. Firstly, an interrupt is generated when an indirect operation has completed. Secondly, an interrupt is generated if an indirect write operation was requested but could not be accepted due to the fact 2 indirect operations have already been buffered by the QSPI module.

Setting the QSPI_INDIRECT_WRITE_XFER_CTRL_REG[0] START_FLD bit starts an indirect write operation. The QSPI_INDIRECT_WRITE_XFER_CTRL_REG[2] WR_STATUS_FLD bit is available to check the status.

11.15.4.9.2.1 Indirect Write Transfer Process

The following sequence can be followed:

- 1. Set-up the indirect transfer's FLASH start address in the QSPI_INDIRECT_WRITE_XFER_START_REG register.
- 2. Set-up the number of bytes to be transferred in the QSPI_INDIRECT_WRITE_XFER_NUM_BYTES_REG register.
- 3. Set-up the indirect transfer's trigger address in the QSPI_IND_AHB_ADDR_TRIGGER_REG register.
- 4. It is functionally valid for software to simply write all the data to the SRAM in one block transfer. However, if the total number of bytes to write is greater than the size of the partitioned SRAM, then it is quite likely the SRAM will become full causing the QSPI module to back-pressure the system data bus for a considerable time. This time is based on the FLASH data-rate and the page-write time of the device. To avoid sending all the write data in one block transfer, the software can make use of the watermark interrupt to identify a convenient time to send data a page at a time to the SRAM module. Alternatively, the software can poll the SRAM fill level register directly to identify how empty the SRAM is at any one time in order to make a choise as to when is the most practical time to send the next part of the transfer.
- 5. If the watermark interrupt feature is to be used, set the SRAM fill level watermark register (see QSPI_INDIRECT_WRITE_XFER_WATERMARK_REG and QSPI_SRAM_FILL_REG registers) which will cause an interrupt to be generated when the fill level falls below the watermark. The watermark should be set to a number between zero and a page size. If the page size is 256 bytes, then setting the watermark to a value between 10 and 250 is reasonable and will cause the interrupt to trigger when the fill level drops below the programmed number. Setting the watermark can be useful to provide an indication to software when to write the next page of data to the SRAM.
- 6. Trigger Indirect Write access by setting the QSPI_INDIRECT_WRITE_XFER_CTRL_REG[0] START_FLD bit to 1.
- 7. If the remaining number of bytes still to be transferred into the QSPI SRAM for the current indirect transfer is greater than a FLASH page, then write 1 FLASH page worth of data to the SRAM. Otherwise send the remaining data from the indirect transfer to the SRAM.
- 8. If all the data in the indirect transfer has now been sent to the SRAM, then goto step 10 and await indirect complete status. Otherwise if there is more data still to be transferred then either
  - If the watermark interrupt feature is being used, then wait for watermark interrupt.
  - Alternatively the SRAM fill level can be checked to identify a convenient time to send more data.
- 9. Loop back to step 7.
- 10. Optional: The completion status of the Indirect write operation can be polled via the QSPI_INDIRECT_WRITE_XFER_CTRL_REG[2] WR_STATUS_FLD bit.
- 11. An Indirect Complete interrupt will be generated when the Indirect write operation has completed.

11.15.4.9.3 Indirect Access Queuing

Software is permitted to queue up to two indirect transfers for both the indirect write controller and the indirect read controller. Supporting two indirect operations allows a short turnaround time between the completion of one indirect operation and the start of the second. Any attempt to queue more than two operations will cause an interrupt to be generated. To take advantage of this feature, software should attempt to keep both indirect programming slots full at all times.

From the software perspective, indirect access queuing is achieved by triggering bit 0 of the indirect transfer control register (QSPI_INDIRECT_READ_XFER_CTRL_REG[0] START_FLD bit or QSPI_INDIRECT_WRITE_XFER_CTRL_REG[0] START_FLD bit) twice in short succession. The indirect number of bytes register (QSPI_INDIRECT_READ_XFER_NUM_BYTES_REG or QSPI_INDIRECT_WRITE_XFER_NUM_BYTES_REG register) and the indirect FLASH start address register (QSPI_INDIRECT_READ_XFER_START_REG or QSPI_INDIRECT_WRITE_XFER_START_REG register) must be set-up with the relevant transfer data before START_FLD bit can be triggered for each transfer. Since these registers will change regularly, the hardware must keep sampled versions of these registers for the duration of the indirect transfer.

The internal register block will only issue an indirect start trigger to the key underlying datapath blocks one at a time. There are 2 independent datapath blocks in the indirect access controller that will receive and independently sample this information. The first is the datapath block on the data bus side of the SRAM. For indirect reads, this is a read interface, for indirect writes, it is a write interface. The second is the datapath block on the FLASH side of the SRAM. For indirect reads, this is a write interface, for indirect writes, it is a read interface. Both blocks will process the indirect transfers at different times. For example, for an indirect read operation, the datapath block on the FLASH side of the SRAM will be able to start processing the second queued transfer as soon as the last byte of the first transfer has been written to the SRAM. Before commencing the second transfer, this block must resample the QSPI_INDIRECT_READ_XFER_NUM_BYTES_REG and QSPI_INDIRECT_READ_XFER_START_REG registers. Similarly, the datapath block on the bus side will resample the same registers locally when it has forwarded all the FLASH data associated with the first indirect transfer from the SRAM onto the data bus.

11.15.4.9.4 Consecutive Writes and Reads Using Indirect Transfers

It is permitted for software to trigger an indirect read operation while an indirect write operation is in progress. Similarly it is permitted to trigger an indirect write while an indirect read operation is in progress. Indirect write operations will take overall precedence.

11.15.4.9.5 Accessing the SRAM

The SRAM is separated in two segments. The lower segment is reserved for indirect read use. The upper segment is for indirect write use only. The size of each segment is programmable via the QSPI_SRAM_PARTITION_CFG_REG register. This feature allows to allocate how many bits of the SRAM address bus are allocated to indirect read. By default, this is set so that exactly half of the SRAM is portioned for use by the indirect read controller. To ensure the read data bus is not directly fed by the SRAM read data through combinatorial logic, an extra bank of holding registers is included in the indirect read data path. These registers act as an extra location to be added to the allocated number of SRAM locations for indirect read.

To illustrate how the SRAM (and the extra bank of holding registers) can be allocated between indirect read and write, the following example is provided. The depth of the SRAM in this example is configured to be 8 bits. This is equal to 256 locations.

- If the QSPI_SRAM_PARTITION_CFG_REG[7-0] ADDR_FLD field is set to 00h, then 256 locations are allocated to indirect writes and 1 location to indirect reads.
- If the QSPI_SRAM_PARTITION_CFG_REG[7-0] ADDR_FLD field is set to 01h, then 255 locations are allocated to indirect writes and 2 locations to indirect reads.
- If the QSPI_SRAM_PARTITION_CFG_REG[7-0] ADDR_FLD field is set to 02h, then 254 locations are allocated to indirect writes and 3 locations to indirect reads.
- And so on until.
- If the QSPI_SRAM_PARTITION_CFG_REG[7-0] ADDR_FLD field is set to FDh, then 3 locations are allocated to indirect writes and 254 locations to indirect reads.
- If the QSPI_SRAM_PARTITION_CFG_REG[7-0] ADDR_FLD field is set to FEh, then 2 locations are allocated to indirect writes and 255 locations to indirect reads.
- If the QSPI_SRAM_PARTITION_CFG_REG[7-0] ADDR_FLD field is set to FFh, then 1 location is allocated to indirect writes and 256 locations to indirect reads.

NOTE: A value of FFh or 00h in the QSPI_SRAM_PARTITION_CFG_REG should be avoided by software, as only the bottom 8 bits of the SRAM fill level are accessible through software (up to 255 limit) via the QSPI_SRAM_FILL_REG register. If the fill level reaches 256 on either the indirect read or write side, it will appear when reading the Fill Level to be 0.

There are four SRAM sources that are arbitrated and muxed onto the single SRAM port. Up to three sources can access this port at any one time. The sources are described as follows:

- Indirect Write, Write source. This is located on the data bus side of the SRAM.
- Indirect Write, Read source. This is located on the FLASH side of the SRAM.
- Indirect Read, Write source. This is located on the FLASH side of the SRAM.
- Indirect Read, Read source. This is located on the data bus side of the SRAM.

A fixed priority arbitration scheme is implemented. Table 11-3210 shows priority allocated to these sources.

Table 11-3210. SRAM Access Priority

SRAM Access Priority		
Indirect Write	Write to SRAM (From System Data Bus)	3rd (exclusive with Data Bus Read Request)
	Read from SRAM (From QSPI Module)	2nd
Indirect Read	Write to SRAM (From QSPI Module)	1st
	Read from SRAM (From System Data Bus)	3rd (exclusive with Data Bus Write Request)

NOTE: With the exception of the write port during an Indirect Read operation (on the FLASH side of the SRAM), the logic driving all four sources must not assume single cycle completion. Writes to the SRAM during an indirect read must be allowed to complete immediately to avoid data loss. Therefore this port is given maximum priority.

11.15.4.10 Software-Triggered Instruction Generator (STIG)

The DAC and INDAC are used to transfer data. In order to access the volatile and non-volatile configuration registers, the legacy SPI Status register, other status/protection registers as well as to perform ERASE functions, a separate software controller is required. The software triggered instruction generator (STIG) is controlled using the QSPI_FLASH_CMD_CTRL_REG register by setting up the command to issue to the FLASH device. This is a generic controller and can be used to perform any instruction that the FLASH device supports from the extended SPI protocol. Configuring of instructions which are not compliant with the specification of the FLASH Devices could cause unpredicted behavior of the controller. The QSPI_FLASH_CMD_CTRL_REG[0] CMD_EXEC_FLD bit is used to trigger the command. The QSPI_FLASH_CMD_CTRL_REG[1] CMD_EXEC_STATUS_FLD bit is used by software to poll the status of the command execution. For reads, when the command has been serviced (QSPI_FLASH_CMD_CTRL_REG[1] CMD_EXEC_STATUS_FLD bit toggles from '1' to '0'), up to 8 bytes of read data will be placed in the QSPI_FLASH_RD_DATA_LOWER_REG and QSPI_FLASH_RD_DATA_UPPER_REG registers. For writes, the write data should be placed in the QSPI_FLASH_WR_DATA_LOWER_REG and QSPI_FLASH_WR_DATA_UPPER_REG registers.

11.15.4.10.1 Servicing a STIG Request

A STIG request will cause the QSPI Flash controller to interrogate the QSPI_FLASH_CMD_CTRL_REG register to determine what and how many bytes it should send to the FLASH device. The QSPI_FLASH_CMD_CTRL_REG[31-24] CMD_OPCODE_FLD field of this register indicate the instruction to be sent and is always pushed first. If there is an address to send, then the address (the size of which is also programmed in the same register) is sent next. The address itself is stored in the QSPI_FLASH_CMD_ADDR_REG register. If there are any dummy bytes to send (the size of which is also programmed in QSPI_FLASH_CMD_CTRL_REG register) then those are sent next. If there is data to write or read (the size of which is also programmed in QSPI_FLASH_CMD_CTRL_REG register) then for the case of writes, up to 8 bytes can be sent (as stored in the Flash Command Write Data registers,

QSPI_FLASH_WR_DATA_LOWER_REG and QSPI_FLASH_WR_DATA_UPPER_REG registers) next. In the read case, when the read data has been collected from the FLASH device, the QSPI Flash Controller stores that in the Flash Command Read Data Registers (QSPI_FLASH_RD_DATA_LOWER_REG and QSPI_FLASH_RD_DATA_UPPER_REG registers). When the QSPI Flash controller starts to service a STIG request, it sets the QSPI_FLASH_CMD_CTRL_REG[1] CMD_EXEC_STATUS_FLD bit to indicate a command execution is in progress. When the QSPI Flash controller is in the auto-polling state, servicing a STIG request is slightly different. Most of devices are largely inaccessible after a program operation until the device has completed that write. Some group of them has a possibility to suspend programming page. It can be controlled by the QSPI_POLLING_FLASH_STATUS_REG[8] DEVICE_STATUS_VALID_FLD bit, which indicate active auto-polling phase. After requesting a STIG, the QSPI Flash Controller immediately issues appropriate OPCODE to Memory. During servicing a STIG (in auto-polling phase) the status bit of command execution remains steady and other parts of transfer such as ADDRESS or DUMMY BITS, and so forth, are disabled (to issued Program Suspend Command is needed OPCODE only). There is a programmable option to add delay between every repetitive poll operation (delay is defined by MSB of QSPI_WRITE_COMPLETION_CTRL_REG register). This feature is implemented to free up SPI bandwidth if needed.

NOTE: The QSPI data is sent LSB first, while address is sent MSB first.

NOTE: The STIG complete status bit gets cleared before the actual flash access completes. Software should wait for about 700 ns if there is any dependency on actual access completion.

11.15.4.11 Arbitration Between Direct / Indirect Access Controller and STIG

When multiple controllers are active simultaneously, a simple fixed-priority arbitration scheme is used to arbitrate between each interface and access the external FLASH. The fixed priority is defined as follows, highest priority first.

- The Indirect Access Write
- The Direct Access Write
- The STIG
- The Direct Access Read
- The Indirect Access Read

11.15.4.12 SPI Command Translation

Requests issued by the direct access controller, the indirect access controller or the STIG will be translated into a sequence of byte transfers to send downstream (before serialization to the FLASH device). These sequences depend on the requested transfer but an example of a typical 1-byte non sequential READ is shown below:

INSTRUCTION OPCODE -> ADDRESS -> Mode Byte -> Dummy Bytes -> 1 byte of don't care

For sequential accesses, an extra byte of data per read is pushed to the FLASH device on the back of the above sequence assuming it can be done so with no gap between each transferred byte.

The actual sequence sent to the FLASH device depends on the requested transfers, whether the transfer is non-sequential or sequential, whether the device has been configured in XIP mode and the state of the main Device Instruction Type programmable registers (QSPI_DEV_INSTR_RD_CONFIG_REG and QSPI_DEV_INSTR_WR_CONFIG_REG registers).

For writes, the Write Enable Latch (or WEL) within the FLASH device itself must be high before a write sequence can be issued. The QSPI Flash Controller will automatically issue the write enable latch command before triggering a write command via the direct or indirect access controllers (DAC/INDAC) – that is the user does not need to perform this operation. For increasing flexibility and performance user can turn off this feature by set-up bit 8th into the QSPI_DEV_INSTR_WR_CONFIG_REG register. The op-code for WREN is typically 06h and is common between devices.

When write requests from the direct or indirect access controllers are no longer being received and all outstanding requests have been sent, the FLASH device will automatically start the page program write cycle. Any incoming request at this time will be held in wait states until the cycle has completed. The QSPI Flash Controller will automatically poll the FLASH device legacy SPI status register to identify when the write cycle has completed. This is achieved by sending the RDSR opcode to the FLASH device and waiting until the device itself has indicated the write cycle has completed (until the Write in Progress bit [0] has cleared to zero and the write enable latch bit has also cleared to zero). The WREN and the RDSR device instructions are the only ones that are sent by the controller under the hood. For any other specific instruction that the user determines should be sent to the device (for example if the device needs to be unprotected before a write command is issued), these should be handled separately by issuing FLASH commands via the STIG.

11.15.4.13 Selecting the Flash Instruction Type

In order to send the correct READ and WRITE opcodes, software should initialize the QSPI_DEV_INSTR_RD_CONFIG_REG register and the QSPI_DEV_INSTR_WR_CONFIG_REG register. These registers include fields to set-up the required instruction op-codes that is intended to be used to access the FLASH (default is basic READ and basic page program) as well as the instruction type and whether the instruction uses single, dual or quad pins for address and data transfer. Providing this level of control to the user provides a future proofed generic solution. To ensure the controller can operate from a reset state, the registers will be reset to an op-code compatible with SIO devices.

Despite being applicable for both READs and WRITEs, the QSPI_DEV_INSTR_RD_CONFIG_REG[9-8] INSTR_TYPE_FLD field only appears once – it is not included in the QSPI_DEV_INSTR_WR_CONFIG_REG register. If software sets this to anything other than '0', then the address xfer type and the data xfer type bits of both QSPI_DEV_INSTR_RD_CONFIG_REG and QSPI_DEV_INSTR_WR_CONFIG_REG registers become don't care. It is made available to allow software to support the less common FLASH instructions where the opcode, address and data are sent on 2 or 4 lanes (the opcode from most instructions are sent serially to the FLASH device, even for dual/quad instructions). Also note that for devices that support instructions that can send the OPCODE over 2/4 lanes, the name for these instructions are not consistently named in the FLASH datasheets.

One of the devices that support these instructions is the Numonyx (Micron) N25Q128. The extra READ instructions are called DCFR and QCFR. The WRITE instructions are DCPP and QCPP. Table 11-3211 shows how software should configure the QSPI module for each specific READ and WRITE instruction supported by the N25Q128 device.

Table 11-3211. READ and WRITE Instruction Configuration

READS						
OPCODE	OPCODE sent over how many lanes?	ADDRESS/ DUMMY/ MODE sent over how many lanes?	DATA bytes sent over how many lanes?	QSPI Instruction TYPE (Device Read Instruction Configuration Register)	QSPI Address Xfer type (Device Read Instruction Configuration Register)	QSPI Data Xfer type (Device Read Instruction Configuration Register)
READ	1	1	1	0	0	0
FAST_READ	1	1	1	0	0	0
DOFR (Dual O/p Fast Read)	1	1	2	0	0	1
DIOFR (Dual I/O Fast Read)	1	2	2	0	1	1
QOFR (Quad O/p Fast Read)	1	1	4	0	0	2
QIOFR (Quad I/O Fast Read)	1	4	4	0	2	2

Table 11-3211. READ and WRITE Instruction Configuration (continued)

DCFR (Dual Command Fast Read)	2	2	2	1	Don't care	Don't care
QCFR (Quad Command Fast Read)	4	4	4	2	Don't care	Don't care
WRITE						
OPCODE	OPCODE sent over how many lanes?	ADDRESS/ DUMMY/ MODE sent over how many lanes?	DATA sent over how many lanes?	QSPI Instruction TYPE (Device Read Instruction Configuration Register)	QSPI Address Xfer type (Device Write Instruction Configuration Register)	QSPI Data Xfer type (Device Write Instruction Configuration Register)
PP	1	1	1	0	0	0
DIFP (Dual Input Fast Program)	1	1	2	0	0	1
DIEFP (Dual Input Extended Fast Program)	1	2	2	0	1	1
QIFP (Quad Input Fast Program)	1	1	4	0	0	2
QIEFP (Quad Input Extended Fast Program)	1	4	4	0	2	2
DCPP (Dual Command Fast Program)	2	2	2	1	Don't care	Don't care
QCPP (Quad Command Fast Program)	4	4	4	2	Don't care	Don't care

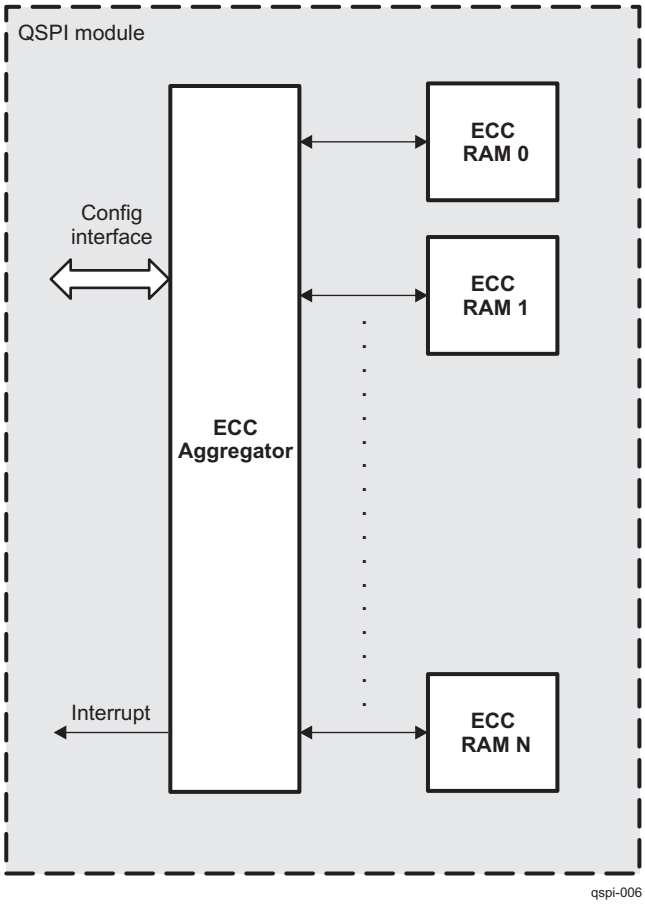
11.15.4.14 ECC Aggregator module

This section describes the architecture and functional details of the ECC Aggregator module.

11.15.4.14.1 Overview

Figure 11-1310 shows the ECC Aggregator module block diagram.

Figure 11-1310. ECC Aggregator Block Diagram



The ECC Aggregator module supports the following general features:

- Automatically detects and connects all the RAM ECC interfaces in the QSPI module.
- Provides a mechanism to control and monitor the ECC RAMs in the QSPI module.
- Provides software access to all the ECC related registers.
- Supports software readable status of ECC single/double-bit errors and associated info such as RAM address and data bit(s) that are in error.
- Aggregates level pending status from the ECC RAMs into a single interrupt to the host.

The following feature is not supported:

- Statistics such as tracking the number of single and double-bit errors. If needed these operations can be handled by software.

11.15.4.14.2 ECC Aggregator Registers

The ECC Aggregator module supports all the ECC control, status and interrupt registers for all the ECC RAMs in the QSPI module. The ECC registers are memory mapped to a 1 KB address space in the configuration port of the QSPI module (address range: from 0294 0400h to 0294 07FFh).

There are 3 types of register accesses in the ECC aggregator module:

- Global registers that are common to all ECC RAMs. This includes the revision and ECC vector registers. Every ECC RAM serviced by the ECC Aggregator module is assigned a unique ID by the module. Writing this ID to the ECC vector register selects the ECC RAM to be controlled or read the ECC status from.
- ECC control and status registers. These are specific to each ECC RAM and selected by the RAM ID written to the ECC vector register. The accesses to these registers are done via the configuration interface.
- Interrupt registers. These include the standard interrupt status, interrupt enable, clear and EOI (End Of Interrupt) registers that are part of a standard interrupt module.

11.15.4.14.3 Reads to ECC Control and Status Registers

Due to the long latencies in the order of several thousands of clock cycles on the configuration interface, the reads to the ECC control and status registers for each ECC RAM are triggered by writing a 'read message' to the ECC vector register as described below:

- Software writes the ECC ID to the QSPI_ECC_VECTOR[10-0] RAM_ID bit field, the 'read message' to the QSPI_ECC_VECTOR[15] TRIGGER_READ bit, and the read address to the QSPI_ECC_VECTOR[23-16] READ_ADDRESS bit field. The read address corresponds to any of the ECC control or status registers (QSPI_ECC_WRAPPER_REVISION, QSPI_ECC_CONTROL, QSPI_ECC_ERROR_CONTROL1, QSPI_ECC_ERROR_CONTROL2, QSPI_ECC_ERROR_STATUS1, QSPI_ECC_ERROR_STATUS2) which require configuration interface access to the ECC RAM(s).
- Software then polls the QSPI_ECC_VECTOR[24] READ_DONE bit to check if it is '1'. This indicates that the read operation has completed on the configuration interface.
- Software reads the data from the ECC control or status register. The following clock cycle returns the read data.

11.15.4.14.4 ECC Interrupts

The ECC aggregator module aggregates all the level pending status from the ECC RAMs to a single EOI-handshake based interrupt to the host. Software is expected to follow the sequence described below:

- Software enables the interrupts for the ECC RAM(s) by writing to the QSPI_ECC_INT_ENABLE register.
- On receiving an interrupt, software checks the QSPI_ECC_INT_STATUS to see which ECC RAM(s) caused the error.
- Software writes the RAM ID in the QSPI_ECC_VECTOR[10-0] RAM_ID field along with the 'read message' to the QSPI_ECC_VECTOR[15] TRIGGER_READ bit (to trigger the read), writing the QSPI_ECC_ERROR_STATUS1 register address to the QSPI_ECC_VECTOR[23-16] READ_ADDRESS field. Software will need to load the 'read message' in the QSPI_ECC_VECTOR register again if it needs to read the QSPI_ECC_ERROR_STATUS2 register.
- Software polls the QSPI_ECC_VECTOR[24] READ_DONE bit. When this is set, it does a read of the QSPI_ECC_ERROR_STATUS1/QSPI_ECC_ERROR_STATUS2 registers.
- After the interrupt has been serviced, software will clear the interrupt status by writing to bits 8, 9 or 10 in the QSPI_ECC_ERROR_STATUS1 register depending on the type of ECC error.
- Software has to poll the QSPI_ECC_ERROR_STATUS1 register to guarantee that the status bit has been cleared. Otherwise there is no indication that the write has actually completed over the configuration interface.
- Software will write to the QSPI_ECC_EOI register to clear the interrupt.

11.15.5 QSPI Programming Guide

11.15.5.1 QSPI Configuration Sequence

The following sequence assumes the QSPI module is enabled and the SPI protocol mode is correctly configured (see Table 11-3212).

Table 11-3212. QSPI Configuration Sequence

Step
1. Set baud rate to correspond to < 25 MHz operation.
2. Issue a RDID (Read ID) command in STIG mode of operation. Store the returned data for future reference.
3. Bump up baud rate to desired speed.
a. If the desired rate is < 25 MHz, there is no need of data training (step 4).
b. If the desired rate is > 25 MHz but < 50 MHz then data training is required. Clock loopback mode is not required in this case and should not be used.
c. If the desired rate is > 50 MHz, loopback clock needs to be enabled and data training is required as well.
4. For data training the software should sweep through the 16 read capture delay values, 0 to 15, and for every value issue a RDID (Read ID) command in STIG mode of operation. Compare the returned value with the reference value read during slow MHz operation. Mark the delay value as valid or invalid based on the comparison result. The complete sweep operation should fetch atleast 3 valid taps. Select the central tap in the valid window for future flash accesses.

11.15.5.2 Direct Access Controller Programming Sequence

The following sequence assumes that QSPI module is enabled, baud rate programmed and data training, if required, done for the given baud rate (see Table 11-3213).

Table 11-3213. Direct Access Controller Programming Sequence

Step	Register/Bit Field/Programming Model	Value
1. Set address remap enable to 1 and chip select decode to 0 (in the QSPI Configuration Register).	QSPI_CONFIG_REG[16] ENB_AHB_ADDR_REMAP_FLD	1h
	QSPI_CONFIG_REG[9] PERIPH_SEL_DEC_FLD	0h
2. Program the remap address register with a byte aligned value, so that the final remap address is byte aligned (in the Remap Address Register).	QSPI_REMAP_ADDR_REG[31-0] VALUE_FLD	-h
3. Set DAC enable to 1 (in the QSPI Configuration Register).	QSPI_CONFIG_REG[7] ENB_DIR_ACC_CTRL_FLD	1h
4. Program chip select value (in QSPI Configuration Register).	QSPI_CONFIG_REG[13-10] PERIPH_CS_LINES_FLD	-h
5. Set the SPI modes by programming clock phase and clock polarity (in the QSPI Configuration Register).	QSPI_CONFIG_REG[2] SEL_CLK_PHASE_FLD	-h
	QSPI_CONFIG_REG[1] SEL_CLK_POL_FLD	-h
6. Program the QSPI Device Delay Register with required values (in the QSPI Device Delay Register).	QSPI_DEV_DELAY_REG	-h
7. Set number of bytes per block. Use 10h for Micron and Spansion as both have 65535 bytes per block (in the Device Size Configuration Register).	QSPI_DEV_SIZE_CONFIG_REG[20-16] BYTES_PER_SUBSECTOR_FLD	10h
8. Set number of bytes per page. Use 100h for Micron and Spansion since both have 256 bytes per page (in the Device Size Configuration Register).	QSPI_DEV_SIZE_CONFIG_REG[15-4] BYTES_PER_DEVICE_PAGE_FLD	100h
9. Set number of address bytes. Use 2h for Micron and Spansion since both memories accept 24 bit address (in the Device Size Configuration Register).	QSPI_DEV_SIZE_CONFIG_REG[3-0] NUM_ADDR_BYTES_FLD	2h

Table 11-3213. Direct Access Controller Programming Sequence (continued)

Step	Register/Bit Field/Programming Model	Value
10. Program the write and read opcode with proper values based on the flash memories being accessed. Make sure the values programmed for data transfer type, address transfer type and instruction type matches the opcode value programmed (in the Device Read Instruction Configuration Register and the Device Write Instruction Configuration Register).	QSPI_DEV_INSTR_RD_CONFIG_REG	-h
	QSPI_DEV_INSTR_WR_CONFIG_REG	-h
11. Set write dummy cycle to 0 and read dummy cycle to a value based on the opcode. If Micron then 15, if Spansion the 4,6 or 8 (in the Device Read Instruction Configuration Register and the Device Write Instruction Configuration Register).	QSPI_DEV_INSTR_RD_CONFIG_REG[28-24] DUMMY_RD_CLK_CYCLES_FLD	-h
	QSPI_DEV_INSTR_WR_CONFIG_REG[28-24] DUMMY_WR_CLK_CYCLES_FLD	0h
12. Set mode bit to 0 (in the Device Read Instruction Configuration Register).	QSPI_DEV_INSTR_RD_CONFIG_REG[20] MODE_BIT_ENABLE_FLD	0h
13. If the WEL Disable bit in the Device Write Instruction Configuration Register is '0', enable the flash for write operation. This is done through STIG. (in the Flash Command Control Register).	QSPI_DEV_INSTR_WR_CONFIG_REG	WEL Disable bit is '0'
	QSPI_FLASH_CMD_CTRL_REG[15] ENB_WRITE_DATA_FLD	1h
a. Set command opcode to 06h, with rest of the fields programmed to default values.	QSPI_FLASH_CMD_CTRL_REG[31-24] CMD_OPCODE_FLD	06h
b. Then set execute command, in order to initiate STIG transaction to the flash.	QSPI_FLASH_CMD_CTRL_REG[0] CMD_EXEC_FLD	1h
14. In order to perform a transaction to the flash in Dual I/O mode or Quad I/O mode, following sequence needs to be followed.		
a. Set command opcode to 61h, with write data enabled (in the Flash Command Control Register).	QSPI_FLASH_CMD_CTRL_REG[31-24] CMD_OPCODE_FLD	61h
b. Program the write data register with a value BFh if Dual and 7Fh if Quad, and number of bytes to 1 byte (in the Flash Command Write Data Register (Lower)).	QSPI_FLASH_WR_DATA_LOWER_REG[31-0] DATA_FLD	BFh (Dual mode)
		7Fh (Quad mode)
c. Then set execute command, in order to initiate STIG transaction to the flash (in the Flash Command Control Register). The above opcode and data values are for Micron memories. For Spansion memories, replace the opcode with 01h and value with 0200h for Quad I/O operation. No programming is required for SERIAL and Dual I/O operation.	QSPI_FLASH_CMD_CTRL_REG[0] CMD_EXEC_FLD	1h
	QSPI_FLASH_CMD_CTRL_REG[31-24] CMD_OPCODE_FLD	01h
	QSPI_FLASH_WR_DATA_LOWER_REG[31-0] DATA_FLD	0200h (Quad mode)
15. Check the flash memory register, to see if it has been programmed as required.		
a. Set command opcode to 65h, and read data enable to 1 (in the Flash Command Control Register).	QSPI_FLASH_CMD_CTRL_REG[31-24] CMD_OPCODE_FLD	65h
	QSPI_FLASH_CMD_CTRL_REG[15] ENB_WRITE_DATA_FLD	1h
b. Read the value from the read data register, and compare if the value matches with the written data (in the Flash Command Read Data Register (Lower)).	QSPI_FLASH_RD_DATA_LOWER_REG	
16. Initiate a data bus read/write burst/single transactions. Make sure the data bus address is byte aligned.		

NOTE: As long as direct access is enabled, and even though the address lies within the trigger address range with indirect access disabled, the transaction goes as direct access.

NOTE: For direct accesses, the transaction cannot be started immediately after programming the write and read opcode in the device configuration register. Due to synchronization delays software is expected to wait 3 QSPI_DATA_BUS_CLK cycles before starting the DAC operation.

NOTE: DAC mode needs to be enabled for proper legacy mode operation.

11.15.5.3 Indirect Access Controller Programming Sequence

The following sequence assumes that QSPI is enabled, baud rate programmed and data training, if required, done for the given baud rate (see Table 11-3214).

Table 11-3214. Indirect Access Controller Programming Sequence

Step	Register/Bit Field/Programming Model	Value
1. Program chip select (in the QSPI Configuration Register).	QSPI_CONFIG_REG[13-10] PERIPH_CS_LINES_FLD	-h
2. Set SPI mode by setting clock phase and clock polarity as required (in the QSPI Configuration Register).	QSPI_CONFIG_REG[2] SEL_CLK_PHASE_FLD	-h
	QSPI_CONFIG_REG[1] SEL_CLK_POL_FLD	-h
3. Program the QSPI Device Delay Register with necessary values (in the QSPI Device Delay Register).	QSPI_DEV_DELAY_REG	-h
4. Set number of bytes per block. Use 10h for both Micron and Spansion as they have 65535 bytes per block (in the Device Size Configuration Register 14h).	QSPI_DEV_SIZE_CONFIG_REG[20-16] BYTES_PER_SUBSECTOR_FLD	10h
5. Set number of bytes per page. Use 100h for Micron and Spansion as they both have 256 bytes per page (in the Device Size Configuration Register).	QSPI_DEV_SIZE_CONFIG_REG[15-4] BYTES_PER_DEVICE_PAGE_FLD	100h
6. Set number of address bytes. Use 2h since both memories accept 24 bit address (in the Device Size Configuration Register).	QSPI_DEV_SIZE_CONFIG_REG[3-0] NUM_ADDR_BYTES_FLD	2h
7. Program the write and read opcode with proper values based on the flash memories being accessed. Make sure the values programmed for data transfer type, address transfer type and instruction type matches the opcode value programmed (in the Device Read Instruction Configuration Register and the Device Write Instruction Configuration Register).	QSPI_DEV_INSTR_RD_CONFIG_REG	-h
	QSPI_DEV_INSTR_WR_CONFIG_REG	-h
8. Set write dummy cycle to 0 and read dummy cycle to a value based on the opcode. If Micron then 15, if Spansion the 4, 6 or 8 (in the Device Read Instruction Configuration Register and the Device Write Instruction Configuration Register).	QSPI_DEV_INSTR_RD_CONFIG_REG[28-24] DUMMY_RD_CLK_CYCLES_FLD	-h
	QSPI_DEV_INSTR_WR_CONFIG_REG[28-24] DUMMY_WR_CLK_CYCLES_FLD	0h
9. Set mode bits to 0 (in the Device Read Configuration Instruction Register).	QSPI_DEV_INSTR_RD_CONFIG_REG[20] MODE_BIT_ENABLE_FLD	0h
10. If the WEL Disable bit in the Device Write Instruction Configuration Register is '0', enable the flash for write operation. This is done through STIG (in the Flash Command Control Register).	QSPI_DEV_INSTR_WR_CONFIG_REG	WEL Disable bit is '0'
	QSPI_FLASH_CMD_CTRL_REG[15] ENB_WRITE_DATA_FLD	1h
a. Set command opcode to 06h, with reset of the fields programmed to default value.	QSPI_FLASH_CMD_CTRL_REG[31-24] CMD_OPCODE_FLD	06h
b. Then set execute command, in order to initiate STIG transaction to the flash.	QSPI_FLASH_CMD_CTRL_REG[0] CMD_EXEC_FLD	1h
11. In order to perform a transaction to the flash in Dual I/O mode or Quad I/O, following sequence needs to be followed.		
a. Set command opcode to 61h, with write data enabled in the flash command control register (in the Flash Command Control Register).	QSPI_FLASH_CMD_CTRL_REG[31-24] CMD_OPCODE_FLD	61h
b. Program the write data register with a value BFh if Dual and 7Fh if Quad, and number of bytes to 1 byte (in the Flash Command Write Data Register (Lower)).	QSPI_FLASH_WR_DATA_LOWER_REG[31-0] DATA_FLD	BFh (Dual mode)
		7Fh (Quad mode)

Table 11-3214. Indirect Access Controller Programming Sequence (continued)

Step	Register/Bit Field/Programming Model	Value
c. Then set execute command, in order to initiate STIG transaction to the flash (in the Flash Command Control Register). The above opcode and data values are for Micron memories. For Spansion memories, replace the opcode with 01h and value with 0200h for Quad I/O operation. No programming is required for SERIAL and Dual I/O operation).	QSPI_FLASH_CMD_CTRL_REG[0] CMD_EXEC_FLD	1h
	QSPI_FLASH_CMD_CTRL_REG[31-24] CMD_OPCODE_FLD	01h
	QSPI_FLASH_WR_DATA_LOWER_REG[31-0] DATA_FLD	0200h (Quad mode)
12. Check the flash memory register, to see if it has been programmed as required.		
a. Set command opcode to 65h, and read data enable to 1 (in the Flash Command Control Register).	QSPI_FLASH_CMD_CTRL_REG[31-24] CMD_OPCODE_FLD	65h
	QSPI_FLASH_CMD_CTRL_REG[15] ENB_WRITE_DATA_FLD	1h
b. Read the value from the read data register, and compare if the value matches with the written data (in Flash Command Read Data Register (Lower)).	QSPI_FLASH_RD_DATA_LOWER_REG	
13. If targeting the flash (Micron memory) to perform transaction in EXTENDED I/O mode or Spansion memory in SERIAL or Dual I/O mode, then the above three steps can be skipped.		
14. Program the SRAM partition and watermark registers with appropriate values.	QSPI_SRAM_PARTITION_CFG_REG	-h
	QSPI_INDIRECT_READ_XFER_WATERMARK_REG	-h
	QSPI_INDIRECT_WRITE_XFER_WATERMARK_REG	-h
15. Program the INDAC write control and read control register with proper start address, trigger address, write and read number of bytes.	QSPI_INDIRECT_READ_XFER_CTRL_REG	-h
	QSPI_INDIRECT_WRITE_XFER_CTRL_REG	-h
	QSPI_INDIRECT_READ_XFER_START_REG	-h
	QSPI_INDIRECT_WRITE_XFER_START_REG	-h
	QSPI_IND_AHB_ADDR_TRIGGER_REG	-h
	QSPI_INDIRECT_READ_XFER_NUM_BYTES_REG	-h
	QSPI_INDIRECT_WRITE_XFER_NUM_BYTES_REG	-h
16. For writes:		
a. Set the start INDAC write field of the INDAC write control register to 1. Wait for couple of cycles of QSPI_REF_CLK (since this bit is internally synchronized by the QSPI module).	QSPI_INDIRECT_WRITE_XFER_CTRL_REG[0] START_FLD	1h
b. Initiate a data bus Write burst/single transaction in order to start the write transaction. The transactions are recommended to start at the trigger address and shouldn't be longer than 64 bytes.		
c. Poll indirect write completion status bit, then wait until the transaction is complete at the SPI end.	QSPI_INDIRECT_WRITE_XFER_CTRL_REG[2] WR_STATUS_FLD	
17. For reads:		
a. Set start INDAC read field of the INDAC read control register to 1. Wait for couple of cycles of QSPI_REF_CLK until this bit is internally synchronized by the QSPI module.	QSPI_INDIRECT_READ_XFER_CTRL_REG[0] START_FLD	1h
b. Monitor SRAM fill level.	QSPI_SRAM_FILL_REG	
c. Initiate a data bus Read burst/single transaction when enough data is available. The transactions are recommended to start at the trigger address and shouldn't be longer than 64 bytes.		
d. Poll for indirect read completion status bit to be set.	QSPI_INDIRECT_READ_XFER_CTRL_REG[2] RD_STATUS_FLD	
Make sure that the trigger address is 64 byte aligned, and the complete transaction is always within the trigger boundary (trigger address to trigger address + 16). Also make sure the INDAC start address is word aligned.		

NOTE: For INDAC burst accesses, the address of every data bus phase should be within the trigger range. Burst start address should be within trigger and trigger + 64 byte limit and burst start address + byte-count should never exceed trigger + 64 byte limit. Recommended access scheme would be to always start at trigger address and not do bursts greater than 64 bytes.

NOTE: Indirect write status bit gets set before the last 12 bytes of SPI transaction are sent. Hence the software should wait for about 2.5 us before starting another transaction.

NOTE: In indirect access, execute QSPI_FLASH_CMD_CTRL_REG[0] CMD_EXEC_FLD bit has to be set before data bus transaction is pushed through.

11.15.5.4 Write Completion Polling for Both DAC and INDAC

If polling for write completion at the flash end is required then following steps needs to be flowed, before initiating any transaction (in the QSPI_WRITE_COMPLETION_CTRL_REG register). The following sequence assumes that QSPI module is enabled, baud rate programmed and data training, if required, done for the given baud rate (see Table 11-3215).

Table 11-3215. Write Completion Polling for Both DAC and INDAC

Step	Register/Bit Field/Programming Model	Value
1. For Micron memory:		
a. Program the opcode value to 70h	QSPI_WRITE_COMPLETION_CTRL_REG[7-0] OPCODE	70h
b. Polling bit index to value 7h	QSPI_WRITE_COMPLETION_CTRL_REG[10-8] POLLING_BIT_INDEX	7h
c. Polling polarity to 1	QSPI_WRITE_COMPLETION_CTRL_REG[13] POLLING_POLARITY	1h
d. Disable polling to 0 by default	QSPI_WRITE_COMPLETION_CTRL_REG[14] DISABLE_POLLING	0h
e. Poll count to 1	QSPI_WRITE_COMPLETION_CTRL_REG[23-16] POLL_COUNT	1h
2. For Spansion memory:		
a. Program the opcode value to 05h	QSPI_WRITE_COMPLETION_CTRL_REG[7-0] OPCODE	05h
b. Polling bit index to value 0h	QSPI_WRITE_COMPLETION_CTRL_REG[10-8] POLLING_BIT_INDEX	0h
c. Polling polarity to 0	QSPI_WRITE_COMPLETION_CTRL_REG[13] POLLING_POLARITY	0h
d. Disable polling to 0 by default	QSPI_WRITE_COMPLETION_CTRL_REG[14] DISABLE_POLLING	0h
e. Poll count to 1	QSPI_WRITE_COMPLETION_CTRL_REG[23-16] POLL_COUNT	1h
The controller polls until the poll count value programmed in the Polling Expiration Register is reached. An interrupt gets generated when Poll count reaches the value programmed in the register.		

NOTE: Flash write status poll is done more times than the programmed value in the QSPI_WRITE_COMPLETION_CTRL_REG[23-16] POLL_COUNT field. The status value, though, is discarded if polling is disabled.

11.15.5.5 ECC Programming Sequence

Table 11-3216. ECC Programming Sequence

Step	Register/Bit Field/Programming Model	Value
1. Write into ECC Interrupt Enable register to set the interrupt bit if needed in ECC Interrupt Set Register (address: 480h).	QSPI_ECC_INT_ENABLE[N-1:0] BITMASK	1h
2. Set the ECC enable, ECC Check bits to 1 (enabled by default) and force sec/ded bits in ECC Control Register (address: 414h).	QSPI_ECC_CONTROL[0] ECC_ENABLE	1h
	QSPI_ECC_CONTROL[1] ECC_CHECK	1h
	QSPI_ECC_CONTROL[3] FORCE_SEC	1h
	QSPI_ECC_CONTROL[4] FORCE_DED	1h
3. Then write to ECC Vector Register (address: 408h) with bits [23-16] set to the actual address to be updated which in this case is the ECC Control register and bit 15 to 0 to indicate a write.	QSPI_ECC_VECTOR[23-16] READ_ADDRESS	414h
	QSPI_ECC_VECTOR[15] TRIGGER_READ	0h

Table 11-3216. ECC Programming Sequence (continued)

Step	Register/Bit Field/Programming Model	Value
4. Set the bit position1 to be corrupted and the ecc row to be corrupted in ECC Control1 register (address: 418h).	QSPI_ECC_ERROR_CONTROL1[31-16] ECC_BIT1	1h
	QSPI_ECC_ERROR_CONTROL1[15-0] ECC_ROW	1h
5. Then write to ECC Vector Register (address: 408h) with bits [23-16] set to the actual address to be updated which in this case is the ECC Control1 register and bit 15 to 0 to indicate a write.	QSPI_ECC_VECTOR[23-16] READ_ADDRESS	418h
	QSPI_ECC_VECTOR[15] TRIGGER_READ	0h
6. Set the bit position2 to be corrupted in ECC Control2 register (address: 81Ch).	QSPI_ECC_ERROR_CONTROL2[15-0] ECC_BIT2	1h
7. Then write to ECC Vector Register (address: 408h) with bits [23-16] set to the actual address to be updated which in this case is the ECC Control2 register and bit 15 to 0 to indicate a write.	QSPI_ECC_VECTOR[23-16] READ_ADDRESS	81Ch
	QSPI_ECC_VECTOR[15] TRIGGER_READ	0h
8. To ensure that writes are complete, write into ECC Vector register with [23-16] set to ECC Control register and bit 15 to 1 to initiate a read.	QSPI_ECC_VECTOR[23-16] READ_ADDRESS	414h
	QSPI_ECC_VECTOR[15] TRIGGER_READ	1h
9. Keep Polling the ECC Vector register till the 23rd bit (read_done) bit becomes a 1.	QSPI_ECC_VECTOR[24] READ_DONE	1h (polling)
10. Then, read the ECC Control register to get the read value.	QSPI_ECC_VECTOR[23-16] READ_ADDRESS	414h
	QSPI_ECC_VECTOR[15] TRIGGER_READ	1h
11. If SRAM read is performed on the particular row in the ECC Control1 register (by either the flash controller or the software) and if force_sec/force_ded bits are set, an interrupt is raised if enabled.		
12. Read the ECC status registers by using the ECC Vector register method as mentioned in steps 8-10.	QSPI_ECC_VECTOR[23-16] READ_ADDRESS	420h
	QSPI_ECC_VECTOR[15] TRIGGER_READ	1h
	QSPI_ECC_VECTOR[24] READ_DONE	1h (polling)
	QSPI_ECC_VECTOR[23-16] READ_ADDRESS	420h
	QSPI_ECC_VECTOR[15] TRIGGER_READ	1h
	QSPI_ECC_VECTOR[23-16] READ_ADDRESS	424h
	QSPI_ECC_VECTOR[15] TRIGGER_READ	1h
	QSPI_ECC_VECTOR[24] READ_DONE	1h (polling)
	QSPI_ECC_VECTOR[23-16] READ_ADDRESS	424h
	QSPI_ECC_VECTOR[15] TRIGGER_READ	1h
13. Clear the status bits by writing back 1 into ECC status register using the ECC Vector register method as in steps 2-3.	QSPI_ECC_CONTROL[0] ECC_ENABLE	1h
	QSPI_ECC_CONTROL[1] ECC_CHECK	1h
	QSPI_ECC_CONTROL[3] FORCE_SEC	1h
	QSPI_ECC_CONTROL[4] FORCE_DED	1h
	QSPI_ECC_VECTOR[23-16] READ_ADDRESS	420h
	QSPI_ECC_VECTOR[15] TRIGGER_READ	0h
14. Finally, write into EOI register to clear the interrupt.	QSPI_ECC_EOI[0] EOI_WR	0h

11.15.6 QSPI Registers

Table 11-3218 lists the memory-mapped registers for the QSPI module. All register offset addresses not listed in Table 11-3218 should be considered as reserved locations and the register contents should not be modified.

Table 11-3217. QSPI Instances

Instance	Base Address
QSPI	0294 0000h

Table 11-3218. QSPI Registers

Offset	Acronym	Register Name	QSPI Physical Address	Section
0h	QSPI_CONFIG_REG	QSPI Configuration Register	0294 0000h	Section 11.15.6.1
4h	QSPI_DEV_INSTR_RD_CONFIG_REG	Device Read Instruction Configuration Register	0294 0004h	Section 11.15.6.2
8h	QSPI_DEV_INSTR_WR_CONFIG_REG	Device Write Instruction Configuration Register	0294 0008h	Section 11.15.6.3
Ch	QSPI_DEV_DELAY_REG	QSPI Device Delay Register	0294 000Ch	Section 11.15.6.4
10h	QSPI_RD_DATA_CAPTURE_REG	Read Data Capture Register	0294 0010h	Section 11.15.6.5
14h	QSPI_DEV_SIZE_CONFIG_REG	Device Size Configuration Register	0294 0014h	Section 11.15.6.6
18h	QSPI_SRAM_PARTITION_CFG_REG	SRAM Partition Configuration Register	0294 0018h	Section 11.15.6.7
1Ch	QSPI_IND_AHB_ADDR_TRIGGER_REG	Indirect AHB Address Trigger Register	0294 001Ch	Section 11.15.6.8
24h	QSPI_REMAP_ADDR_REG	Remap Address Register	0294 0024h	Section 11.15.6.9
28h	QSPI_MODE_BIT_CONFIG_REG	Mode Bit Configuration Register	0294 0028h	Section 11.15.6.10
2Ch	QSPI_SRAM_FILL_REG	SRAM Fill Level Register	0294 002Ch	Section 11.15.6.11
30h	QSPI_TX_THRESH_REG	TX Threshold Register	0294 0030h	Section 11.15.6.12
34h	QSPI_RX_THRESH_REG	RX Threshold Register	0294 0034h	Section 11.15.6.13
38h	QSPI_WRITE_COMPLETION_CTRL_REG	Write Completion Control Register	0294 0038h	Section 11.15.6.14
3Ch	QSPI_NO_OF_POLLS_BEF_EXP_REG	Polling Expiration Register	0294 003Ch	Section 11.15.6.15
40h	QSPI_IRQ_STATUS_REG	Interrupt Status Register	0294 0040h	Section 11.15.6.16
44h	QSPI_IRQ_MASK_REG	Interrupt Mask	0294 0044h	Section 11.15.6.17
50h	QSPI_LOWER_WR_PROT_REG	Lower Write Protection Register	0294 0050h	Section 11.15.6.18
54h	QSPI_UPPER_WR_PROT_REG	Upper Write Protection Register	0294 0054h	Section 11.15.6.19
58h	QSPI_WR_PROT_CTRL_REG	Write Protection Control Register	0294 0058h	Section 11.15.6.20
60h	QSPI_INDIRECT_READ_XFER_CTRL_REG	Indirect Read Transfer Control Register	0294 0060h	Section 11.15.6.21
64h	QSPI_INDIRECT_READ_XFER_WATERMARK_REG	Indirect Read Transfer Watermark Register	0294 0064h	Section 11.15.6.22

Table 11-3218. QSPI Registers (continued)

Offset	Acronym	Register Name	QSPI Physical Address	Section
68h	QSPI_INDIRECT_READ_XFER_START_REG	Indirect Read Transfer Start Address Register	0294 0068h	Section 11.15.6.23
6Ch	QSPI_INDIRECT_READ_XFER_NUM_BYTES_REG	Indirect Read Transfer Number Bytes Register	0294 006Ch	Section 11.15.6.24
70h	QSPI_INDIRECT_WRITE_XFER_CTRL_REG	Indirect Write Transfer Control Register	0294 0070h	Section 11.15.6.25
74h	QSPI_INDIRECT_WRITE_XFER_WATERMARK_REG	Indirect Write Transfer Watermark Register	0294 0074h	Section 11.15.6.26
78h	QSPI_INDIRECT_WRITE_XFER_START_REG	Indirect Write Transfer Start Address Register	0294 0078h	Section 11.15.6.27
7Ch	QSPI_INDIRECT_WRITE_XFER_NUM_BYTES_REG	Indirect Write Transfer Number Bytes Register	0294 007Ch	Section 11.15.6.28
90h	QSPI_FLASH_CMD_CTRL_REG	Flash Command Control Register	0294 0090h	Section 11.15.6.29
94h	QSPI_FLASH_CMD_ADDR_REG	Flash Command Address Registers	0294 0094h	Section 11.15.6.30
A0h	QSPI_FLASH_RD_DATA_LOWER_REG	Flash Command Read Data Register (Lower)	0294 00A0h	Section 11.15.6.31
A4h	QSPI_FLASH_RD_DATA_UPPER_REG	Flash Command Read Data Register (Upper)	0294 00A4h	Section 11.15.6.32
A8h	QSPI_FLASH_WR_DATA_LOWER_REG	Flash Command Write Data Register (Lower)	0294 00A8h	Section 11.15.6.33
ACh	QSPI_FLASH_WR_DATA_UPPER_REG	Flash Command Write Data Register (Upper)	0294 00ACh	Section 11.15.6.34
B0h	QSPI_POLLING_FLASH_STATUS_REG	Polling Flash Status Register	0294 00B0h	Section 11.15.6.35
FCh	QSPI_MODULE_ID_REG	Module ID Register	0294 00FCh	Section 11.15.6.36
400h	QSPI_ECC_REVISION	ECC Revision Register	0294 0400h	Section 11.15.6.37
408h	QSPI_ECC_VECTOR	ECC Vector Register	0294 0408h	Section 11.15.6.38
40Ch	QSPI_ECC_MISC_STATUS	ECC Miscellaneous Status Register	0294 040Ch	Section 11.15.6.39
410h	QSPI_ECC_WRAPPER_REVISION	ECC Revision Wrapper Register	0294 0410h	Section 11.15.6.40
414h	QSPI_ECC_CONTROL	ECC Control Register	0294 0414h	Section 11.15.6.41
418h	QSPI_ECC_ERROR_CONTROL1	ECC Control1 Register	0294 0418h	Section 11.15.6.42
41Ch	QSPI_ECC_ERROR_CONTROL2	ECC Control2 Register	0294 041Ch	Section 11.15.6.43
420h	QSPI_ECC_ERROR_STATUS1	ECC Status1 Register	0294 0420h	Section 11.15.6.44
424h	QSPI_ECC_ERROR_STATUS2	ECC Status2 Register	0294 0424h	Section 11.15.6.45
43Ch	QSPI_ECC_EOI	ECC EOI Register	0294 043Ch	Section 11.15.6.46
440h to 47Ch	QSPI_ECC_INT_STATUS_0 to QSPI_ECC_INT_STATUS_15	ECC Interrupt Status Registers	0294 0440h to 0294 047Ch	Section 11.15.6.47
480h to 4BCh	QSPI_ECC_INT_ENABLE_0 to QSPI_ECC_INT_ENABLE_15	ECC Interrupt Enable Registers	0294 0480h to 0294 04BCh	Section 11.15.6.48
4C0h to 4FCh	QSPI_ECC_INT_CLEAR_0 to QSPI_ECC_INT_CLEAR_15	ECC Interrupt Clear Registers	0294 04C0h to 0294 04FCh	Section 11.15.6.49

11.15.6.1 QSPI_CONFIG_REG Register (Offset = 0h) [reset = 80780081h]

QSPI_CONFIG_REG is shown in Figure 11-1311 and described in Table 11-3220.

Table 11-3219. QSPI_CONFIG_REG Instances

Instance	Physical Address
QSPI	0294 0000h

Figure 11-1311. QSPI_CONFIG_REG Register

31	30	29	28	27	26	25	24
QSPI_IDLE_FLD	RESERVED						
R-1h	R-0h						
23	22	21	20	19	18	17	16
RESERVED	MSTR_BAUD_DIV_FLD				ENTER_XIP_MODE_IMM_FLD	ENTER_XIP_MODE_FLD	ENB_AHB_ADDR_REMAP_FLD
R-0h	R/W-Fh				R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED		PERIPH_CS_LINES_FLD				PERIPH_SEL_DEC_FLD	ENB_LEGACY_IP_MODE_FLD
R-0h		R/W-0h				R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
ENB_DIR_ACC_CTLR_FLD	RESERVED				SEL_CLK_PHASE_FLD	SEL_CLK_POL_FLD	ENB_QSPI_FLD
R/W-1h	R-0h				R/W-0h	R/W-0h	R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3220. QSPI_CONFIG_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	QSPI_IDLE_FLD	R	1h	Serial interface and QSPI pipeline is IDLE: This is a STATUS read-only bit. Note this is a retimed signal, so there will be some inherent delay on the generation of this status signal.
30-23	RESERVED	R	0h	Reserved
22-19	MSTR_BAUD_DIV_FLD	R/W	Fh	Master Mode Baud Rate Divisor (2 to 32): SPI baud rate = QSPI_REF_CLK / BD. Where Baud Devisor (BD) is: 0000 = 2 0001 = 4 0010 = 6 0011 = 8 0100 = 10 0101 = 12 0110 = 14 0111 = 16 1000 = 18 … 1111 = 32 Set this register field up before enabling the QSPI module.

Table 11-3220. QSPI_CONFIG_REG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	ENTER_XIP_MODE_IMM_FLD	R/W	0h	Enter XIP Mode immediately: 0h: If XIP is enabled, then setting to 0 will cause the controller to exit XIP mode on the next READ instruction. 1h: Operate the device in XIP mode immediately. Use this register bit when the external device wakes up in XIP mode (as per the contents of its non- volatile configuration register). The controller will assume the next READ instruction will be passed to the device as an XIP instruction, and therefore will not require the READ opcode to be transferred. NOTE: To exit XIP mode, this bit should be set to 0. This will take effect in the attached device only after the next READ instruction is executed. Software therefore should ensure that at least one READ instruction is requested after resetting this bit in order to be sure that XIP mode is exited.
17	ENTER_XIP_MODE_FLD	R/W	0h	Enter XIP Mode on next READ: 0h: If XIP is enabled, then setting to 0 will cause the controller to exit XIP mode on the next READ instruction, 1h: If XIP is disabled, then setting to 1 will inform the controller that the device is ready to enter XIP on the next READ instruction. The controller will therefore send the appropriate command sequence, including mode bits to cause the device to enter XIP mode. Use this register bit after the controller has ensured the FLASH device has been configured to be ready to enter XIP mode. NOTE: To exit XIP mode, this bit should be set to 0. This will take effect in the attached device only after the next READ instruction is executed. Software should therefore ensure that at least one READ instruction is requested after resetting this bit before it can be sure XIP mode in the device is exited.
16	ENB_AHB_ADDR_REMAP_FLD	R/W	0h	Enable AHB Address Re-mapping (Direct Access Mode Only): When set to 1, the incoming AHB address will be adapted and sent to the FLASH device as address + N, where N is the value stored in the QSPI_REMAP_ADDR_REG register.
15-14	RESERVED	R	0h	Reserved
13-10	PERIPH_CS_LINES_FLD	R/W	0h	Peripheral Chip Select Lines: If QSPI_CONFIG_REG[9] PERIPH_SEL_DEC_FLD is set to 0, ss[3:0] are output thus: ss[3:0] → n_ss_out[3:0] xxx0 → 1110 xx01 → 1101 x011 → 1011 0111 → 0111 1111 → 1111 (no peripheral selected) else ss[3:0] directly drives n_ss_out[3:0]
9	PERIPH_SEL_DEC_FLD	R/W	0h	Peripheral select decode: 0h: only 1 of 4 selects n_ss_out[3:0] is active, 1h: allow external 4-to-16 decode (n_ss_out = ss).
8	ENB_LEGACY_IP_MODE_FLD	R/W	0h	Legacy QSPI Mode Enable: 0h: Use Direct Access Controller / Indirect Access Controller. 1h: Legacy Mode is enabled. In this mode, any write to the controller via the AHB is serialized and sent to the FLASH device. Any valid AHB read will pop the internal RX-FIFO, retrieving data that was forwarded by the external FLASH device on the SPI lines, 4 , 2 or 1 byte transfers are permitted.

Table 11-3220. QSPI_CONFIG_REG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	ENB_DIR_ACC_CTLR_FLD	R/W	1h	Enable Direct Access Controller: 0h: disable the Direct Access Controller once current transfer of the data word is complete, 1h: enable the Direct Access Controller. When the Direct Access Controller and Indirect Access Controller are both disabled, all AHB requested are completed with an error response.
6-3	RESERVED	R	0h	Reserved
2	SEL_CLK_PHASE_FLD	R/W	0h	Clock phase: If SEL_CLK_POL_FLD = 0h: 0h: Data launched on falling edge and captured on rising edge (Mode 0), 1h: Data launched on rising edge and captured on falling edge (Mode 1). If SEL_CLK_POL_FLD = 1h: 0h: Data launched on rising edge and captured on falling edge (Mode 2). 1h: Data launched on falling edge and captured on rising edge (Mode 3).
1	SEL_CLK_POL_FLD	R/W	0h	Clock polarity outside SPI word: 0h: the SPI clock is static low, 1h: the SPI clock is static high.
0	ENB_QSPI_FLD	R/W	1h	QSPI Enable: 0h: Disable the QSPI module. This should be done after the current data word transfer has already been completed, 1h: Enable the QSPI module.

Table 11-3221. Register Call Summary for QSPI_CONFIG_REG

QSPI Functional Description • Data Interface Address Remapping: [0] • Indirect read transfer process: [0] • Access Forwarding: [0] • QSPI Modes: [0][1] • Direct Access Controller (DAC): [0][1]
QSPI Registers • QSPI_IRQ_STATUS_REG Register (Offset = 40h) [reset = 100h]: [0] • QSPI Registers: [0] • QSPI_CONFIG_REG Register (Offset = 0h) [reset = 80780081h]: [0][1]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0][1][2] • Direct Access Controller Programming Sequence: [0][1][2][3][4][5]

11.15.6.2 QSPI_DEV_INSTR_RD_CONFIG_REG Register (Offset = 4h) [reset = 3h]

QSPI_DEV_INSTR_RD_CONFIG_REG is shown in Figure 11-1312 and described in Table 11-3223.

Table 11-3222. QSPI_DEV_INSTR_RD_CONFIG_REG Instances

Instance	Physical Address
QSPI	0294 0004h

Figure 11-1312. QSPI_DEV_INSTR_RD_CONFIG_REG Register

31	30	29	28	27	26	25	24
RESERVED			DUMMY_RD_CLK_CYCLES_FLD				
R-0h			R/W-0h				
23	22	21	20	19	18	17	16
RESERVED			MODE_BIT_ENABLE_FLD	RESERVED		DATA_XFER_TYPE_EXT_MODE_FLD	
R-0h			R/W-0h	R-0h		R/W-0h	
15	14	13	12	11	10	9	8
RESERVED		ADDR_XFER_TYPE_STD_MODE_FLD		RESERVED		INSTR_TYPE_FLD	
R-0h		R/W-0h		R-0h		R/W-0h	
7	6	5	4	3	2	1	0
RD_OPCODE_NON_XIP_FLD							
R/W-3h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3223. QSPI_DEV_INSTR_RD_CONFIG_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved
28-24	DUMMY_RD_CLK_CYCLES_FLD	R/W	0h	Dummy Read Clock Cycles: Number of dummy clock cycles required by device for read instruction.
23-21	RESERVED	R	0h	Reserved
20	MODE_BIT_ENABLE_FLD	R/W	0h	Mode Bit Enable: Set this field to 1 to ensure that the mode bits defined in the QSPI_MODE_BIT_CONFIG_REG register are sent following the address bytes.
19-18	RESERVED	R	0h	Reserved
17-16	DATA_XFER_TYPE_EXT_MODE_FLD	R/W	0h	Data Transfer Type for Standard SPI modes: 0h: SIO mode data is shifted to the device on QSPI_D0 only and from the device on QSPI_D1 only, 1h: Used for Dual Input/Output instructions. For data transfers, QSPI_D0 and QSPI_D1 are used as both inputs and outputs. 2h: Used for Quad Input/Output instructions. For data transfers, QSPI_D0,QSPI_D1,QSPI_D2 and QSPI_D3 are used as both inputs and outputs.
15-14	RESERVED	R	0h	Reserved
13-12	ADDR_XFER_TYPE_STD_MODE_FLD	R/W	0h	Address Transfer Type for Standard SPI modes: 0h: Addresses can be shifted to the device on QSPI_D0 only. 1h: Addresses can be shifted to the device on QSPI_D0 and QSPI_D1 only. 2h: Addresses can be shifted to the device on QSPI_D0, QSPI_D1, QSPI_D2 and QSPI_D3.

Table 11-3223. QSPI_DEV_INSTR_RD_CONFIG_REG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11-10	RESERVED	R	0h	Reserved
9-8	INSTR_TYPE_FLD	R/W	0h	Instruction Type: 0h: Use Standard SPI mode (instruction always shifted into the device on QSPI_D0 only). 1h: Use DIO-SPI mode (Instructions, Address and Data always sent on QSPI_D0 and QSPI_D1). 2h: Use QIO-SPI mode (Instructions, Address and Data always sent on QSPI_D0, QSPI_D1, QSPI_D2 and DQSPI_D3).
7-0	RD_OPCODE_NON_XIP_FLD	R/W	3h	Read Opcode in non-XIP mode: Read Opcode to use when not in XIP mode.

Table 11-3224. Register Call Summary for QSPI_DEV_INSTR_RD_CONFIG_REG

QSPI Functional Description • SPI Command Translation: [0] • Selecting the Flash Instruction Type: [0][1][2]
QSPI Registers • QSPI Registers: [0] • QSPI_DEV_INSTR_RD_CONFIG_REG Register (Offset = 4h) [reset = 3h]: [0]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0][1][2] • Direct Access Controller Programming Sequence: [0][1][2]

11.15.6.3 QSPI_DEV_INSTR_WR_CONFIG_REG Register (Offset = 8h) [reset = 2h]

QSPI_DEV_INSTR_WR_CONFIG_REG is shown in Figure 11-1313 and described in Table 11-3226.

Table 11-3225. QSPI_DEV_INSTR_WR_CONFIG_REG Instances

Instance	Physical Address
QSPI	0294 0008h

Figure 11-1313. QSPI_DEV_INSTR_WR_CONFIG_REG Register

31	30	29	28	27	26	25	24
RESERVED			DUMMY_WR_CLK_CYCLES_FLD				
R-0h			R/W-0h				
23	22	21	20	19	18	17	16
RESERVED						DATA_XFER_TYPE_EXT_MODE_FLD	
R-0h						R/W-0h	
15	14	13	12	11	10	9	8
RESERVED		ADDR_XFER_TYPE_STD_MODE_FLD		RESERVED			
R-0h		R/W-0h		R-0h			
7	6	5	4	3	2	1	0
WR_OPCODE_FLD							
R/W-2h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3226. QSPI_DEV_INSTR_WR_CONFIG_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved
28-24	DUMMY_WR_CLK_CYCLES_FLD	R/W	0h	Dummy Write Clock Cycles: Number of dummy clock cycles required by device for write instruction.
23-18	RESERVED	R	0h	Reserved
17-16	DATA_XFER_TYPE_EXT_MODE_FLD	R/W	0h	Data Transfer Type for Standard SPI modes: 0h: SIO mode data is shifted to the device on QSPI_D0 only and from the device on QSPI_D1 only. 1h: Used for Dual Input/Output instructions. For data transfers, QSPI_D0 and QSPI_D1 are used as both inputs and outputs. 2h: Used for Quad Input/Output instructions. For data transfers, QSPI_D0,QSPI_D1,QSPI_D2 and QSPI_D3 are used as both inputs and outputs.
15-14	RESERVED	R	0h	Reserved
13-12	ADDR_XFER_TYPE_STD_MODE_FLD	R/W	0h	Address Transfer Type for Standard SPI modes: 0h: Addresses can be shifted to the device on QSPI_D0 only. 1h: Addresses can be shifted to the device on QSPI_D0 and QSPI_D1 only. 2h: Addresses can be shifted to the device on QSPI_D0, QSPI_D1, QSPI_D2 and QSPI_D3.
11-8	RESERVED	R	0h	Reserved
7-0	WR_OPCODE_FLD	R/W	2h	Write Opcode: Write Opcode Field.

Table 11-3227. Register Call Summary for QSPI_DEV_INSTR_WR_CONFIG_REG

QSPI Functional Description • SPI Command Translation: [0][1] • Selecting the Flash Instruction Type: [0][1][2]
QSPI Registers • QSPI_DEV_INSTR_WR_CONFIG_REG Register (Offset = 8h) [reset = 2h]: [0] • QSPI Registers: [0]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0][1][2] • Direct Access Controller Programming Sequence: [0][1][2]

11.15.6.4 QSPI_DEV_DELAY_REG Register (Offset = Ch) [reset = 0h]

QSPI_DEV_DELAY_REG is shown in Figure 11-1314 and described in Table 11-3229.

This register is used to introduce relative delays into the generation of the master output signals. All timings are defined in cycles of the QSPI_REF_CLK/QSPI_CLK.

Table 11-3228. QSPI_DEV_DELAY_REG Instances

Instance	Physical Address
QSPI	0294 000Ch

Figure 11-1314. QSPI_DEV_DELAY_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D_NSS_FLD								D_BTWN_FLD							
R/W-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D_AFTER_FLD								D_INIT_FLD							
R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3229. QSPI_DEV_DELAY_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	D_NSS_FLD	R/W	0h	Clock Delay for Chip Select Deassert: Delay in master reference clocks for the length that the master mode chip select outputs are de-asserted between transactions. The minimum delay is always QSPI_CLK period to ensure the chip select is never re-asserted within an QSPI_CLK period.
23-16	D_BTWN_FLD	R/W	0h	Clock Delay for Chip Select Deactivation: Delay in master reference clocks between one chip select being de-activated and the activation of another. This is used to ensure a quiet period between the selection of two different slaves and requires the transmit FIFO to be empty.
15-8	D_AFTER_FLD	R/W	0h	Clock Delay for Last Transaction Bit: Delay in master reference clocks between last bit of current transaction and deasserting the device chip select (n_ss_out). By default, the chip select will be deasserted on the cycle following the completion of the current transaction.
7-0	D_INIT_FLD	R/W	0h	Clock Delay with n_ss_out: Delay in master reference clocks between setting n_ss_out low and first bit transfer.

Table 11-3230. Register Call Summary for QSPI_DEV_DELAY_REG

QSPI Functional Description • Read Data Capture: [0]
QSPI Registers • QSPI_DEV_DELAY_REG Register (Offset = Ch) [reset = 0h]: [0] • QSPI Registers: [0]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0] • Direct Access Controller Programming Sequence: [0]

11.15.6.5 QSPI_RD_DATA_CAPTURE_REG Register (Offset = 10h) [reset = 1h]

QSPI_RD_DATA_CAPTURE_REG is shown in Figure 11-1315 and described in Table 11-3232.

Table 11-3231. QSPI_RD_DATA_CAPTURE_REG Instances

Instance	Physical Address
QSPI	0294 0010h

Figure 11-1315. QSPI_RD_DATA_CAPTURE_REG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		SAMPLE_EDGE_SEL_FLD	DELAY_FLD				BYPASS_FLD
R-0h		R/W-0h	R/W-0h				R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3232. QSPI_RD_DATA_CAPTURE_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	SAMPLE_EDGE_SEL_FLD	R/W	0h	Sample edge selection (of the flash memory data outputs). 0h = data outputs from flash memory are sampled on falling edge of the QSPI_REF_CLK. 1h = data outputs from flash memory are sampled on rising edge of the QSPI_REF_CLK.
4-1	DELAY_FLD	R/W	0h	Read Delay: Delay the read data capturing logic by the programmed number of QSPI_REF_CLK cycles.
0	BYPASS_FLD	R/W	1h	Bypass: Bypass the adapted loopback clock circuit.

Table 11-3233. Register Call Summary for QSPI_RD_DATA_CAPTURE_REG

QSPI Functional Description • Read Data Capture: [0][1][2][3][4]
QSPI Registers • QSPI_RD_DATA_CAPTURE_REG Register (Offset = 10h) [reset = 1h]: [0] • QSPI Registers: [0]

11.15.6.6 QSPI_DEV_SIZE_CONFIG_REG Register (Offset = 14h) [reset = 101002h]

QSPI_DEV_SIZE_CONFIG_REG is shown in Figure 11-1316 and described in Table 11-3235.

Table 11-3234. QSPI_DEV_SIZE_CONFIG_REG Instances

Instance	Physical Address
QSPI	0294 0014h

Figure 11-1316. QSPI_DEV_SIZE_CONFIG_REG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED			BYTES_PER_SUBSECTOR_FLD				
R-0h			R/W-10h				
15	14	13	12	11	10	9	8
BYTES_PER_DEVICE_PAGE_FLD							
R/W-100h							
7	6	5	4	3	2	1	0
BYTES_PER_DEVICE_PAGE_FLD				NUM_ADDR_BYTES_FLD			
R/W-100h				R/W-2h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3235. QSPI_DEV_SIZE_CONFIG_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-21	RESERVED	R	0h	Reserved
20-16	BYTES_PER_SUBSECTOR_FLD	R/W	10h	Number of Bytes per Block: This is required by the controller for performing the write protection logic. The number of bytes per block must be a power of 2 number. 0 = 1 byte 1 = 2 bytes 2 = 4 bytes 3 = 8 bytes 16 = 65535 bytes ...
15-4	BYTES_PER_DEVICE_PAGE_FLD	R/W	100h	Number of Bytes per Device Page: This is required by the controller for performing FLASH writes up to and across page boundaries.
3-0	NUM_ADDR_BYTES_FLD	R/W	2h	Number of address Bytes: A value of 0 indicates 1 byte.

Table 11-3236. Register Call Summary for QSPI_DEV_SIZE_CONFIG_REG

QSPI Functional Description • Write Protection: [0]
QSPI Registers • QSPI_UPPER_WR_PROT_REG Register (Offset = 54h) [reset = 0h]: [0] • QSPI_LOWER_WR_PROT_REG Register (Offset = 50h) [reset = 0h]: [0] • QSPI Registers: [0] • QSPI_DEV_SIZE_CONFIG_REG Register (Offset = 14h) [reset = 101002h]: [0]

Table 11-3236. Register Call Summary for QSPI_DEV_SIZE_CONFIG_REG (continued)

QSPI Programming Guide • Indirect Access Controller programming sequence: [0][1][2] • Direct Access Controller Programming Sequence: [0][1][2]

11.15.6.7 QSPI_SRAM_PARTITION_CFG_REG Register (Offset = 18h) [reset = 80h]

QSPI_SRAM_PARTITION_CFG_REG is shown in Figure 11-1317 and described in Table 11-3238.

Table 11-3237. QSPI_SRAM_PARTITION_CFG_REG Instances

Instance	Physical Address
QSPI	0294 0018h

Figure 11-1317. QSPI_SRAM_PARTITION_CFG_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								ADDR_FLD							
R-0h																								R/W-80h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3238. QSPI_SRAM_PARTITION_CFG_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	ADDR_FLD	R/W	80h	Indirect Read Partition Size: Defines the size of the indirect read partition in the SRAM, in units of SRAM locations. By default, half of the SRAM is reserved for indirect read operation, and half for indirect write. The size of this register will scale with the depth of the SRAM.

Table 11-3239. Register Call Summary for QSPI_SRAM_PARTITION_CFG_REG

QSPI Functional Description • Accessing the SRAM: [0][1][2][3][4][5][6][7]
QSPI Registers • QSPI_SRAM_PARTITION_CFG_REG Register (Offset = 18h) [reset = 80h]: [0] • QSPI Registers: [0]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0]

11.15.6.8 QSPI_IND_AHB_ADDR_TRIGGER_REG Register (Offset = 1Ch) [reset = 0h]

QSPI_IND_AHB_ADDR_TRIGGER_REG is shown in Figure 11-1318 and described in Table 11-3241.

Table 11-3240. QSPI_IND_AHB_ADDR_TRIGGER_REG Instances

Instance	Physical Address
QSPI	0294 001Ch

Figure 11-1318. QSPI_IND_AHB_ADDR_TRIGGER_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR_FLD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3241. QSPI_IND_AHB_ADDR_TRIGGER_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR_FLD	R/W	0h	Indirect Trigger Address: This is the base address that will be used by the AHB Controller. When the incoming AHB read access address matches a range of addresses from this trigger address to the trigger address + 15, then the AHB request will be completed by fetching data from the Indirect Controllers SRAM.

Table 11-3242. Register Call Summary for QSPI_IND_AHB_ADDR_TRIGGER_REG

QSPI Functional Description • Indirect write transfer process: [0] • Indirect Read Controller: [0] • Indirect Write Controller: [0] • Indirect read transfer process: [0]
QSPI Registers • QSPI_IND_AHB_ADDR_TRIGGER_REG Register (Offset = 1Ch) [reset = 0h]: [0] • QSPI Registers: [0]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0]

11.15.6.9 QSPI_REMAP_ADDR_REG Register (Offset = 24h) [reset = 0h]

QSPI_REMAP_ADDR_REG is shown in Figure 11-1319 and described in Table 11-3244.

Table 11-3243. QSPI_REMAP_ADDR_REG Instances

Instance	Physical Address
QSPI	0294 0024h

Figure 11-1319. QSPI_REMAP_ADDR_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE_FLD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3244. QSPI_REMAP_ADDR_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VALUE_FLD	R/W	0h	Remap Address Register: This register is used to remap an incoming AHB address to a different address used by the FLASH device.

Table 11-3245. Register Call Summary for QSPI_REMAP_ADDR_REG

QSPI Functional Description • Data Interface Address Remapping: [0]
QSPI Registers • QSPI_REMAP_ADDR_REG Register (Offset = 24h) [reset = 0h]: [0] • QSPI Registers: [0] • QSPI_CONFIG_REG Register (Offset = 0h) [reset = 80780081h]: [0]
QSPI Programming Guide • Direct Access Controller Programming Sequence: [0]

11.15.6.10 QSPI_MODE_BIT_CONFIG_REG Register (Offset = 28h) [reset = 0h]

QSPI_MODE_BIT_CONFIG_REG is shown in Figure 11-1320 and described in Table 11-3247.

Table 11-3246. QSPI_MODE_BIT_CONFIG_REG Instances

Instance	Physical Address
QSPI	0294 0028h

Figure 11-1320. QSPI_MODE_BIT_CONFIG_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								MODE_FLD							
R-0h																								R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3247. QSPI_MODE_BIT_CONFIG_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	MODE_FLD	R/W	0h	Mode Bits: These are the 8 mode bits that are sent to the device following the address bytes if mode bit transmission has been enabled via QSPI_FLASH_CMD_CTRL_REG[8] ENB_MODE_BIT_FLD bit.

Table 11-3248. Register Call Summary for QSPI_MODE_BIT_CONFIG_REG

QSPI Registers
- QSPI_MODE_BIT_CONFIG_REG Register (Offset = 28h) [reset = 0h]: [0]
- QSPI_FLASH_CMD_CTRL_REG Register (Offset = 90h) [reset = 0h]: [0]
- QSPI Registers: [0]
- QSPI_DEV_INSTR_RD_CONFIG_REG Register (Offset = 4h) [reset = 3h]: [0]

11.15.6.11 QSPI_SRAM_FILL_REG Register (Offset = 2Ch) [reset = 0h]

QSPI_SRAM_FILL_REG is shown in Figure 11-1321 and described in Table 11-3250.

Table 11-3249. QSPI_SRAM_FILL_REG Instances

Instance	Physical Address
QSPI	0294 002Ch

Figure 11-1321. QSPI_SRAM_FILL_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRAM_FILL_INDAC_WRITE_FLD															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRAM_FILL_INDAC_READ_FLD															
R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3250. QSPI_SRAM_FILL_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	SRAM_FILL_INDAC_WRITE_FLD	R	0h	SRAM Fill Level (Indirect Write Partition): Identifies the current fill level of the SRAM Indirect Write partition.
15-0	SRAM_FILL_INDAC_READ_FLD	R	0h	SRAM Fill Level (Indirect Read Partition): Identifies the current fill level of the SRAM Indirect Read partition.

Table 11-3251. Register Call Summary for QSPI_SRAM_FILL_REG

QSPI Functional Description • Indirect write transfer process: [0] • Accessing the SRAM: [0] • Indirect Read Controller: [0] • Indirect Write Controller: [0] • Indirect read transfer process: [0]
QSPI Registers • QSPI Registers: [0] • QSPI_SRAM_FILL_REG Register (Offset = 2Ch) [reset = 0h]: [0]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0]

11.15.6.12 QSPI_TX_THRESH_REG Register (Offset = 30h) [reset = 1h]

QSPI_TX_THRESH_REG is shown in Figure 11-1322 and described in Table 11-3253.

Table 11-3252. QSPI_TX_THRESH_REG Instances

Instance	Physical Address
QSPI	0294 0030h

Figure 11-1322. QSPI_TX_THRESH_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												LEVEL_FLD			
R-0h												R/W-1h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3253. QSPI_TX_THRESH_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-0	LEVEL_FLD	R/W	1h	Level TX FIFO not full: Defines the level at which the small TX FIFO not full interrupt is generated.

Table 11-3254. Register Call Summary for QSPI_TX_THRESH_REG

QSPI Registers
- QSPI_TX_THRESH_REG Register (Offset = 30h) [reset = 1h]: [0]
- QSPI Registers: [0]

11.15.6.13 QSPI_RX_THRESH_REG Register (Offset = 34h) [reset = 1h]

QSPI_RX_THRESH_REG is shown in Figure 11-1323 and described in Table 11-3256.

QSPI Instruction Configuration Register

Table 11-3255. QSPI_RX_THRESH_REG Instances

Instance	Physical Address
QSPI	0294 0034h

Figure 11-1323. QSPI_RX_THRESH_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												LEVEL_FLD			
R-0h												R/W-1h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3256. QSPI_RX_THRESH_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-0	LEVEL_FLD	R/W	1h	Level RX FIFO not empty: Defines the level at which the small RX FIFO not empty interrupt is generated.

Table 11-3257. Register Call Summary for QSPI_RX_THRESH_REG

QSPI Registers
- QSPI Registers: [0]
- QSPI_RX_THRESH_REG Register (Offset = 34h) [reset = 1h]: [0]

11.15.6.14 QSPI_WRITE_COMPLETION_CTRL_REG Register (Offset = 38h) [reset = 10005h]

QSPI_WRITE_COMPLETION_CTRL_REG is shown in Figure 11-1324 and described in Table 11-3259.

This register defines how the controller will poll the device following a write transfer.

Table 11-3258. QSPI_WRITE_COMPLETION_CTRL_REG Instances

Instance	Physical Address
QSPI	0294 0038h

Figure 11-1324. QSPI_WRITE_COMPLETION_CTRL_REG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
POLL_COUNT_FLD							
R/W-1h							
15	14	13	12	11	10	9	8
WR_COMP_CTRL_RESV1_FLD	DISABLE_POLLING_FLD	POLLING_POLARITY_FLD	RESERVED		POLLING_BIT_INDEX_FLD		
R-0h	R/W-0h	R/W-0h	R-0h		R/W-0h		
7	6	5	4	3	2	1	0
OPCODE_FLD							
R/W-5h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3259. QSPI_WRITE_COMPLETION_CTRL_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	POLL_COUNT_FLD	R/W	1h	Polling Count: Defines the number of times the controller should expect to see a true result from the polling in successive reads of the device register.
15	RESERVED	R	0h	Reserved
14	DISABLE_POLLING_FLD	R/W	0h	Disable Polling: This switches off the automatic polling function.
13	POLLING_POLARITY_FLD	R/W	0h	Polling Polarity: Defines the polling polarity. If '1', then the write transfer to the device will be complete if the polled bit is equal to '1'. If '0', then the write transfer to the device will be complete if the polled bit is equal to '0'.
12-11	RESERVED	R	0h	Reserved
10-8	POLLING_BIT_INDEX_FLD	R/W	0h	Polling Bit Index: Defines the bit index that should be polled. A value of 2h means that bit 2 of the returned data will be polled for. A value of 7h means that bit 7 of the returned data will be polled for.
7-0	OPCODE_FLD	R/W	5h	Opcode: Defines the opcode that should be issued by the controller when it is automatically polling for device program completion. This command is issued followed all device write operations. By default, this will poll the standard device STATUS register using opcode 05h.

Table 11-3260. Register Call Summary for QSPI_WRITE_COMPLETION_CTRL_REG

QSPI Functional Description • Servicing a STIG request: [0]
QSPI Registers • QSPI_WRITE_COMPLETION_CTRL_REG Register (Offset = 38h) [reset = 10005h]: [0] • QSPI Registers: [0]
QSPI Programming Guide • Write Completion Polling for both DAC and INDAC: [0][1][2][3][4][5][6][7][8][9][10][11]

11.15.6.15 QSPI_NO_OF_POLLS_BEF_EXP_REG Register (Offset = 3Ch) [reset = FFFFFFFFh]

QSPI_NO_OF_POLLS_BEF_EXP_REG is shown in Figure 11-1325 and described in Table 11-3262.

Table 11-3261. QSPI_NO_OF_POLLS_BEF_EXP_REG Instances

Instance	Physical Address
QSPI	0294 003Ch

Figure 11-1325. QSPI_NO_OF_POLLS_BEF_EXP_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NO_OF_POLLS_BEF_EXP_FLD																															
R/W-FFFFFFFFh																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3262. QSPI_NO_OF_POLLS_BEF_EXP_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NO_OF_POLLS_BEF_EXP_FLD	R/W	FFFFFFFFh	Defines the numbers of polls cycles after which polling expiration interrupt is generated. NOTE: Max no of polls > 2(TX_FIFO_DEPTH – 1) + 1. Polling cycles are queued in FIFO, so expiration requires to make FIFO empty.

Table 11-3263. Register Call Summary for QSPI_NO_OF_POLLS_BEF_EXP_REG

QSPI Registers
- QSPI Registers: [0]
- QSPI_NO_OF_POLLS_BEF_EXP_REG Register (Offset = 3Ch) [reset = FFFFFFFFh]: [0]

11.15.6.16 QSPI_IRQ_STATUS_REG Register (Offset = 40h) [reset = 100h]

QSPI_IRQ_STATUS_REG is shown in Figure 11-1326 and described in Table 11-3265.

The status fields in this register are set when the described event occurs and the interrupt is enabled in the QSPI_IRQ_MASK_REG register. When any of these bit fields are set, the interrupt output is asserted high. The fields are each cleared by writing a 1 to the field. Note that bit fields 6 thru 10 are only valid when legacy SPI mode is active.

Table 11-3264. QSPI_IRQ_STATUS_REG Instances

Instance	Physical Address
QSPI	0294 0040h

Figure 11-1326. QSPI_IRQ_STATUS_REG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			INDRD_SRAM_FULL_FLD	RX_FIFO_FULL_FLD	RX_FIFO_NOT_EMPTY_FLD	TX_FIFO_FULL_FLD	TX_FIFO_NOT_FULL_FLD
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h
7	6	5	4	3	2	1	0
RECV_OVERFLOW_FLD	INDIRECT_XFER_LEVEL_BREACH_FLD	ILLEGAL_ACCESS_DET_FLD	PROT_WR_ATTEMPT_FLD	INDIRECT_READ_REJECT_FLD	INDIRECT_OP_DONE_FLD	UNDERFLOW_DET_FLD	MODE_M_FAIL_FLD
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3265. QSPI_IRQ_STATUS_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12	INDRD_SRAM_FULL_FLD	R/W	0h	Indirect Read Partition overflow: Indirect Read Partition of SRAM is full and unable to immediately complete indirect operation.
11	RX_FIFO_FULL_FLD	R/W	0h	Small RX FIFO full: Current FIFO status can be ignored in non-SPI legacy mode. 0h: FIFO is not full. 1h: FIFO is full.
10	RX_FIFO_NOT_EMPTY_FLD	R/W	0h	Small RX FIFO not empty: Current FIFO status can be ignored in non-SPI legacy mode. 0h: FIFO has less than RX THRESHOLD entries. 1h: FIFO has >= THRESHOLD entries.
9	TX_FIFO_FULL_FLD	R/W	0h	Small TX FIFO full: Current FIFO status can be ignored in non-SPI legacy mode. 0h: FIFO is not full. 1h: FIFO is full.

Table 11-3265. QSPI_IRQ_STATUS_REG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	TX_FIFO_NOT_FULL_FLD	R/W	1h	Small TX FIFO not full: Current FIFO status can be ignored in non-SPI legacy mode. 0h: FIFO has >= THRESHOLD entries. 1h: FIFO has less than THRESHOLD entries.
7	RECV_OVERFLOW_FLD	R/W	0h	Receive Overflow: This should only occur in Legacy SPI mode. Set if an attempt is made to push the RX FIFO when it is full. This bit is reset only by a system reset and cleared only when this register is read. If a new push to the RX FIFO occurs coincident with a register read this flag will remain set. 0h: no overflow has been detected. 1h: an overflow has occurred.
6	INDIRECT_XFER_LEVEL_BREACH_FLD	R/W	0h	Transfer Watermark Breach: Indirect Transfer Watermark Level Breached.
5	ILLEGAL_ACCESS_DET_FLD	R/W	0h	Illegal AHB Access Detected: AHB wrapping bursts and the use of SPLIT/RETRY accesses will cause this error interrupt to trigger.
4	PROT_WR_ATTEMPT_FLD	R/W	0h	Protected Area Write Attempt: Write to protected area was attempted and rejected.
3	INDIRECT_READ_REJECT_FLD	R/W	0h	Indirect Read Reject: Indirect operation was requested but could not be accepted. Two indirect operations already in storage.
2	INDIRECT_OP_DONE_FLD	R/W	0h	Indirect Operation Complete: Controller has completed last triggered indirect operation.
1	UNDERFLOW_DET_FLD	R/W	0h	Underflow Detected: 0h: no underflow has been detected. 1h: underflow is detected and an attempt to transfer data is made when the small TX FIFO is empty. This may occur when AHB write data is being supplied too slowly to keep up with the requested write operation This bit is reset only by a system reset and cleared only when the register is read.
0	MODE_M_FAIL_FLD	R/W	0h	Mode M Failure: Mode M failure indicates the voltage on pin n_ss_in is inconsistent with the SPI mode. Set =1 if n_ss_in is low in master mode (multi-master contention). These conditions will clear the QSPI_CONFIG_REG[0] ENB_QSPI_FLD bit and disable the SPI. This bit is reset only by a system reset and cleared only when this register is read. 0h: no mode fault has been detected. 1h: a mode fault has occurred.

Table 11-3266. Register Call Summary for QSPI_IRQ_STATUS_REG

QSPI Functional Description • QSPI Interrupt Requests: [0][1][2][3][4][5][6][7][8][9][10][11][12]
QSPI Registers • QSPI_IRQ_STATUS_REG Register (Offset = 40h) [reset = 100h]: [0] • QSPI Registers: [0]

11.15.6.17 QSPI_IRQ_MASK_REG Register (Offset = 44h) [reset = 0h]

QSPI_IRQ_MASK_REG is shown in Figure 11-1327 and described in Table 11-3268.

0h: the interrupt for the corresponding interrupt status register bit is disabled.
1h: the interrupt for the corresponding interrupt status register bit is enabled.

Table 11-3267. QSPI_IRQ_MASK_REG Instances

Instance	Physical Address
QSPI	0294 0044h

Figure 11-1327. QSPI_IRQ_MASK_REG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			INDRD_SRAM_FULL_MASK_FLD	RX_FIFO_FULL_MASK_FLD	RX_FIFO_NOT_EMPTY_MASK_FLD	TX_FIFO_FULL_MASK_FLD	TX_FIFO_NOT_FULL_MASK_FLD
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RECV_OVERFLOW_MASK_FLD	INDIRECT_XFER_LEVEL_BREACH_MASK_FLD	ILLEGAL_ACCESS_DET_MASK_FLD	PROT_WR_ATTEMPT_MASK_FLD	INDIRECT_READ_REJECT_MASK_FLD	INDIRECT_OP_DONE_MASK_FLD	UNDERFLOW_DET_MASK_FLD	MODE_M_FAIL_MASK_FLD
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3268. QSPI_IRQ_MASK_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12	INDRD_SRAM_FULL_MASK_FLD	R/W	0h	Indirect Read Partition overflow mask
11	RX_FIFO_FULL_MASK_FLD	R/W	0h	Small RX FIFO full Mask
10	RX_FIFO_NOT_EMPTY_MASK_FLD	R/W	0h	Small RX FIFO not empty Mask
9	TX_FIFO_FULL_MASK_FLD	R/W	0h	Small TX FIFO full Mask
8	TX_FIFO_NOT_FULL_MASK_FLD	R/W	0h	Small TX FIFO not full Mask
7	RECV_OVERFLOW_MASK_FLD	R/W	0h	Receive Overflow Mask
6	INDIRECT_XFER_LEVEL_BREACH_MASK_FLD	R/W	0h	Transfer Watermark Breach Mask
5	ILLEGAL_ACCESS_DET_MASK_FLD	R/W	0h	Illegal Access Detected Mask
4	PROT_WR_ATTEMPT_MASK_FLD	R/W	0h	Protected Area Write Attempt Mask
3	INDIRECT_READ_REJECT_MASK_FLD	R/W	0h	Indirect Read Reject Mask
2	INDIRECT_OP_DONE_MASK_FLD	R/W	0h	Indirect Complete Mask

Table 11-3268. QSPI_IRQ_MASK_REG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	UNDERFLOW_DET_MASK_FLD	R/W	0h	Underflow Detected Mask
0	MODE_M_FAIL_MASK_FLD	R/W	0h	Mode M Failure Mask

Table 11-3269. Register Call Summary for QSPI_IRQ_MASK_REG

QSPI Functional Description • QSPI Interrupt Requests: [0][1][2][3][4][5][6][7][8][9][10][11][12]
QSPI Registers • QSPI_IRQ_STATUS_REG Register (Offset = 40h) [reset = 100h]: [0] • QSPI_IRQ_MASK_REG Register (Offset = 44h) [reset = 0h]: [0] • QSPI Registers: [0]

11.15.6.18 QSPI_LOWER_WR_PROT_REG Register (Offset = 50h) [reset = 0h]

QSPI_LOWER_WR_PROT_REG is shown in Figure 11-1328 and described in Table 11-3271.

Table 11-3270. QSPI_LOWER_WR_PROT_REG Instances

Instance	Physical Address
QSPI	0294 0050h

Figure 11-1328. QSPI_LOWER_WR_PROT_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SUBSECTOR_FLD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3271. QSPI_LOWER_WR_PROT_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SUBSECTOR_FLD	R/W	0h	Lower Block Number: The block number that defines the lower block in the range of blocks that is to be locked from writing. The definition of a block in terms of number of bytes is programmable via the QSPI_DEV_SIZE_CONFIG_REG register.

Table 11-3272. Register Call Summary for QSPI_LOWER_WR_PROT_REG

QSPI Functional Description • Write Protection: [0]
QSPI Registers • QSPI_LOWER_WR_PROT_REG Register (Offset = 50h) [reset = 0h]: [0] • QSPI Registers: [0]

11.15.6.19 QSPI_UPPER_WR_PROT_REG Register (Offset = 54h) [reset = 0h]

QSPI_UPPER_WR_PROT_REG is shown in Figure 11-1329 and described in Table 11-3274.

Table 11-3273. QSPI_UPPER_WR_PROT_REG Instances

Instance	Physical Address
QSPI	0294 0054h

Figure 11-1329. QSPI_UPPER_WR_PROT_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SUBSECTOR_FLD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3274. QSPI_UPPER_WR_PROT_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SUBSECTOR_FLD	R/W	0h	Upper Block Number: The block number that defines the upper block in the range of blocks that is to be locked from writing. The definition of a block in terms of number of bytes is programmable via the QSPI_DEV_SIZE_CONFIG_REG register.

Table 11-3275. Register Call Summary for QSPI_UPPER_WR_PROT_REG

QSPI Functional Description • Write Protection: [0]
QSPI Registers • QSPI_UPPER_WR_PROT_REG Register (Offset = 54h) [reset = 0h]: [0] • QSPI Registers: [0]

11.15.6.20 QSPI_WR_PROT_CTRL_REG Register (Offset = 58h) [reset = 0h]

QSPI_WR_PROT_CTRL_REG is shown in Figure 11-1330 and described in Table 11-3277.

Table 11-3276. QSPI_WR_PROT_CTRL_REG Instances

Instance	Physical Address
QSPI	0294 0058h

Figure 11-1330. QSPI_WR_PROT_CTRL_REG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ENB_FLD	INV_FLD
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3277. QSPI_WR_PROT_CTRL_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	ENB_FLD	R/W	0h	Write Protection Enable Bit: When set to 1, any AHB write access with an address within the protection region defined in the lower and upper write protection registers is rejected. An AHB error response is generated and an interrupt source triggered. When set to 0, the protection region is disabled.
0	INV_FLD	R/W	0h	Write Protection Inversion Bit: When set to 1, the protection region defined in the lower and upper write protection registers is inverted meaning it is the region that the system is permitted to write to. When set to 0, the protection region defined in the lower and upper write protection registers is the region that the system is not permitted to write to.

Table 11-3278. Register Call Summary for QSPI_WR_PROT_CTRL_REG

QSPI Functional Description • Write Protection: [0][1][2]
QSPI Registers • QSPI Registers: [0] • QSPI_WR_PROT_CTRL_REG Register (Offset = 58h) [reset = 0h]: [0]

11.15.6.21 QSPI_INDIRECT_READ_XFER_CTRL_REG Register (Offset = 60h) [reset = 0h]

QSPI_INDIRECT_READ_XFER_CTRL_REG is shown in Figure 11-1331 and described in Table 11-3280.

Table 11-3279. QSPI_INDIRECT_READ_XFER_CTRL_REG Instances

Instance	Physical Address
QSPI	0294 0060h

Figure 11-1331. QSPI_INDIRECT_READ_XFER_CTRL_REG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
NUM_IND_OPS_DONE_FLD		IND_OPS_DONE_STATUS_FLD	RD_QUEUED_FLD	SRAM_FULL_FLD	RD_STATUS_FLD	CANCEL_FLD	START_FLD
R-0h		R/W-0h	R-0h	R/W-0h	R-0h	W-0h	W-0h

LEGEND: R = Read Only; R/W = Read/Write; W = Write Only; -*n* = value after reset

Table 11-3280. QSPI_INDIRECT_READ_XFER_CTRL_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	NUM_IND_OPS_DONE_FLD	R	0h	Completed Indirect Operations: This field contains the number of indirect operations which have been completed. This is used in conjunction with the QSPI_INDIRECT_READ_XFER_CTRL_REG[5] IND_OPS_DONE_STATUS_FLD bit. It is incremented by hardware when an indirect operation has completed. Write a 1 to QSPI_INDIRECT_READ_XFER_CTRL_REG[5] IND_OPS_DONE_STATUS_FLD bit to decrement it.
5	IND_OPS_DONE_STATUS_FLD	R/W	0h	Indirect Completion Status: This field is set to 1 when an indirect operation has completed. Write a 1 to this field to clear it.
4	RD_QUEUED_FLD	R	0h	Queued Indirect Read Operations: Two indirect read operations have been queued.
3	SRAM_FULL_FLD	R/W	0h	SRAM Full: SRAM full and unable to immediately complete an indirect operation. Write a 1 to this field to clear it.
2	RD_STATUS_FLD	R	0h	Indirect Read Status: Indirect read operation in progress (status).
1	CANCEL_FLD	W	0h	Cancel Indirect Read: Writing a 1 to this bit will cancel all ongoing indirect read operations.
0	START_FLD	W	0h	Start Indirect Read: Writing a 1 to this bit will trigger an indirect read operation. The assumption is that the indirect start address and the indirect number of bytes register is set-up before triggering the indirect read operation.

Table 11-3281. Register Call Summary for QSPI_INDIRECT_READ_XFER_CTRL_REG

QSPI Functional Description • Indirect Access Queuing: [0] • Indirect Read Controller: [0][1][2][3] • Indirect read transfer process: [0][1]
QSPI Registers • QSPI_INDIRECT_READ_XFER_CTRL_REG Register (Offset = 60h) [reset = 0h]: [0][1][2] • QSPI Registers: [0]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0][1][2]

11.15.6.22 QSPI_INDIRECT_READ_XFER_WATERMARK_REG Register (Offset = 64h) [reset = 0h]

QSPI_INDIRECT_READ_XFER_WATERMARK_REG is shown in Figure 11-1332 and described in Table 11-3283.

Table 11-3282. QSPI_INDIRECT_READ_XFER_WATERMARK_REG Instances

Instance	Physical Address
QSPI	0294 0064h

Figure 11-1332. QSPI_INDIRECT_READ_XFER_WATERMARK_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LEVEL_FLD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3283. QSPI_INDIRECT_READ_XFER_WATERMARK_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LEVEL_FLD	R/W	0h	Watermark Value: This represents the minimum fill level of the SRAM. When the SRAM fill level passes the watermark, an interrupt is also generated. This field can be disabled by writing a value of all zeroes.

Table 11-3284. Register Call Summary for QSPI_INDIRECT_READ_XFER_WATERMARK_REG

QSPI Functional Description • Indirect Read Controller: [0][1] • Indirect read transfer process: [0]
QSPI Registers • QSPI_INDIRECT_READ_XFER_WATERMARK_REG Register (Offset = 64h) [reset = 0h]: [0] • QSPI Registers: [0]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0]

11.15.6.23 QSPI_INDIRECT_READ_XFER_START_REG Register (Offset = 68h) [reset = 0h]

QSPI_INDIRECT_READ_XFER_START_REG is shown in Figure 11-1333 and described in Table 11-3286.

Table 11-3285. QSPI_INDIRECT_READ_XFER_START_REG Instances

Instance	Physical Address
QSPI	0294 0068h

Figure 11-1333. QSPI_INDIRECT_READ_XFER_START_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR_FLD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3286. QSPI_INDIRECT_READ_XFER_START_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR_FLD	R/W	0h	Start of Indirect Access: This is the start address from which the indirect access will commence its READ operation.

Table 11-3287. Register Call Summary for QSPI_INDIRECT_READ_XFER_START_REG

QSPI Functional Description • Indirect Access Queuing: [0][1] • Indirect Read Controller: [0][1][2] • Indirect read transfer process: [0]
QSPI Registers • QSPI_INDIRECT_READ_XFER_START_REG Register (Offset = 68h) [reset = 0h]: [0] • QSPI Registers: [0]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0]

11.15.6.24 QSPI_INDIRECT_READ_XFER_NUM_BYTES_REG Register (Offset = 6Ch) [reset = 0h]

QSPI_INDIRECT_READ_XFER_NUM_BYTES_REG is shown in Figure 11-1334 and described in Table 11-3289.

Table 11-3288. QSPI_INDIRECT_READ_XFER_NUM_BYTES_REG Instances

Instance	Physical Address
QSPI	0294 006Ch

Figure 11-1334. QSPI_INDIRECT_READ_XFER_NUM_BYTES_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE_FLD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3289. QSPI_INDIRECT_READ_XFER_NUM_BYTES_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VALUE_FLD	R/W	0h	Indirect Number of Bytes: This is the number of bytes that the indirect access will consume. This can be bigger than the configured size of SRAM.

Table 11-3290. Register Call Summary for QSPI_INDIRECT_READ_XFER_NUM_BYTES_REG

QSPI Functional Description • Indirect Access Queuing: [0][1] • Indirect Read Controller: [0][1] • Indirect read transfer process: [0]
QSPI Registers • QSPI_INDIRECT_READ_XFER_NUM_BYTES_REG Register (Offset = 6Ch) [reset = 0h]: [0] • QSPI Registers: [0]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0]

11.15.6.25 QSPI_INDIRECT_WRITE_XFER_CTRL_REG Register (Offset = 70h) [reset = 0h]

QSPI_INDIRECT_WRITE_XFER_CTRL_REG is shown in Figure 11-1335 and described in Table 11-3292.

Table 11-3291. QSPI_INDIRECT_WRITE_XFER_CTRL_REG Instances

Instance	Physical Address
QSPI	0294 0070h

Figure 11-1335. QSPI_INDIRECT_WRITE_XFER_CTRL_REG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
NUM_IND_OPS_DONE_FLD		IND_OPS_DONE_STATUS_FLD	WR_QUEUED_FLD	INDIR_WR_XFER_RESV1_FLD	WR_STATUS_FLD	CANCEL_FLD	START_FLD
R-0h		R/W-0h	R-0h	R-0h	R-0h	W-0h	W-0h

LEGEND: R = Read Only; R/W = Read/Write; W = Write Only; -*n* = value after reset

Table 11-3292. QSPI_INDIRECT_WRITE_XFER_CTRL_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	NUM_IND_OPS_DONE_FLD	R	0h	Completed Indirect Operations: This field contains the number of indirect operations which have been completed. This is used in conjunction with the the QSPI_INDIRECT_WRITE_XFER_CTRL_REG[5] IND_OPS_DONE_STATUS_FLD bit. It is incremented by hardware when an indirect operation has completed. Write a 1 to the the QSPI_INDIRECT_WRITE_XFER_CTRL_REG[5] IND_OPS_DONE_STATUS_FLD bit to decrement it.
5	IND_OPS_DONE_STATUS_FLD	R/W	0h	Indirect Completion Status: This field is set to 1 when an indirect operation has completed. Write a 1 to this field to clear it.
4	WR_QUEUED_FLD	R	0h	Queued Indirect Write Operations: Two indirect write operations have been queued.
3	RESERVED	R	0h	Reserved
2	WR_STATUS_FLD	R	0h	Indirect Write Status: Indirect write operation in progress (status).
1	CANCEL_FLD	W	0h	Cancel Indirect Write: Writing a 1 to this bit will cancel all ongoing indirect write operations.
0	START_FLD	W	0h	Start Indirect Write: Writing a 1 to this bit will trigger an indirect write operation. The assumption is that the indirect start address and the indirect number of bytes register is set-up before triggering the indirect write operation.

Table 11-3293. Register Call Summary for QSPI_INDIRECT_WRITE_XFER_CTRL_REG

QSPI Functional Description • Indirect Access Queuing: [0] • Indirect write transfer process: [0][1] • Indirect Write Controller: [0][1][2][3]
QSPI Registers • QSPI_INDIRECT_WRITE_XFER_CTRL_REG Register (Offset = 70h) [reset = 0h]: [0][1][2] • QSPI Registers: [0]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0][1][2]

11.15.6.26 QSPI_INDIRECT_WRITE_XFER_WATERMARK_REG Register (Offset = 74h) [reset = FFFFFFFFh]

QSPI_INDIRECT_WRITE_XFER_WATERMARK_REG is shown in Figure 11-1336 and described in Table 11-3295.

Table 11-3294. QSPI_INDIRECT_WRITE_XFER_WATERMARK_REG Instances

Instance	Physical Address
QSPI	0294 0074h

Figure 11-1336. QSPI_INDIRECT_WRITE_XFER_WATERMARK_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LEVEL_FLD																															
R/W-FFFFFFFFh																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3295. QSPI_INDIRECT_WRITE_XFER_WATERMARK_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LEVEL_FLD	R/W	FFFFFFFFh	Watermark Value: This represents the maximum fill level of the SRAM. When the SRAM fill level falls below the watermark, an interrupt is also generated. This field can be disabled by writing a value of all ones.

Table 11-3296. Register Call Summary for QSPI_INDIRECT_WRITE_XFER_WATERMARK_REG

QSPI Functional Description • Indirect write transfer process: [0] • Indirect Write Controller: [0][1]
QSPI Registers • QSPI_INDIRECT_WRITE_XFER_WATERMARK_REG Register (Offset = 74h) [reset = FFFFFFFFh]: [0] • QSPI Registers: [0]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0]

11.15.6.27 QSPI_INDIRECT_WRITE_XFER_START_REG Register (Offset = 78h) [reset = 0h]

QSPI_INDIRECT_WRITE_XFER_START_REG is shown in Figure 11-1337 and described in Table 11-3298.

Table 11-3297. QSPI_INDIRECT_WRITE_XFER_START_REG Instances

Instance	Physical Address
QSPI	0294 0078h

Figure 11-1337. QSPI_INDIRECT_WRITE_XFER_START_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR_FLD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3298. QSPI_INDIRECT_WRITE_XFER_START_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR_FLD	R/W	0h	Start of Indirect Access: This is the start address from which the indirect access will commence its WRITE operation.

Table 11-3299. Register Call Summary for QSPI_INDIRECT_WRITE_XFER_START_REG

QSPI Functional Description • Indirect Access Queuing: [0] • Indirect write transfer process: [0] • Indirect Write Controller: [0][1][2]
QSPI Registers • QSPI Registers: [0] • QSPI_INDIRECT_WRITE_XFER_START_REG Register (Offset = 78h) [reset = 0h]: [0]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0]

11.15.6.28 QSPI_INDIRECT_WRITE_XFER_NUM_BYTES_REG Register (Offset = 7Ch) [reset = 0h]

QSPI_INDIRECT_WRITE_XFER_NUM_BYTES_REG is shown in Figure 11-1338 and described in Table 11-3301.

Table 11-3300. QSPI_INDIRECT_WRITE_XFER_NUM_BYTES_REG Instances

Instance	Physical Address
QSPI	0294 007Ch

Figure 11-1338. QSPI_INDIRECT_WRITE_XFER_NUM_BYTES_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE_FLD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3301. QSPI_INDIRECT_WRITE_XFER_NUM_BYTES_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VALUE_FLD	R/W	0h	Indirect Number of Bytes: This is the number of bytes that the indirect access will consume. This can be bigger than the configured size of SRAM.

Table 11-3302. Register Call Summary for QSPI_INDIRECT_WRITE_XFER_NUM_BYTES_REG

QSPI Functional Description • Indirect Access Queuing: [0] • Indirect write transfer process: [0] • Indirect Write Controller: [0][1]
QSPI Registers • QSPI_INDIRECT_WRITE_XFER_NUM_BYTES_REG Register (Offset = 7Ch) [reset = 0h]: [0] • QSPI Registers: [0]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0]

11.15.6.29 QSPI_FLASH_CMD_CTRL_REG Register (Offset = 90h) [reset = 0h]

QSPI_FLASH_CMD_CTRL_REG is shown in Figure 11-1339 and described in Table 11-3304.

Table 11-3303. QSPI_FLASH_CMD_CTRL_REG Instances

Instance	Physical Address
QSPI	0294 0090h

Figure 11-1339. QSPI_FLASH_CMD_CTRL_REG Register

31	30	29	28	27	26	25	24
CMD_OPCODE_FLD							
R/W-0h							
23	22	21	20	19	18	17	16
ENB_READ_DATA_FLD	NUM_RD_DATA_BYTES_FLD			ENB_COMD_ADDR_FLD	ENB_MODE_BIT_FLD	NUM_ADDR_BYTES_FLD	
R/W-0h	R/W-0h			R/W-0h	R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
ENB_WRITE_DATA_FLD	NUM_WR_DATA_BYTES_FLD			NUM_DUMMY_BYTES_FLD			
R/W-0h	R/W-0h			R/W-0h			
7	6	5	4	3	2	1	0
NUM_DUMMY_BYTES_FLD	RESERVED					CMD_EXEC_STATUS_FLD	CMD_EXEC_FLD
R/W-0h	R-0h					R-0h	W-0h

LEGEND: R = Read Only; R/W = Read/Write; W = Write Only; -*n* = value after reset

Table 11-3304. QSPI_FLASH_CMD_CTRL_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	CMD_OPCODE_FLD	R/W	0h	Command Opcode: The command opcode field should be set-up before triggering the command. For example, 20h maps to SubSector Erase. Writing to the QSPI_FLASH_CMD_CTRL_REG[0] CMD_EXEC_FLD bit launches the command. NOTE: Using this approach to issue commands to an external flash device makes use of the instruction type of the device instruction configuration register. ·If this field is set to 0h, then the command opcode, command address, command dummy bytes and command data will all be transferred in a serial fashion. ·If this field is set to 1h, then the command opcode, command address, command dummy bytes and command data will all be transferred in parallel using QSPI_D0 and QSPI_D1 pins. ·If this field is set to 2h, then the command opcode, command address, command dummy bytes and command data will all be transferred in parallel using QSPI_D0, QSPI_D1, QSPI_D2 and QSPI_D3 pins.
23	ENB_READ_DATA_FLD	R/W	0h	Read Data Enable: Set to 1 if the command specified in the QSPI_FLASH_CMD_CTRL_REG[31-24] CMD_OPCODE_FLD field requires read data bytes to be received from the external flash device.
22-20	NUM_RD_DATA_BYTES_FLD	R/W	0h	Number of Read Data Bytes: Up to 8 data bytes may be read using this command. Set to 0 for 1 byte and 7 for 8 bytes.

Table 11-3304. QSPI_FLASH_CMD_CTRL_REG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	ENB_COMD_ADDR_FLD	R/W	0h	Command Address Enable: Set to 1 if the QSPI_FLASH_CMD_CTRL_REG[31-24] CMD_OPCODE_FLD field requires an address. This should be set-up before triggering the command via writing a 1 to the QSPI_FLASH_CMD_CTRL_REG[0] CMD_EXEC_FLD bit.
18	ENB_MODE_BIT_FLD	R/W	0h	Mode Bit Enable: Set to 1 to ensure the mode bits defined in the QSPI_MODE_BIT_CONFIG_REG[7-0] MODE_FLD field are sent following the address bytes.
17-16	NUM_ADDR_BYTES_FLD	R/W	0h	Number of Address Bytes: Set to the number of address bytes required (the address itself is programmed in the QSPI_FLASH_CMD_ADDR_REG register). This should be set-up before triggering the command via QSPI_FLASH_CMD_CTRL_REG[0] CMD_EXEC_FLD bit of this register. 0h: 1 address byte, 1h: 2 address bytes, 2h: 3 address bytes, 3h: 4 address bytes.
15	ENB_WRITE_DATA_FLD	R/W	0h	Write Data Enable: Set to 1 if the command specified in the command opcode field (CMD_OPCODE_FLD) of this register requires write data bytes to be sent to the external flash device.
14-12	NUM_WR_DATA_BYTES_FLD	R/W	0h	Number of Write Data Bytes: Up to 8 Data bytes may be written using this command. Set to 0 for 1 byte, 7 for 8 bytes.
11-7	NUM_DUMMY_BYTES_FLD	R/W	0h	Number of Dummy Bytes: Set to the number of dummy bytes required. This should be set-up before triggering the command via the QSPI_FLASH_CMD_CTRL_REG[0] CMD_EXEC_FLD bit of this register.
6-2	RESERVED	R	0h	Reserved
1	CMD_EXEC_STATUS_FLD	R	0h	Command Execution Status: 1h: Command execution in progress.
0	CMD_EXEC_FLD	W	0h	Execute Command: 1h: Execute the command.

Table 11-3305. Register Call Summary for QSPI_FLASH_CMD_CTRL_REG

QSPI Functional Description • Servicing a STIG request: [0][1][2][3][4] • Software Triggered Instruction Generator (STIG): [0][1][2][3] • Configuration Slave Port: [0]
QSPI Registers • QSPI_FLASH_WR_DATA_LOWER_REG Register (Offset = A8h) [reset = 0h]: [0][1] • QSPI_FLASH_CMD_ADDR_REG Register (Offset = 94h) [reset = 0h]: [0][1] • QSPI_MODE_BIT_CONFIG_REG Register (Offset = 28h) [reset = 0h]: [0] • QSPI_FLASH_WR_DATA_UPPER_REG Register (Offset = ACh) [reset = 0h]: [0][1] • QSPI Registers: [0] • QSPI_FLASH_CMD_CTRL_REG Register (Offset = 90h) [reset = 0h]: [0][1][2][3][4][5][6]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0][1][2][3][4][5][6][7][8] • Direct Access Controller Programming Sequence: [0][1][2][3][4][5][6][7]

11.15.6.30 QSPI_FLASH_CMD_ADDR_REG Register (Offset = 94h) [reset = 0h]

QSPI_FLASH_CMD_ADDR_REG is shown in Figure 11-1340 and described in Table 11-3307.

Table 11-3306. QSPI_FLASH_CMD_ADDR_REG Instances

Instance	Physical Address
QSPI	0294 0094h

Figure 11-1340. QSPI_FLASH_CMD_ADDR_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR_FLD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3307. QSPI_FLASH_CMD_ADDR_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR_FLD	R/W	0h	Command Address: This should be set-up before triggering the QSPI_FLASH_CMD_CTRL_REG[0] CMD_EXEC_FLD bit. It is the address used by the command specified in the opcode QSPI_FLASH_CMD_CTRL_REG[31-24] CMD_OPCODE_FLD field.

Table 11-3308. Register Call Summary for QSPI_FLASH_CMD_ADDR_REG

QSPI Functional Description • Servicing a STIG request: [0]
QSPI Registers • QSPI_FLASH_CMD_ADDR_REG Register (Offset = 94h) [reset = 0h]: [0] • QSPI Registers: [0] • QSPI_FLASH_CMD_CTRL_REG Register (Offset = 90h) [reset = 0h]: [0]

11.15.6.31 QSPI_FLASH_RD_DATA_LOWER_REG Register (Offset = A0h) [reset = 0h]

QSPI_FLASH_RD_DATA_LOWER_REG is shown in Figure 11-1341 and described in Table 11-3310.

Table 11-3309. QSPI_FLASH_RD_DATA_LOWER_REG Instances

Instance	Physical Address
QSPI	0294 00A0h

Figure 11-1341. QSPI_FLASH_RD_DATA_LOWER_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_FLD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3310. QSPI_FLASH_RD_DATA_LOWER_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA_FLD	R/W	0h	Command Read Data (Lower byte): This is the data that is returned by the flash device for any status or configuration read operation carried out by triggering the event in the control register. The register will be valid when the polling bit in the control register is low.

Table 11-3311. Register Call Summary for QSPI_FLASH_RD_DATA_LOWER_REG

QSPI Functional Description • Servicing a STIG request: [0] • Software Triggered Instruction Generator (STIG): [0]
QSPI Registers • QSPI_FLASH_RD_DATA_LOWER_REG Register (Offset = A0h) [reset = 0h]: [0] • QSPI Registers: [0]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0] • Direct Access Controller Programming Sequence: [0]

11.15.6.32 QSPI_FLASH_RD_DATA_UPPER_REG Register (Offset = A4h) [reset = 0h]

QSPI_FLASH_RD_DATA_UPPER_REG is shown in Figure 11-1342 and described in Table 11-3313.

Device Instruction Configuration Register.

Table 11-3312. QSPI_FLASH_RD_DATA_UPPER_REG Instances

Instance	Physical Address
QSPI	0294 00A4h

Figure 11-1342. QSPI_FLASH_RD_DATA_UPPER_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_FLD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3313. QSPI_FLASH_RD_DATA_UPPER_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA_FLD	R/W	0h	Command Read Data (Upper byte): This is the data that is returned by the FLASH device for any status or configuration read operation carried out by triggering the event in the control register. The register will be valid when the polling bit in the control register is low.

Table 11-3314. Register Call Summary for QSPI_FLASH_RD_DATA_UPPER_REG

QSPI Functional Description • Servicing a STIG request: [0] • Software Triggered Instruction Generator (STIG): [0]
QSPI Registers • QSPI_FLASH_RD_DATA_UPPER_REG Register (Offset = A4h) [reset = 0h]: [0] • QSPI Registers: [0]

11.15.6.33 QSPI_FLASH_WR_DATA_LOWER_REG Register (Offset = A8h) [reset = 0h]

QSPI_FLASH_WR_DATA_LOWER_REG is shown in Figure 11-1343 and described in Table 11-3316.

Table 11-3315. QSPI_FLASH_WR_DATA_LOWER_REG Instances

Instance	Physical Address
QSPI	0294 00A8h

Figure 11-1343. QSPI_FLASH_WR_DATA_LOWER_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_FLD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3316. QSPI_FLASH_WR_DATA_LOWER_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA_FLD	R/W	0h	Command Write Data Lower Byte: This is the command write data lower byte. This should be set-up before triggering the command with execute QSPI_FLASH_CMD_CTRL_REG[0] CMD_EXEC_FLD bit. It is the data that is to be written to the flash for any status or configuration write operation carried out by triggering the event in the QSPI_FLASH_CMD_CTRL_REG register.

Table 11-3317. Register Call Summary for QSPI_FLASH_WR_DATA_LOWER_REG

QSPI Functional Description • Servicing a STIG request: [0] • Software Triggered Instruction Generator (STIG): [0]
QSPI Registers • QSPI_FLASH_WR_DATA_LOWER_REG Register (Offset = A8h) [reset = 0h]: [0] • QSPI Registers: [0]
QSPI Programming Guide • Indirect Access Controller programming sequence: [0][1] • Direct Access Controller Programming Sequence: [0][1]

11.15.6.34 QSPI_FLASH_WR_DATA_UPPER_REG Register (Offset = ACh) [reset = 0h]

QSPI_FLASH_WR_DATA_UPPER_REG is shown in Figure 11-1344 and described in Table 11-3319.

Table 11-3318. QSPI_FLASH_WR_DATA_UPPER_REG Instances

Instance	Physical Address
QSPI	0294 00ACh

Figure 11-1344. QSPI_FLASH_WR_DATA_UPPER_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_FLD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3319. QSPI_FLASH_WR_DATA_UPPER_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA_FLD	R/W	0h	Command Write Data Upper Byte: This is the command write data upper byte. This should be set-up before triggering the command with execute QSPI_FLASH_CMD_CTRL_REG[0] CMD_EXEC_FLD bit. It is the data that is to be written to the flash for any status or configuration write operation carried out by triggering the event in the QSPI_FLASH_CMD_CTRL_REG register.

Table 11-3320. Register Call Summary for QSPI_FLASH_WR_DATA_UPPER_REG

QSPI Functional Description • Servicing a STIG request: [0] • Software Triggered Instruction Generator (STIG): [0]
QSPI Registers • QSPI Registers: [0] • QSPI_FLASH_WR_DATA_UPPER_REG Register (Offset = ACh) [reset = 0h]: [0]

11.15.6.35 QSPI_POLLING_FLASH_STATUS_REG Register (Offset = B0h) [reset = 0h]

QSPI_POLLING_FLASH_STATUS_REG is shown in Figure 11-1345 and described in Table 11-3322.

Table 11-3321. QSPI_POLLING_FLASH_STATUS_REG Instances

Instance	Physical Address
QSPI	0294 00B0h

Figure 11-1345. QSPI_POLLING_FLASH_STATUS_REG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							DEVICE_STATUS_VALID_FLD
R-0h							R-0h
7	6	5	4	3	2	1	0
DEVICE_STATUS_FLD							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3322. QSPI_POLLING_FLASH_STATUS_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	DEVICE_STATUS_VALID_FLD	R	0h	Polling Status Valid: This bit is set when value in the QSPI_POLLING_FLASH_STATUS_REG[7-0] DEVICE_STATUS_FLD field is valid.
7-0	DEVICE_STATUS_FLD	R	0h	Flash Status: Defines actual Status Register of Device.

Table 11-3323. Register Call Summary for QSPI_POLLING_FLASH_STATUS_REG

QSPI Functional Description • Servicing a STIG request: [0]
QSPI Registers • QSPI Registers: [0] • QSPI_POLLING_FLASH_STATUS_REG Register (Offset = B0h) [reset = 0h]: [0][1]

11.15.6.36 QSPI_MODULE_ID_REG Register (Offset = FCh) [reset = 1010309h]

QSPI_MODULE_ID_REG is shown in Figure 11-1346 and described in Table 11-3325.

Table 11-3324. QSPI_MODULE_ID_REG Instances

Instance	Physical Address
QSPI	0294 00FCh

Figure 11-1346. QSPI_MODULE_ID_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								VALUE_FLD							
R-1h								R-10309h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE_FLD															
R-10309h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3325. QSPI_MODULE_ID_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	1h	Reserved
23-0	VALUE_FLD	R	10309h	Module ID number

Table 11-3326. Register Call Summary for QSPI_MODULE_ID_REG

QSPI Registers
- QSPI_MODULE_ID_REG Register (Offset = FCh) [reset = 1010309h]: [0]
- QSPI Registers: [0]

11.15.6.37 QSPI_ECC_REVISION Register (Offset = 400h) [reset = 4E100001h]

QSPI_ECC_REVISION is shown in Figure 11-1347 and described in Table 11-3328.

Table 11-3327. QSPI_ECC_REVISION Instances

Instance	Physical Address
QSPI	0294 0400h

Figure 11-1347. QSPI_ECC_REVISION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-4E100001h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3328. QSPI_ECC_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4E100001h	TI internal data. Identifies revision of peripheral.

Table 11-3329. Register Call Summary for QSPI_ECC_REVISION

QSPI Registers
- QSPI_ECC_REVISION Register (Offset = 400h) [reset = 4E100001h]: [0]
- QSPI Registers: [0]

11.15.6.38 QSPI_ECC_VECTOR Register (Offset = 408h) [reset = 0h]

QSPI_ECC_VECTOR is shown in Figure 11-1348 and described in Table 11-3331.

Table 11-3330. QSPI_ECC_VECTOR Instances

Instance	Physical Address
QSPI	0294 0408h

Figure 11-1348. QSPI_ECC_VECTOR Register

31	30	29	28	27	26	25	24
RESERVED							READ_DONE
R-0h							R-0h
23	22	21	20	19	18	17	16
READ_ADDRESS							
R/W-0h							
15	14	13	12	11	10	9	8
TRIGGER_READ	RESERVED				RAM_ID		
R/W-0h	R-0h				R/W-0h		
7	6	5	4	3	2	1	0
RAM_ID							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3331. QSPI_ECC_VECTOR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24	READ_DONE	R	0h	Status indicating that the read is complete.
23-16	READ_ADDRESS	R/W	0h	Read address: Can be any of the registers 0294 0410h to 0294 0424h.
15	TRIGGER_READ	R/W	0h	Trigger a read operation to the specified read address.
14-11	RESERVED	R	0h	Reserved
10-0	RAM_ID	R/W	0h	ECC RAM ID to select which ECC RAM to control or read status from.

Table 11-3332. Register Call Summary for QSPI_ECC_VECTOR

QSPI Functional Description • ECC Interrupts: [0][1][2][3][4] • Reads to ECC control and status registers: [0][1][2][3]
QSPI Registers • QSPI_ECC_VECTOR Register (Offset = 408h) [reset = 0h]: [0] • QSPI Registers: [0]
QSPI Programming Guide • ECC Programming sequence: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22]

11.15.6.39 QSPI_ECC_MISC_STATUS Register (Offset = 40Ch) [reset = -h]

QSPI_ECC_MISC_STATUS is shown in Figure 11-1349 and described in Table 11-3334.

Table 11-3333. QSPI_ECC_MISC_STATUS Instances

Instance	Physical Address
QSPI	0294 040Ch

Figure 11-1349. QSPI_ECC_MISC_STATUS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					NUM_RAMS										
R-0h																					R--h										

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3334. QSPI_ECC_MISC_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Reserved
10-0	NUM_RAMS	R	-h [(1)]	Number of ECC RAMs serviced by the aggregator.

(1) Number of rams serviced by the aggregator

Table 11-3335. Register Call Summary for QSPI_ECC_MISC_STATUS

QSPI Registers
- QSPI Registers: [0]
- QSPI_ECC_MISC_STATUS Register (Offset = 40Ch) [reset = -h]: [0]

11.15.6.40 QSPI_ECC_WRAPPER_REVISION Register (Offset = 410h) [reset = -h]

QSPI_ECC_WRAPPER_REVISION is shown in Figure 11-1350 and described in Table 11-3337.

Table 11-3336. QSPI_ECC_WRAPPER_REVISION Instances

Instance	Physical Address
QSPI	0294 0410h

Figure 11-1350. QSPI_ECC_WRAPPER_REVISION Register

31	30	29	28	27	26	25	24
SCHEME		RESERVED		MODID			
R-1h		R-0h		R-E11			
23	22	21	20	19	18	17	16
MODID							
R-E11							
15	14	13	12	11	10	9	8
REVRTL					REVMAJ		
R--h					R-0h		
7	6	5	4	3	2	1	0
REVCUSTOM		REVMIN					
R-0h		R-1h					

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3337. QSPI_ECC_WRAPPER_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	SCHEME	R	1h	Scheme that this register is compliant with.
29-28	RESERVED	R	0h	Reserved
27-16	MODID	R	E11h	Module ID
15-11	REVRTL	R	-h	RTL revision
10-8	REVMAJ	R	0h	Major revision
7-6	REVCUSTOM	R	0h	Custom revision
5-0	REVMIN	R	1h	Minor revision

Table 11-3338. Register Call Summary for QSPI_ECC_WRAPPER_REVISION

QSPI Functional Description • Reads to ECC control and status registers: [0]
QSPI Registers • QSPI_ECC_WRAPPER_REVISION Register (Offset = 410h) [reset = -h]: [0] • QSPI Registers: [0]

11.15.6.41 QSPI_ECC_CONTROL Register (Offset = 414h) [reset = 7h]

QSPI_ECC_CONTROL is shown in Figure 11-1351 and described in Table 11-3340.

Table 11-3339. QSPI_ECC_CONTROL Instances

Instance	Physical Address
QSPI	0294 0414h

Figure 11-1351. QSPI_ECC_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	ERROR_ONCE	FORCE_N_ROW	FORCE_DED	FORCE_SEC	ENABLE_RMW	ECC_CHECK	ECC_ENABLE
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h	R/W-1h	R/W-1h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3340. QSPI_ECC_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	ERROR_ONCE	R/W	0h	If this bit is set, the FORCE_SEC/FORCE_DED will inject an error to the specified row only once. The FORCE_SEC bit will be cleared once a writeback happens. If writeback is not enabled, this error will be cleared the cycle following the read when the data is corrected. For double-bit errors, the FORCE_DED bit will be cleared the cycle following the double-bit error. Any subsequent reads will not force an error.
5	FORCE_N_ROW	R/W	0h	Force single/double-bit error on the next RAM read.
4	FORCE_DED	R/W	0h	Force double-bit error: Cleared the cycle following the error if ERROR_ONCE is asserted.
3	FORCE_SEC	R/W	0h	Force single-bit error: Cleared on a writeback or the cycle following the error if ERROR_ONCE is asserted.
2	ENABLE_RMW	R/W	1h	Enable read-modify-write on partial word writes.
1	ECC_CHECK	R/W	1h	Enable ECC check: ECC is completely bypassed if both ECC_ENABLE and ECC_CHECK are 0.
0	ECC_ENABLE	R/W	1h	Enable ECC generation.

Table 11-3341. Register Call Summary for QSPI_ECC_CONTROL

QSPI Functional Description • Reads to ECC control and status registers: [0]
QSPI Registers • QSPI Registers: [0] • QSPI_ECC_CONTROL Register (Offset = 414h) [reset = 7h]: [0]

Table 11-3341. Register Call Summary for QSPI_ECC_CONTROL (continued)

QSPI Programming Guide • ECC Programming sequence: [0][1][2][3][4][5][6][7]

11.15.6.42 QSPI_ECC_ERROR_CONTROL1 Register (Offset = 418h) [reset = 0h]

QSPI_ECC_ERROR_CONTROL1 is shown in Figure 11-1352 and described in Table 11-3343.

Table 11-3342. QSPI_ECC_ERROR_CONTROL1 Instances

Instance	Physical Address
QSPI	0294 0418h

Figure 11-1352. QSPI_ECC_ERROR_CONTROL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_BIT1																ECC_ROW															
R/W-0h																R/W-0h															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3343. QSPI_ECC_ERROR_CONTROL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	ECC_BIT1	R/W	0h	Column/Data bit that needs to be flipped when FORCE_SEC or FORCE_DED is set.
15-0	ECC_ROW	R/W	0h	Row address where FORCE_SEC or FORCE_DED needs to be applied. This is ignored if FORCE_N_ROW is set.

Table 11-3344. Register Call Summary for QSPI_ECC_ERROR_CONTROL1

QSPI Functional Description • Reads to ECC control and status registers: [0]
QSPI Registers • QSPI_ECC_ERROR_CONTROL1 Register (Offset = 418h) [reset = 0h]: [0] • QSPI Registers: [0]
QSPI Programming Guide • ECC Programming sequence: [0][1]

11.15.6.43 QSPI_ECC_ERROR_CONTROL2 Register (Offset = 41Ch) [reset = 0h]

QSPI_ECC_ERROR_CONTROL2 is shown in Figure 11-1353 and described in Table 11-3346.

Table 11-3345. QSPI_ECC_ERROR_CONTROL2 Instances

Instance	Physical Address
QSPI	0294 041Ch

Figure 11-1353. QSPI_ECC_ERROR_CONTROL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ECC_BIT2															
R-0h																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3346. QSPI_ECC_ERROR_CONTROL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	ECC_BIT2	R/W	0h	Data bit that needs to be flipped when FORCE_DED is set.

Table 11-3347. Register Call Summary for QSPI_ECC_ERROR_CONTROL2

QSPI Functional Description • Reads to ECC control and status registers: [0]
QSPI Registers • QSPI Registers: [0] • QSPI_ECC_ERROR_CONTROL2 Register (Offset = 41Ch) [reset = 0h]: [0]
QSPI Programming Guide • ECC Programming sequence: [0]

11.15.6.44 QSPI_ECC_ERROR_STATUS1 Register (Offset = 420h) [reset = 0h]

QSPI_ECC_ERROR_STATUS1 is shown in Figure 11-1354 and described in Table 11-3349.

Table 11-3348. QSPI_ECC_ERROR_STATUS1 Instances

Instance	Physical Address
QSPI	0294 0420h

Figure 11-1354. QSPI_ECC_ERROR_STATUS1 Register

31	30	29	28	27	26	25	24
ECC_ROW							
R-0h							
23	22	21	20	19	18	17	16
ECC_ROW							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					CLR_ECC_OTHER	CLR_ECC_DED	CLR_ECC_SEC
R-0h					R/W1TC-0h	R/W1TC-0h	R/W1TC-0h
7	6	5	4	3	2	1	0
RESERVED					ECC_OTHER	ECC_DED	ECC_SEC
R-0h					R/W1TS-0h	R/W1TS-0h	R/W1TS-0h

LEGEND: R = Read Only; R/W1TC = Read/Write 1 to Clear Bit; R/W1TS = Read/Write 1 to Set Bit; *-n* = value after reset

Table 11-3349. QSPI_ECC_ERROR_STATUS1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	ECC_ROW	R	0h	Indicates the row/address where the single or double-bit error occurred.
15-11	RESERVED	R	0h	Reserved
10	CLR_ECC_OTHER	R/W1TC	0h	'1' indicates a successive single-bit error. Writing a '1' clears the status bit.
9	CLR_ECC_DED	R/W1TC	0h	'1' indicates a pending double-bit error. Writing a '1' clears the status bit.
8	CLR_ECC_SEC	R/W1TC	0h	'1' indicates a pending single-bit error. Writing a '1' clears the status bit.
7-3	RESERVED	R	0h	Reserved
2	ECC_OTHER	R/W1TS	0h	'1' – Indicates that successive single-bit errors have occurred while a writeback is still pending. The software can also write a '1' to set the pending status and write a '1' to the corresponding clear bit to clear the status.
1	ECC_DED	R/W1TS	0h	'1' – Indicates pending double-bit error status Since the double-bit error from the ECC logic is a pulsed interrupt, this is also a status set register. The software can also write a '1' to set the pending status and write a '1' to the corresponding clear bit to clear the status.
0	ECC_SEC	R/W1TS	0h	'1' – Indicates pending single-bit error status Since the single-bit error from the ECC logic is a pulsed interrupt, this is also a status set register. The software can also write a '1' to set the pending status and write a '1' to the corresponding clear bit to clear the status.

Table 11-3350. Register Call Summary for QSPI_ECC_ERROR_STATUS1

QSPI Functional Description • ECC Interrupts: [0][1][2][3] • Reads to ECC control and status registers: [0]
QSPI Registers • QSPI_ECC_ERROR_STATUS1 Register (Offset = 420h) [reset = 0h]: [0] • QSPI Registers: [0]

11.15.6.45 QSPI_ECC_ERROR_STATUS2 Register (Offset = 424h) [reset = 0h]

QSPI_ECC_ERROR_STATUS2 is shown in Figure 11-1355 and described in Table 11-3352.

Table 11-3351. QSPI_ECC_ERROR_STATUS2 Instances

Instance	Physical Address
QSPI	0294 0424h

Figure 11-1355. QSPI_ECC_ERROR_STATUS2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ECC_BIT1															
R-0h																R-0h															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3352. QSPI_ECC_ERROR_STATUS2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	ECC_BIT1	R	0h	Indicates the bit position in the ram data that is in error. For example: a value of 1 indicates that bit 1 in the data is in error. This is valid only for single bit errors (sec).

Table 11-3353. Register Call Summary for QSPI_ECC_ERROR_STATUS2

QSPI Functional Description • ECC Interrupts: [0][1] • Reads to ECC control and status registers: [0]
QSPI Registers • QSPI Registers: [0] • QSPI_ECC_ERROR_STATUS2 Register (Offset = 424h) [reset = 0h]: [0]

11.15.6.46 QSPI_ECC_EOI Register (Offset = 43Ch) [reset = 0h]

QSPI_ECC_EOI is shown in Figure 11-1356 and described in Table 11-3355.

Table 11-3354. QSPI_ECC_EOI Instances

Instance	Physical Address
QSPI	0294 043Ch

Figure 11-1356. QSPI_ECC_EOI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EOI_WR
R-0h							W1TS

LEGEND: R = Read Only; W1TS = Write 1 to Set Bit; *-n* = value after reset

Table 11-3355. QSPI_ECC_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EOI_WR	W1TS	0h	Write to this register indicates that software has acknowledged the pending interrupt and the next interrupt can be sent to the host.

Table 11-3356. Register Call Summary for QSPI_ECC_EOI

QSPI Functional Description • ECC Interrupts: [0]
QSPI Registers • QSPI_ECC_EOI Register (Offset = 43Ch) [reset = 0h]: [0] • QSPI Registers: [0]
QSPI Programming Guide • ECC Programming sequence: [0]

11.15.6.47 QSPI_ECC_INT_STATUS_0 to QSPI_ECC_INT_STATUS_15 Register (Offset = 440h to 47Ch) [reset = 0h]

QSPI_ECC_INT_STATUS_0 to QSPI_ECC_INT_STATUS_15 is shown in Figure 11-1357 and described in Table 11-3358.

Raw interrupt status from each of the ECC RAMs. Each bit corresponds to a level pending status from an ECC RAM.

These are the raw level interrupt status bits where each bit corresponds to the pending status from an ECC RAM. The bit associations with the ECC RAMs are assigned by the ECC aggregator module. There is 1 register for up to 32 ECC RAMs. The addresses increment for every additional 32-bit register required to hold the interrupt status for all the ECC RAMs (0 to N-1).

Table 11-3357. QSPI_ECC_INT_STATUS_0 to QSPI_ECC_INT_STATUS_15 Instances

Instance	Physical Address
QSPI	0294 0440h to 0294 047Ch

Figure 11-1357. QSPI_ECC_INT_STATUS_0 to QSPI_ECC_INT_STATUS_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITMASK																															
R-0h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3358. QSPI_ECC_INT_STATUS_0 to QSPI_ECC_INT_STATUS_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITMASK	R	0h	Level interrupt status from each ECC RAM. 1=pending status, 0=not pending.

Table 11-3359. Register Call Summary for QSPI_ECC_INT_STATUS_0

QSPI Registers
- QSPI_ECC_INT_STATUS_0 to QSPI_ECC_INT_STATUS_15 Register (Offset = 440h to 47Ch) [reset = 0h]: [0]
- QSPI Registers: [0]

11.15.6.48 QSPI_ECC_INT_ENABLE_0 to QSPI_ECC_INT_ENABLE_15 Register (Offset = 480h to 4BCh) [reset = 0h]

QSPI_ECC_INT_ENABLE_0 to QSPI_ECC_INT_ENABLE_15 is shown in Figure 11-1358 and described in Table 11-3361.

Interrupt Enable Set Registers.

These are interrupt enables associated with the interrupt from each of the ECC RAMs. Writing a '1' to a bit position in the register enables the interrupt from the associated ECC RAM. There is 1 register for up to 32 ECC RAMs. The addresses increment for every additional 32-bit register required for all ECC RAMs (0 to N-1).

Table 11-3360. QSPI_ECC_INT_ENABLE_0 to QSPI_ECC_INT_ENABLE_15 Instances

Instance	Physical Address
QSPI	0294 0480h to 0294 04BCh

Figure 11-1358. QSPI_ECC_INT_ENABLE_0 to QSPI_ECC_INT_ENABLE_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITMASK																															
W1TS-0h																															

LEGEND: W1TS = Write 1 to Set Bit; *-n* = value after reset

Table 11-3361. QSPI_ECC_INT_ENABLE_0 to QSPI_ECC_INT_ENABLE_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITMASK	W1TS	0h	Write 1 to enable interrupt from the associated ECC RAM.

Table 11-3362. Register Call Summary for QSPI_ECC_INT_ENABLE_0

QSPI Registers
- QSPI_ECC_INT_ENABLE_0 to QSPI_ECC_INT_ENABLE_15 Register (Offset = 480h to 4BCh) [reset = 0h]: [0]
- QSPI Registers: [0]

11.15.6.49 QSPI_ECC_INT_CLEAR_0 to QSPI_ECC_INT_CLEAR_15 Register (Offset = 4C0h to 4FCh) [reset = 0h]

QSPI_ECC_INT_CLEAR_0 to QSPI_ECC_INT_CLEAR_15 is shown in Figure 11-1359 and described in Table 11-3364.

Interrupt Enable Clear Registers.

These are interrupt enable clear bits associated with the interrupt from each of the ECC RAMs. Writing a '1' to a bit position in the register disables the interrupt from the associated ECC RAM. There is 1 register for up to 32 ECC RAMs. The addresses increment for every additional 32-bit register required for all the ECC RAMs (0 to N-1).

Table 11-3363. QSPI_ECC_INT_CLEAR_0 to QSPI_ECC_INT_CLEAR_15 Instances

Instance	Physical Address
QSPI	0294 04C0h to 0294 04FCh

Figure 11-1359. QSPI_ECC_INT_CLEAR_0 to QSPI_ECC_INT_CLEAR_15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITMASK																															
W1TC-0h																															

LEGEND: W1TC = Write 1 to Clear Bit; *-n* = value after reset

Table 11-3364. QSPI_ECC_INT_CLEAR_0 to QSPI_ECC_INT_CLEAR_15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITMASK	W1TC	0h	Write 1 to disable interrupt from the associated ECC RAM.

Table 11-3365. Register Call Summary for QSPI_ECC_INT_CLEAR_0

QSPI Registers
- QSPI_ECC_INT_CLEAR_0 to QSPI_ECC_INT_CLEAR_15 Register (Offset = 4C0h to 4FCh) [reset = 0h]: [0]
- QSPI Registers: [0]

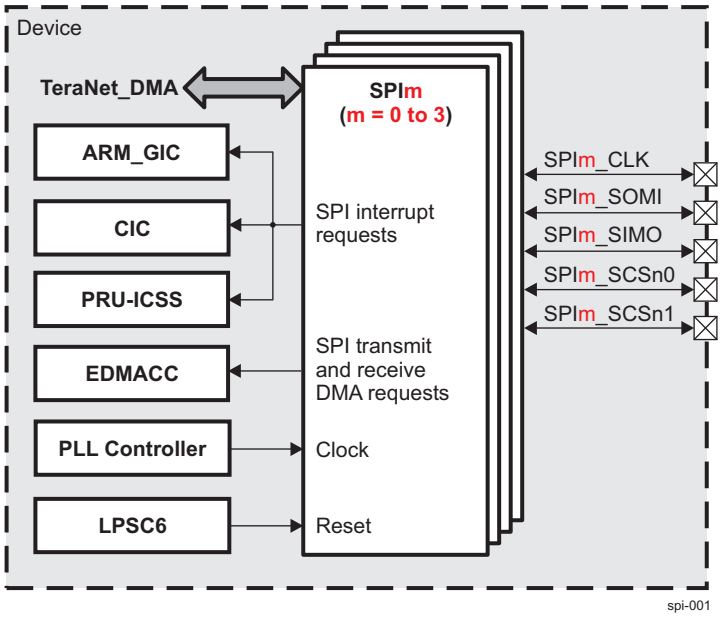
11.16 Serial Peripheral Interface (SPI)

This section describes the Serial Peripheral Interface (SPI) module for the device.

11.16.1 SPI Overview

The SPI module is a master/slave high-speed synchronous serial input/output interface that allows a serial bit stream of programmed length (2 to 16 bits) to be shifted in and out of the device at a programmed bit-transfer rate. There are four separate SPI modules (SPI0, SPI1, SPI2, and SPI3) in the device (see Figure 11-1360). All these four modules support up to two external devices (two chip selects) and are able to work as both master and slave. The SPI module allows multiple programmable chip-selects. It is normally used for communication between the device and external peripherals. Typical applications include interface to external I/O or peripheral expansion via devices such as shift registers, display drivers, SPI EEPROMS, and analog-to-digital converters. The SPI module may be used to connect to serial flash memory devices for booting. The SPI module supports EDMA events and can be used in conjunction with EDMA for data transfer with minimal CPU overhead.

Figure 11-1360. SPI Modules SPI0, SPI1, SPI2, and SPI3



The SPI module has the following features:

- 16-bit Shift register
- 16-bit Receive buffer register (SPIBUF) and 16-bit Receive buffer emulation *alias* register (SPIEMU)
- 16-bit Transmit data register (SPIDAT0) and 16-bit Transmit data and format selection register (SPIDAT1)
- 8-bit Baud clock generator
- Serial clock (SPIm_CLK) I/O pin
- Slave in, master out (SPIm_SIMO) I/O pin
- Slave out, master in (SPIm_SOMI) I/O pin
- 2 Chip select signals (SPIm_SCSn0 and SPIm_SCSn1)
- Programmable SPI clock frequency range
- Programmable character length (2 to 16 bits)
- Programmable clock phase (delay or no delay)
- Programmable clock polarity (high or low)
- Interrupt capability
- DMA support (read/write synchronization events)

- Operates at up to 50 MHz in master mode and 25 MHz in slave mode (actual speed depends on SPI functional clock and SPI clock divider)

The SPI module allows software to program the following options:

- SPIm_CLK frequency (SPI functional clock / 2 through SPI functional clock / 256)
- 3-pin and 4-pin options
- Character length (2 to 16 bits) and shift out direction (MSB/LSB first)
- Clock phase (delay or no delay) and polarity (high or low)
- Delay between transmissions in master mode
- Chip select setup and hold times in master mode
- Chip select hold in master mode

The SPI module does not support the following features:

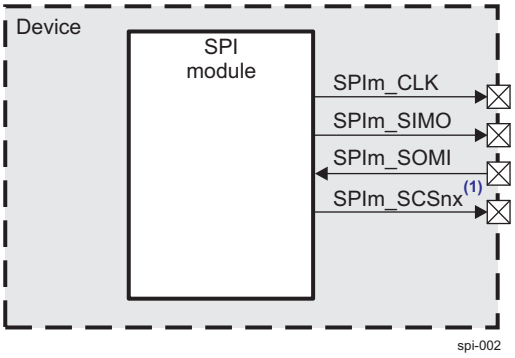
- Multibuffer mode
- Parallel mode and parity
- GPIO mode

11.16.2 SPI Environment

11.16.2.1 Basic SPI Pins for Master Mode

Figure 11-1361 shows all of the SPI interface signals in master mode.

Figure 11-1361. SPI Interface Signals in Master Mode



(1) x = 0 to 1

Table 11-3366 describes the SPI I/O in master mode.

Table 11-3366. SPI I/O Description (Master Mode)

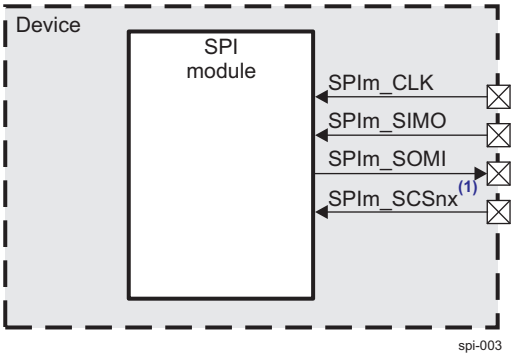
Signal Name(1)	I/O(2)	Description
SPIm_CLK	O	Serial clock output for master mode
SPIm_SIMO	O	Serial data output in master mode
SPIm_SOMI	I	Serial data input in master mode
SPIm_SCSnx(3)	O	Chip-select output for master mode

(1) m = 0 to 3
(2) I = Input; O = Output
(3) x = 0 to 1

11.16.2.2 Basic SPI Pins for Slave Mode

Figure 11-1362 shows all of the SPI interface signals in slave mode.

Figure 11-1362. SPI Interface Signals in Slave Mode



(1) x = 0 or 1

Table 11-3367 describes the SPI I/O in slave mode.

Table 11-3367. SPI I/O Description (Slave Mode)

Signal Name[1]	I/O[2]	Description
SPIm_CLK	I	Serial clock input for slave mode
SPIm_SIMO	I	Serial data input in slave mode
SPIm_SOMI	O	Serial data output in slave mode
SPIm_SCSnx[3]	I	Chip-select input for slave mode

(1) m = 0 to 3
(2) I = Input; O = Output
(3) x = 0 or 1

11.16.2.3 Data Formats

The SPI module provides the capability to configure four independent data formats. These formats are configured by programming the corresponding SPI data format registers (SPIFMTn, where n = 0 to 3). In each data format register, the following characteristics of the SPI operation are selected:

- Character length from 2 to 16 bits: The character length is configured by the SPIFMTn[4-0] CHARLEN field.
- Shift out direction (MSB first or LSB first): The shift out direction is configured by the SPIFMTn[20] SHIFTDIR bit.
- Clock polarity: The clock polarity is configured by the SPIFMTn[17] POLARITY bit.
- Clock phase: The clock phase is configured by the SPIFMTn[16] PHASE bit.

The data format is chosen on each transaction. Transmit data is written to the SPI transmit data register 1 (SPIDAT1) and in the same write the SPIDAT1[25-24] DFSEL field indicates which data format is to be used for the next transaction. Alternatively, the data format can be configured once and applies to all transactions that follow until the data format is changed.

11.16.2.3.1 Character Length

The character length is configured by the SPIFMTn[4-0] CHARLEN field. Legal values are 2 bits (2h) to 16 bits (10h). The character length is independently configured for each of the four data formats; and it must be programmed in both master mode and slave mode.

Transmit data is written to the SPIDAT1 register. The transmit data must be written right-justified irrespective of the character length. The SPI module automatically sends out the data correctly based on the chosen data format.

Figure 11-1363 shows how a 12-bit word (EC9h) must be written to the transmit buffer in order to be transmitted correctly.

Figure 11-1363. Format for Transmitting 12-Bit Word

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
x	x	x	x	1	1	1	0	1	1	0	0	1	0	0	1

The data received in the SPIBUF register is right-justified irrespective of the character length and is padded with 0s when character length is less than 16.

Figure 11-1364 shows how a 10-bit word (3A2h) is stored in the buffer once it is received.

Figure 11-1364. Format for 10-Bit Received Word

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	1	1	1	0	1	0	0	0	1	0

11.16.2.3.2 Shift Direction

The shift out direction is configured as most-significant bit (MSB) first or least significant bit (LSB) first. The SPI module supports automatic right-alignment of receive data independent of shift direction and data word length. Transmit data has to be written right-aligned and the internal shift register will sort them out according to selected shift direction and data word length for correct transfer.

The shift out direction is selected by the SPIFMTn[20] SHIFTDIR bit. The shift out direction is independently configured for each of the four data formats.

- When SPIFMTn[20] SHIFTDIR bit is set to '0h', the transmit data is shifted out with MSB first.
- When SPIFMTn[20] SHIFTDIR bit is set to '1h', the transmit data is shifted out with LSB first.

11.16.2.3.3 Clock Phase and Polarity

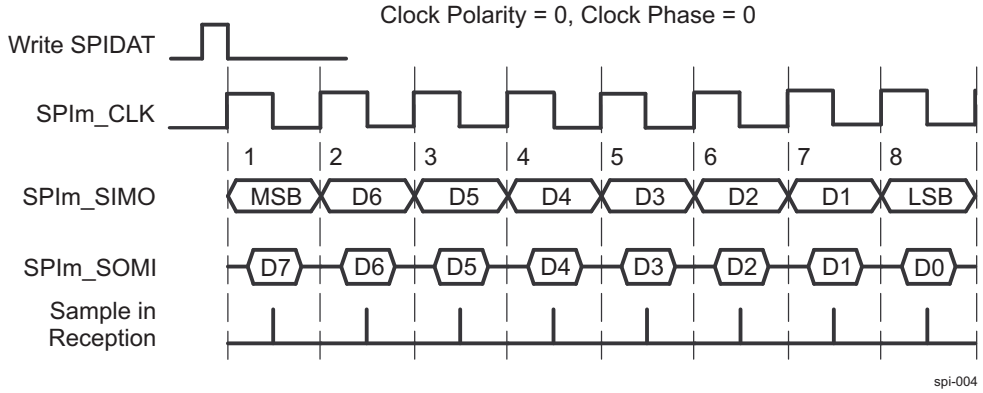
The SPI module provides the flexibility to program four different clock mode combinations in which the SPIm_CLK signal may operate enabling a choice of the clock phase (delay or no delay) and the clock polarity (rising edge or falling edge). When operating with PHASE bit active, the SPI module makes the first bit of data available after the SPIDAT1 register is written and before the first edge of the SPIm_CLK signal. The data input and output edges depend on the values of both the SPIFMTn[17] POLARITY and SPIFMTn[16] PHASE bits as shown in Table 11-3368.

Table 11-3368. Clocking Modes

SPIFMTn[17] POLARITY	SPIFMTn[16] PHASE	Action
0	0	Data is output on the rising edge of SPIm_CLK. Input data is latched on the falling edge.
0	1	Data is output one half-cycle before the first rising edge of SPIm_CLK and on subsequent falling edges. Input data is latched on the rising edge of SPIm_CLK.
1	0	Data is output on the falling edge of SPIm_CLK. Input data is latched on the rising edge.
1	1	Data is output one half-cycle before the first falling edge of SPIm_CLK and on subsequent rising edges. Input data is latched on the falling edge of SPIm_CLK.

Figure 11-1365 to Figure 11-1368 show the four possible modes of the SPIm_CLK signal. Having four signal options allows the SPI module to interface with different types of serial devices. Also shown are the SPIm_CLK control bit polarity and phase values corresponding to each signal.

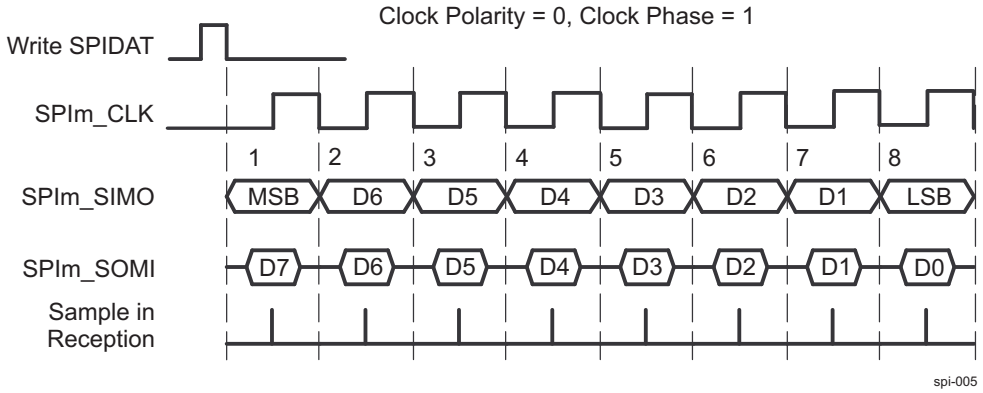
Figure 11-1365. Clock Mode with POLARITY = 0 and PHASE = 0(1)



(1) Clock phase = 0 (SPIm_CLK without delay)

- Data is output on the rising edge of SPIm_CLK.
- Input data is latched on the falling edge of SPIm_CLK.
- A write to the SPIDAT register starts SPIm_CLK.

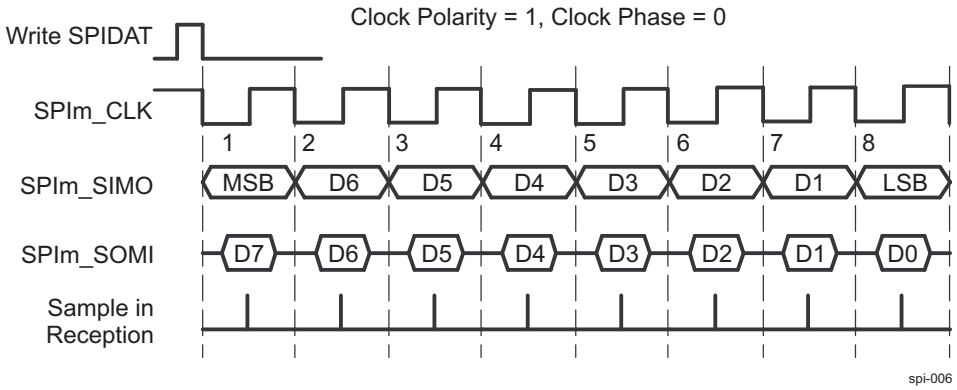
Figure 11-1366. Clock Mode with POLARITY = 0 and PHASE = 1(1)



(1) Clock phase = 1 (SPIm_CLK with delay)

- Data is output one-half cycle before the first rising of SPIm_CLK and on subsequent falling edges of SPIm_CLK.
- Input data is latched on the rising edge of SPIm_CLK.

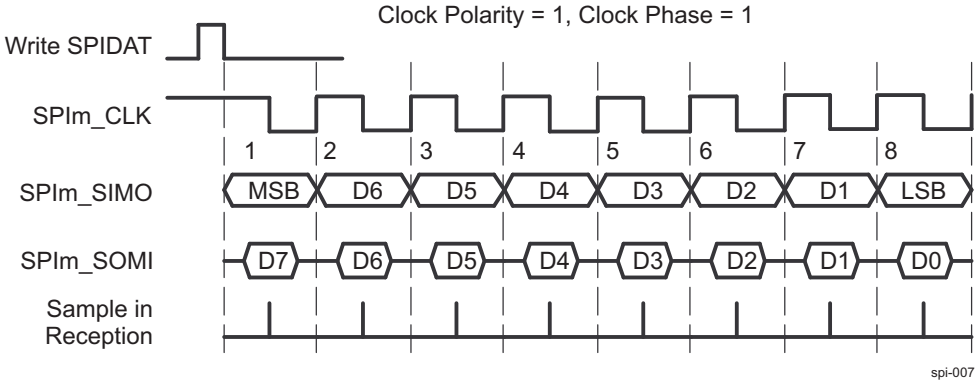
Figure 11-1367. Clock Mode with POLARITY = 1 and PHASE = 0(1)



(1) Clock phase = 0 (SPIm_CLK without delay)

- Data is output on the falling edge of SPIm_CLK.
- Input data is latched on the rising edge of SPIm_CLK.
- A write to the SPIDAT register starts SPIm_CLK.

Figure 11-1368. Clock Mode with POLARITY = 1 and PHASE = 1(1)



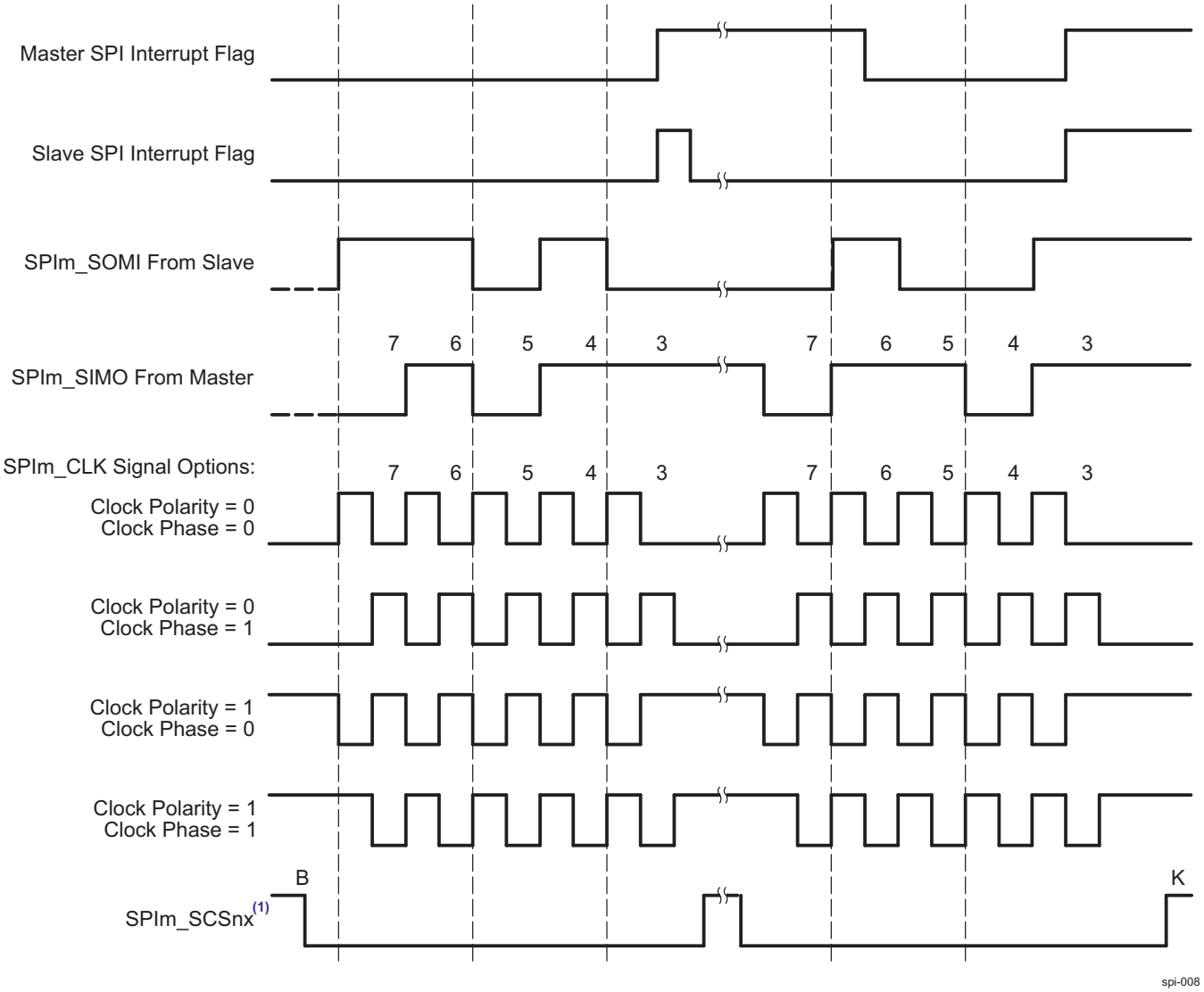
(1) Clock phase = 1 (SPIm_CLK with delay)

- Data is output one-half cycle before the first falling edge of SPIm_CLK and on the subsequent rising edges of SPIm_CLK.
- Input data is latched on the falling edge of SPIm_CLK.

11.16.2.3.4 SPI Data Transfer Example

Figure 11-1369 shows a SPI data transfer between two devices using a character length of five bits.

Figure 11-1369. SPI Data Transfer, Five Bits per Character (4-Pin with Chip Select Option)



(1) x = 0 to 1 (for master mode); x = 0 or 1 (for slave mode)

11.16.3 SPI Integration

This section describes the module integration in the device, including information about clocks, resets, and hardware requests.

Figure 11-1370 shows the SPI module integration.

Figure 11-1370. SPI Integration

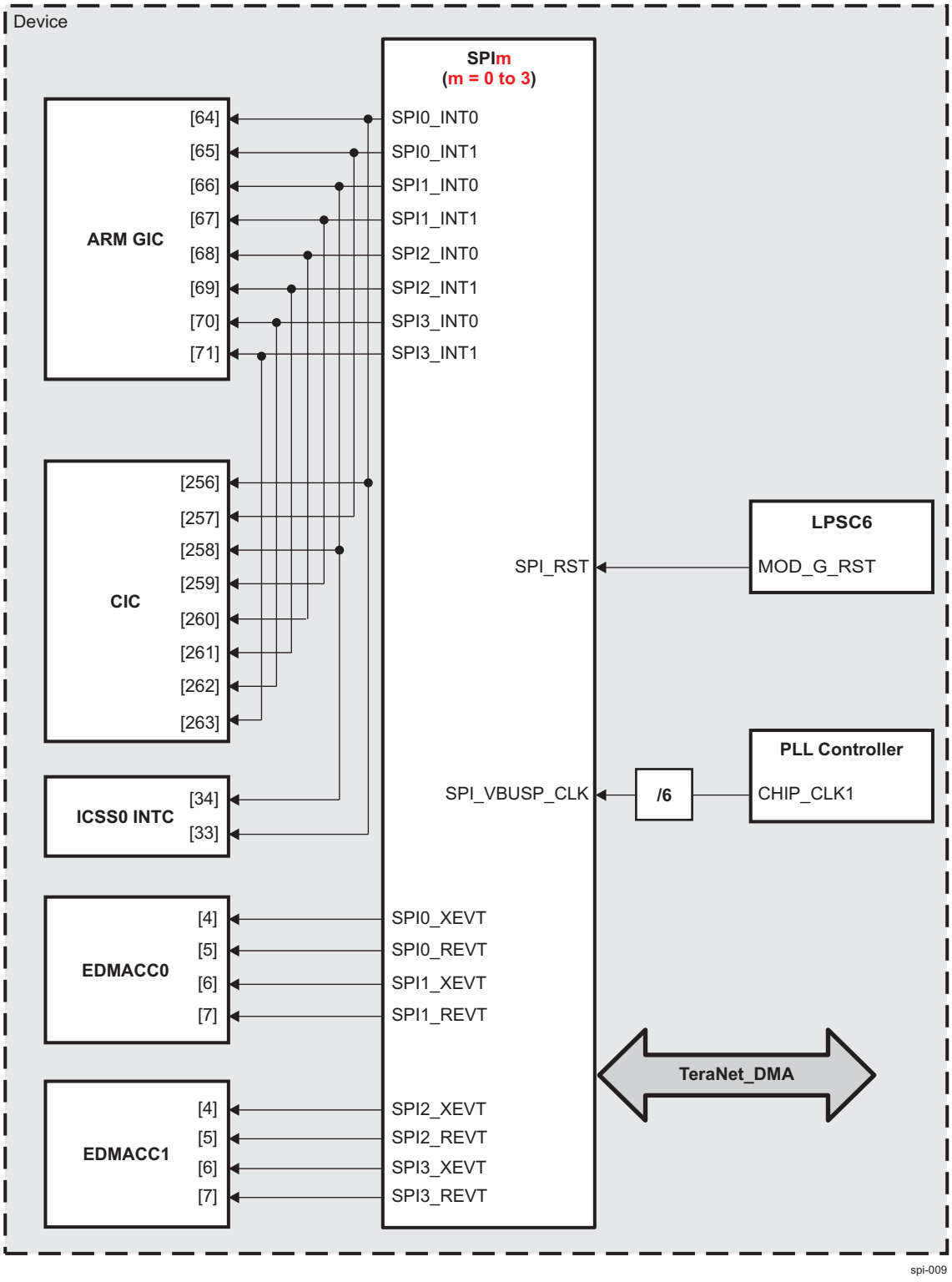


Table 11-3369 through Table 11-3371 summarize the integration of the module in the device.

Table 11-3369. SPI Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
SPI0	PD5	LPSC6	TeraNet_DMA
SPI1	PD5	LPSC6	TeraNet_DMA
SPI2	PD5	LPSC6	TeraNet_DMA
SPI3	PD5	LPSC6	TeraNet_DMA

Table 11-3370. SPI Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
SPI0	SPI_VBUSP_CLK	CHIP_CLK1 / 6	PLL Controller	Interface and Funtional Clock
SPI1	SPI_VBUSP_CLK	CHIP_CLK1 / 6	PLL Controller	Interface and Funtional Clock
SPI2	SPI_VBUSP_CLK	CHIP_CLK1 / 6	PLL Controller	Interface and Funtional Clock
SPI3	SPI_VBUSP_CLK	CHIP_CLK1 / 6	PLL Controller	Interface and Funtional Clock
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
SPI0	SPI_RST	MOD_G_RST	LPSC6	Module Asynchronous Reset
SPI1	SPI_RST	MOD_G_RST	LPSC6	Module Asynchronous Reset
SPI2	SPI_RST	MOD_G_RST	LPSC6	Module Asynchronous Reset
SPI3	SPI_RST	MOD_G_RST	LPSC6	Module Asynchronous Reset

Table 11-3371. SPI Hardware Requests

Interrupt Requests					
Module Instance	**Event Name**	**Mapped To Input Event [Number]**			**Description**
		ARM GIC	**CIC**	**ICSS0 INTC**	
SPI0	SPI0_INT0	[64]	[256]	[33]	SPI0 INT0 Interrupt Request
	SPI0_INT1	[65]	[257]	–	SPI0 INT1 Interrupt Request
SPI1	SPI1_INT0	[66]	[258]	[34]	SPI1 INT0 Interrupt Request
	SPI1_INT1	[67]	[259]	–	SPI1 INT1 Interrupt Request
SPI2	SPI2_INT0	[68]	[260]	–	SPI2 INT0 Interrupt Request
	SPI2_INT1	[69]	[261]	–	SPI2 INT1 Interrupt Request
SPI3	SPI3_INT0	[70]	[262]	–	SPI3 INT0 Interrupt Request
	SPI3_INT1	[71]	[263]	–	SPI3 INT1 Interrupt Request
DMA Requests					
Module Instance	**Event Name**	**Mapped To Input Event [Number]**			**Description**
		EDMACC0	**EDMACC1**		
SPI0	SPI0_XEVT	[4]	–		SPI0 Transmit Event
	SPI0_REVT	[5]	–		SPI0 Receive Event
SPI1	SPI1_XEVT	[6]	–		SPI1 Transmit Event
	SPI1_REVT	[7]	–		SPI1 Receive Event
SPI2	SPI2_XEVT	–	[4]		SPI2 Transmit Event
	SPI2_REVT	–	[5]		SPI2 Receive Event
SPI3	SPI3_XEVT	–	[6]		SPI3 Transmit Event
	SPI3_REVT	–	[7]		SPI3 Receive Event

NOTE: For description of INT0 and INT1 interrupt requests, see Section 11.16.4.5, *Interrupt Support*.

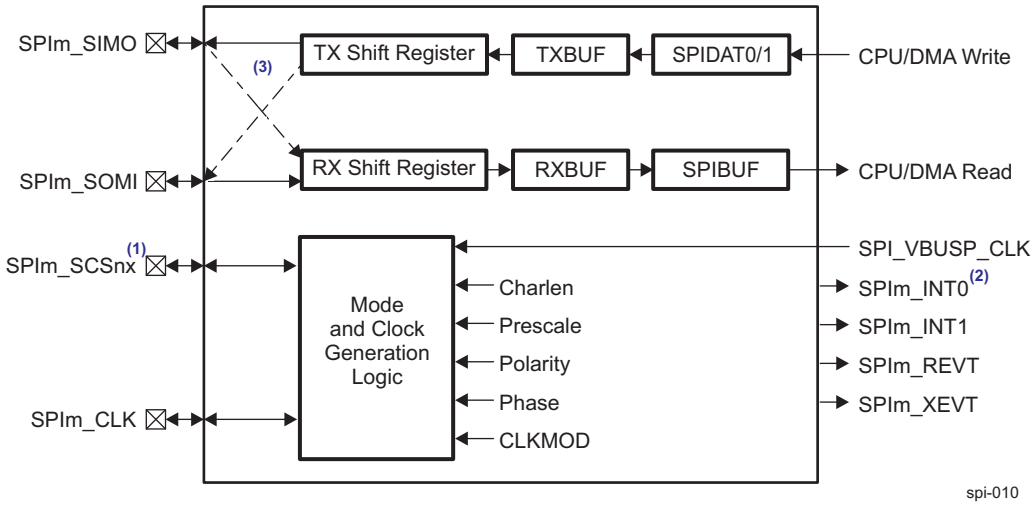
For more information about the DMA requests, see Section 11.16.4.6, *DMA Events Support*.

11.16.4 SPI Functional Description

11.16.4.1 SPI Block Diagram

Figure 11-1371 shows the SPI module.

Figure 11-1371. SPI Block Diagram



(1) x = 0 to 1 (for master mode); x = 0 or 1 (for slave mode)

(2) m = 0 to 3

(3) Solid line represents data flow for SPI master mode and dashed line represents data flow for SPI slave mode.

11.16.4.2 Operation Modes

The SPI module operates in a master or slave mode. The SPIGCR1[0] MASTER bit selects the configuration of the SPIm_SIMO and SPIm_SOMI pins and the SPIGCR1[1] CLKMOD bit determines whether an internal or external clock source will be used. The chip select (SPIm_SCSnx) pins are used as outputs when communicating with multiple slave devices. When the SPI module operates in master mode a write to the SPIDAT1 register activates the SPIm_SCSnx pin which allows a slave connected to that signal to be selected. In slave mode, only one chip select can be used. In this case it is input.

NOTE: Note: Although there are two different bits which control the Master/Slave mode functions, only two of their combinations are valid. For Master mode of operation: MASTER = '1h', CLKMOD = '1h'. For Slave mode of operation: MASTER = '0h', CLKMOD = '0h'. Any other combinations of these two bits may not yield any desirable operation of the module and must be avoided.

The SPI module supports two options:

- 3-pin option
- 4-pin with chip select option

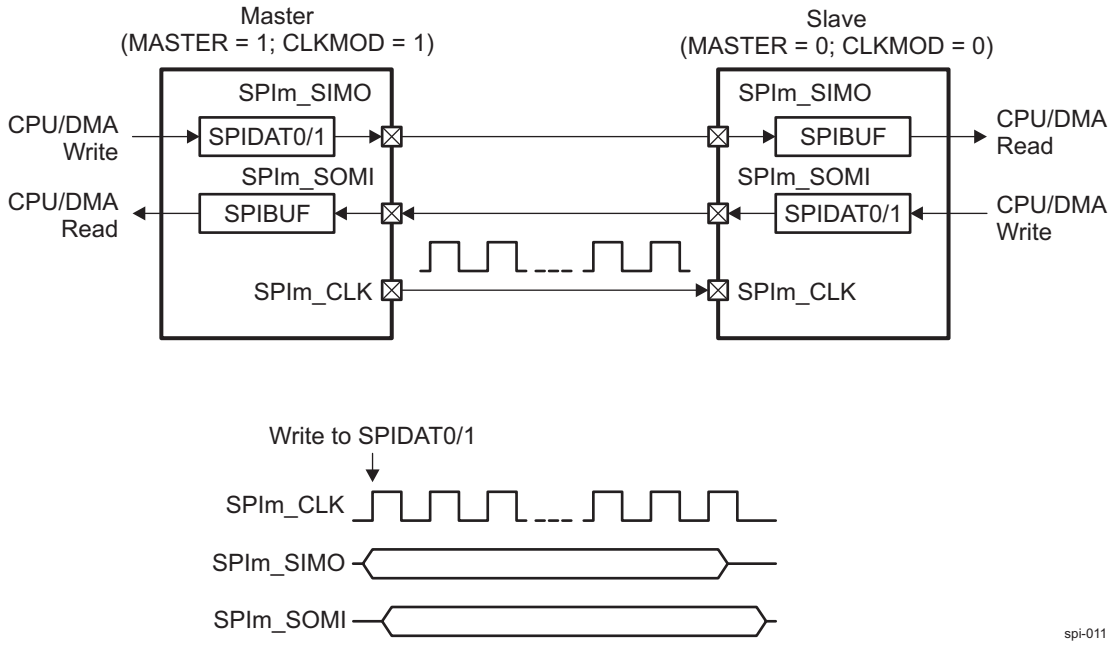
The 3-pin option is the basic clock, data in, and data out SPI interface and uses the SPIm_CLK, SPIm_SIMO, and SPIm_SOMI pins. The 4-pin with chip select option adds the SPIm_SCSnx pin that is used to support multiple SPI slave devices on a single SPI bus.

11.16.4.2.1 SPI Operation: 3-Pin Mode

NOTE: If only unidirectional communication is required, the SPIm_CLK pin and the two data pins (SPIm_SOMI and SPIm_SIMO) must all be configured as functional pins. A 2-pin unidirectional mode is not supported.

In master mode configuration (SPIGCR1[0] MASTER and SPIGCR1[1] CLKMOD bits are set to '1h'), the SPI module provides the serial clock on the SPIm_CLK pin for the entire serial communications network. Data is output on the SPIm_SIMO pin and latched in from the SPIm_SOMI pin (see Figure 11-1372). Data written to the shift register (SPIDAT1) initiates data transmission on the SPIm_SIMO pin, most significant bit (MSB) first. Simultaneously, received data is shifted through the SPIm_SOMI pin. When the selected number of bits has been transmitted, the recevied data in the RX Shift Register is transferred to the SPIBUF register for the CPU to read. Data is stored right-justified in the SPIBUF register.

Figure 11-1372. SPI 3-Pin Option



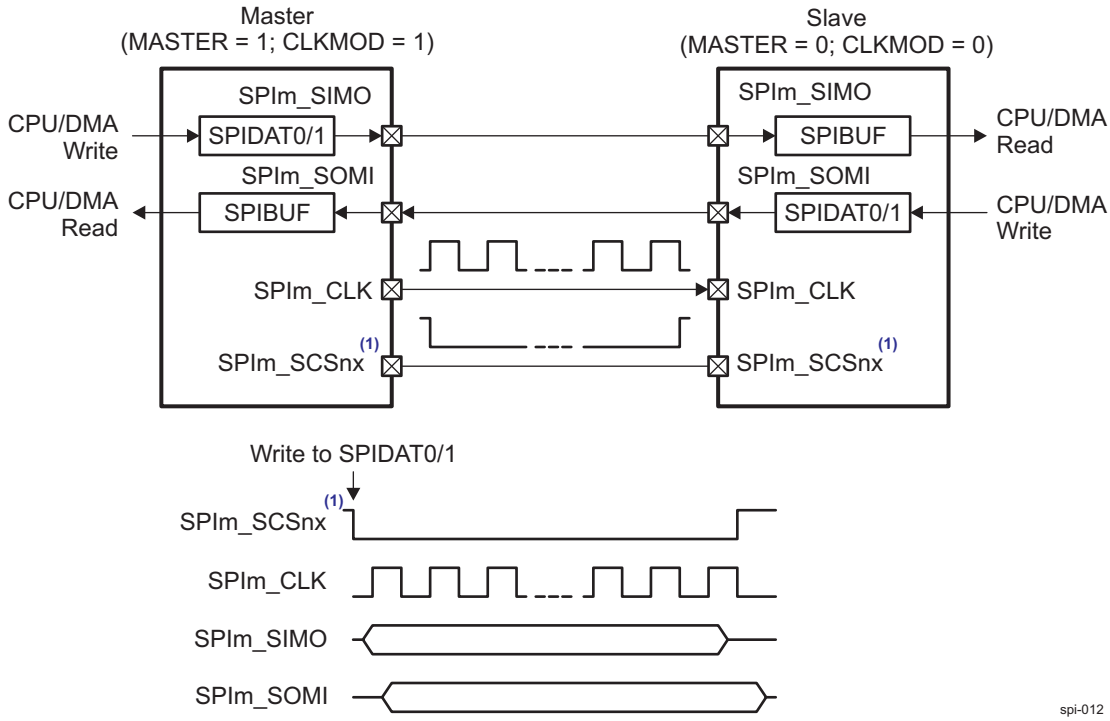
In slave mode configuration (SPIGCR1[0] MASTER and SPIGCR1[1] CLKMOD bits are set to '0h'), data is shifted out on the SPIm_SOMI pin and in on the SPIm_SIMO pin. The SPIm_CLK pin is used as an input for the serial shift clock, which is supplied from the external network master. The transfer rate is defined by this clock. Data written to the SPIDAT1 register is transmitted to the network when the SPIm_CLK signal is received from the network master. To receive data, the SPI module waits for the network master to send the SPIm_CLK signal and then shifts the data present on the SPIm_SIMO pin into the RX Shift Register. If data has to be transmitted by the slave while receiving, the data must be written to the SPIDAT1 register before the beginning of the SPIm_CLK signal.

NOTE: Either the SPIDAT0 or SPIDAT1 register can be used on both master and slave sides.

11.16.4.2.2 SPI Operation: 4-Pin Mode

The three-pin option and the four-pin options of the SPI module are identical in the master mode (SPIGCR1[1] CLKMOD bit is set to '1h'), except that the four-pin option uses SPIm_SCSnx pins. Figure 11-1373 shows SPI 4-Pin Option with SPIm_SCSnx pin.

Figure 11-1373. SPI 4-Pin Option with SPIm_SCSnx pin



(1) x = 0 to 1 (for master mode); x = 0 or 1 (for slave mode)

NOTE: Either the SPIDAT0 or SPIDAT1 register can be used on both master and slave sides.

11.16.4.2.2.1 Multiple Chip Selects

The SPIm_SCSnx pins that are going to be used must be configured as functional (SPIPC0[1-0] SCSFUN). The default value for the SPIm_SCSnx pins when all slaves are deactivated can be configured through the SPIDEF register. There is a separate bit for each chip select signal within this register. This functionality allows different slaves with different chip-select polarity to be activated by the SPI module. During transmission, the value of the SPIDAT1[23-16] CSNR field is applied on the chip select pins. Depending on this value certain chip select is activated to establish data transmission to the associated slave. When in slave mode the SPI module can be selected only by an active value of '0h' on its SPIm_SCSnx pin. However, when in master mode the SPI module is capable of driving both '0h' or '1h' as the active value of any of the SPIm_SCSnx pins. This is fully controlled by the SPIDAT1[23-16] CSNR field.

NOTE: In slave mode, only one chip select pin can be used.

11.16.4.2.3 Master Mode Settings

The two master mode options are defined by the configuration bit settings listed in Table 11-3372. Other configuration bits may take any value in the range listed in Table 11-3373. The values listed in Table 11-3372 and Table 11-3373 should not be changed while the SPIGCR1[24] ENABLE bit is set to '1h'. Note that in certain cases the allowed values may still be ignored. For example, Table 11-3373 indicates that the SPIDELAY register may take a range of values in Master 3-pin mode; however, the SPIDELAY register has no effect in Master 3-pin mode. For complete details on each mode, see Section 11.16.4.2.1, *SPI Operation: 3-Pin Mode*, Section 11.16.4.2.2 *SPI Operation: 4-Pin Mode*, and Section 11.16.4.2.3.1, *Master Mode Timing Options*.

Table 11-3372. SPI Register Settings Defining Master Modes

Register	Bit(s)	Master 3-pin	Master 4-pin Chip Select
SPIGCR0	RESET	1h	1h
SPIGCR1	ENABLE	1h	1h
SPIGCR1	LOOPBACK	0h	0h
SPIGCR1	CLKMOD	1h	1h
SPIGCR1	MASTER	1h	1h
SPIPC0	SOMIFUN	1h	1h
SPIPC0	SIMOFUN	1h	1h
SPIPC0	CLKFUN	1h	1h
SPIPC0	SCSFUN	0h	1h, 2h, 3h

Table 11-3373. Allowed SPI Register Settings in Master Modes

Register	Bit(s)	Master 3-pin	Master 4-pin Chip Select
SPIFMTn	WDELAY	0h to 3Fh	0h to 3Fh
SPIFMTn	SHIFTDIR	-h	-h
SPIFMTn	DISCSTIMERS	-h	-h
SPIFMTn	POLARITY	-h	-h
SPIFMTn	PHASE	-h	-h
SPIFMTn	PRESCALE	0h to FFh	0h to FFh
SPIFMTn	CHARLEN	2h to 10h	2h to 10h
SPIDELAY	C2TDELAY	0h to FFh	0h to FFh
SPIDELAY	T2CDELAY	0h to FFh	0h to FFh

11.16.4.2.3.1 Master Mode Timing Options

In master mode the SPI module supports several options to modify the timing of its generation of the chip select signal (SPIm_SCSnx). This allows the SPI module to support the timing requirements of various slave devices without adding additional overhead to the CPU by generating the appropriate delays automatically.

11.16.4.2.3.1.1 Chip Select Setup Time

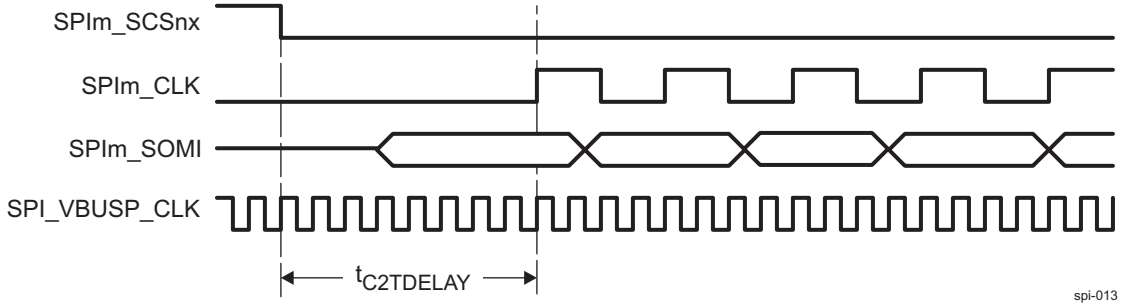
In order to support slow slave devices the SPI module can be configured to delay the data transmission after the chip select is activated. This delay is controlled by the SPIDELAY[31-24] C2TDELAY field and can be configured between 2 and 257 SPI_VBUSP_CLK cycles. The C2TDELAY is applicable only in 4-pin with chip select mode. The C2TDELAY begins when the SPI master asserts SPIm_SCSnx signal. The C2T delay period is specified by:

duration of C2TDELAY period = (SPIDELAY[31-24] C2TDELAY + 2) * SPI_VBUSP_CLK Period

The previous value of the SPIDAT1[28] CSHOLD bit must be set to '0h' for the C2T delay to be enabled.

Figure 11-1374 shows C2TDELAY example.

Figure 11-1374. Example: $t_{C2TDELAY}$ = 8 SPI_VBUSP_CLK Cycles



NOTE: The chip select setup delay configured through the SPIDELAY[31-24] C2TDELAY field is not applied between transactions which have the SPIDAT1[28] CSHOLD bit set to '1h'.

11.16.4.2.3.1.2 Chip Select Hold Time

In order to support slow slave devices the SPI module can also be configured to delay the chip select deactivation after the last data bit transfer. This delay is controlled by the SPIDELAY[23-16] T2CDELAY bit and can be configured between for 1 and 256 SPI_VBUSP_CLK cycles. The T2CDELAY is applicable only in 4-pin with chip select mode. The T2CDELAY begins after the data shifting period ends. The T2C delay period is specified by:

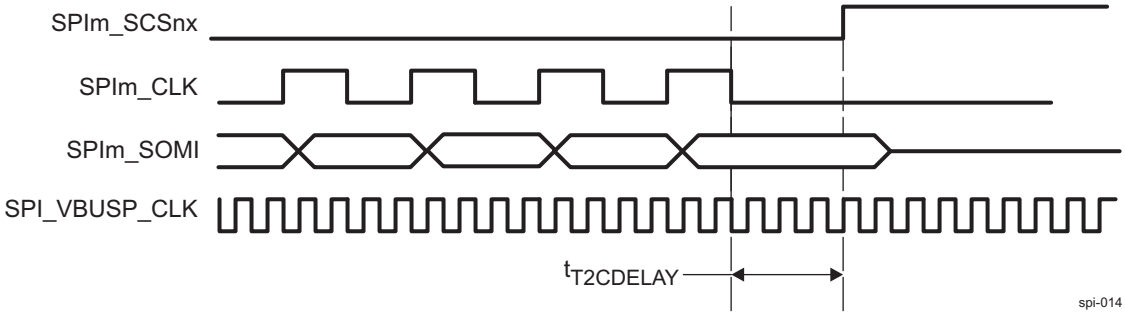
duration of T2CDELAY period = (SPIDELAY[32-16] T2CDELAY + 1) * SPI_VBUSP_CLK Period

If the SPIFMTn[16] PHASE bit is '0h', then the T2CDELAY period lasts for an additional ½ SPIm_CLK signal time over that specified by the above equation.

The current value of the SPIDAT1[28] CSHOLD bit must be set to '0h' for T2C delay to be enabled.

Figure 11-1375 shows T2CDELAY example.

Figure 11-1375. Example: $t_{T2CDELAY}$ = 4 SPI_VBUSP_CLK Cycles



NOTE: The chip select hold delay configured through the SPIDELAY[23-16] T2CDELAY field is not applied between transactions which have the SPIDAT1[28] CSHOLD bit set to '1h'.

11.16.4.2.3.1.3 Automatic Delay Between Transfers

The SPI master can automatically insert a delay of between 2 and 65 SPI_VBUSP_CLK cycles between transmissions. This delay is controlled by the SPIFMTn[29-24] WDELAY field and is enabled by setting the SPIDAT1[26] WDEL bit to '1h'. The WDELAY period begins when the T2CDELAY period terminates (if T2C delay period was enabled) or when the master deasserts SPIm_SCSnx signal (if T2C delay periods is disabled). If a transfer is initiated by writing a 32-bit value to the SPIDAT1 register, then the new values of SPIDAT1[26] WDEL and SPIFMTn[29-24] WDELAY bits are used; otherwise, the old values of SPIDAT1[26] WDEL and SPIFMTn[29-24] WDELAY are used. The WDELAY delay period is specified by:

duration of WDELAY period = (SPIFMTn[29-24] WDELAY + 2) * SPI_VBUSP_CLK Period

11.16.4.2.3.1.4 Chip Select Hold Option

There are slave devices available that require the chip select signal to be held continuously active during several consecutive data word transfers. Other slave devices require the chip select signal to be deactivated between consecutive data word transfers. The SPI module supports both types of slave devices. The SPIDAT1[28] CSHOLD bit selects between the two options.

If the chip select hold option is enabled, the chip select will not toggle between two consecutive accesses. Therefore, the SPIDELAY[23-16] T2CDELAY of the first transfer and the SPIDELAY[31-24] C2TDELAY of the second transfer will not be applied. However, the wait delay could still be applied between the two transactions, if the SPIDAT1[26] WDEL bit is set to '1h'.

When the SPIDAT1[28] CSHOLD bit is '0h', during the data transmission, the value of the SPIDAT1[23-16] CSNR field is put on the chip select SPIm_SCSnx pins. When the transmission finishes, the value of the SPIDEF[1-0] CSDEF field is put on the SPIm_SCSnx pins.

The current and previous values of the SPIDAT1[28] CSHOLD bit are retained. Although the current value of the SPIDAT1[28] CSHOLD bit is initialized to '0h' when the SPIGCR0[0] RESET bit is set to '0h', the previous value of the SPIDAT1[28] CSHOLD bit is not initialized. The previous value of the SPIDAT1[28] CSHOLD bit must be explicitly initialized by writing twice to the SPIDAT1[28] CSHOLD bit.

11.16.4.3 Slave Mode Settings

The two slave mode options are defined by the configuration bit settings listed in Table 11-3374. Other configuration bits may take any value in the range listed in Table 11-3375. The values listed in Table 11-3374 and Table 11-3375 should not be changed while the SPIGCR1[24] ENABLE bit is set to '1h'. For complete details on each mode, see Section 11.16.4.2.1, *SPI Operation: 3-Pin Mode*, Section 11.16.4.2.2 *SPI Operation: 4-Pin Mode*.

Table 11-3374. SPI Register Settings Defining Slave Modes

Register	Bit(s)	Slave 3-pin	Slave 4-pin Chip Select
SPIGCR0	RESET	1h	1h
SPIGCR1	ENABLE	1h	1h
SPIGCR1	CLKMOD	0h	0h
SPIGCR1	MASTER	0h	0h
SPIPC0	SOMIFUN	1h	1h
SPIPC0	SIMOFUN	1h	1h
SPIPC0	CLKFUN	1h	1h
SPIPC0	SCSFUN	0h	1h, 2h

Table 11-3375. Allowed SPI Register Settings in Slave Modes

Register	Bit(s)	Slave 3-pin	Slave 4-pin Chip Select
SPIFMTn	CHARLEN	2h to 10h	2h to 10h
SPIFMTn	SHIFTDIR	-h	-h

11.16.4.4 Clock

The SPI clock signal (SPIm_CLK) is derived from the SPI functional clock (SPI_VBUSP_CLK) divided by 2. The maximum clock rate supported is SPI_VBUSP_CLK / 2, as determined by the SPIFMTn[15-8] PRESCALE field. The SPIm_CLK frequency is calculated as:

SPIm_CLK frequency = SPI_VBUSP_CLK / [SPIFMTn[15-8] PRESCALE + 1]

When SPIFMTn[15-8] PRESCALE is set to '0h', the SPIm_CLK frequency defaults to SPI_VBUSP_CLK / 2.

11.16.4.5 Interrupt Support

The SPI module generates two interrupt requests. The SPI interrupt system is controlled by three registers:

- The SPI interrupt level register (SPILVL) controls which interrupt events to be mapped to INT0 and INT1 interrupt request lines.
- The SPI interrupt register (SPIINT0) contains bits to selectively enable/disable each interrupt event.
- The SPI flag register (SPIFLG) contains flags indicating when each of the interrupt conditions have occurred.

Multiple interrupt sources can be assigned to the same interrupt request line. To identify the interrupt source in the SPI peripheral the SPIFLG register or the INTVECn[5-1] INTVECTn (n = 0 to 1) field must be read. For more details, see SPI_INTVEC0 and SPI_INTVEC1 registers.

11.16.4.6 DMA Events Support

If handling the SPI message traffic on a character-by-character basis requires too much CPU overhead, then the CPU can configure the system DMA to handle the SPI data transfer.

The SPI module has two DMA synchronization event outputs for receive (REVT) and transmit (XEVT), allowing DMA transfers to be triggered by SPI read receive and write transmit events. The SPI module enables DMA requests by enabling the SPIINT0[16] DMAREQEN bit.

When a character is to be transmitted the SPI module signals the DMA via the XEVT signal. The DMA controller then transfers the data from the source buffer into the SPI transmit data register (SPIDAT1). When a character is received, the SPI module signals the DMA via the REVT signal. The DMA controller then reads the data from the SPI receive buffer register (SPIBUF) and transfers it to a destination buffer for read access.

In most cases, if the DMA is being used to service received data from the SPI module, the SPIINT0[8] RXINTEN bit should be set to '0h'. This prevents the CPU from responding to the received data in addition to the DMA.

11.16.4.7 Robustness Features

The SPI module includes many features to make the SPI communication link robust. An internal loopback test mode can be used to facilitate a power on self test routine. Additionally, the SPI master continually monitors the bus for faults on its data line. Section 11.16.4.7.1, *SPI Internal Loopback Test Mode (Master Only)* and Section 11.16.4.7.2, *SPI Transmission Continuous Self-Test* describe these robustness features in more detail.

11.16.4.7.1 SPI Internal Loopback Test Mode (Master Only)

CAUTION

The internal loop-back self-test mode should not be entered during a normal data transaction or unpredictable operation may occur.

To select the loopback mode, the SPIm_CLK, SPIm_SOMI, SPIm_SIMO pins should be configured as functional pins by configuring the SPIPC0 register and by setting the SPIGCR1[16] LOOPBACK bit. The internal loop-back self-test mode can be used to test the SPI transmit and receive paths including the transmit and receive buffers. In this mode, the transmit signal is internally fed back to the receiver and the SPIm_SIMO, SPIm_SOMI, and SPIm_CLK pins are in a high-impedance state. This mode allows the CPU to write into the transmit buffer, and check that the receive buffer contains the correct transmit data. If an error occurs the corresponding error is set within the status field.

11.16.4.7.2 SPI Transmission Continuous Self-Test

During a data transfer, the SPI module captures the value from its data output pin on the appropriate SPIm_CLK signal edge. This value is compared against the expected value and any difference indicates a fault on the SPI bus. If a fault is detected, then the SPIBUF[28] BITERR and the SPIFLG[4] BITERRFLG bits are set and an error interrupt is generated if enabled. The SPI continuous self-test mode is not available in SPI loopback mode.

11.16.4.8 Reset Considerations

This section describes the software and hardware reset considerations.

11.16.4.8.1 Software Reset Considerations

The SPI module contains a software reset bit (SPIGCR0[0] RESET) that is used to reset the SPI module. As a result of a reset, the SPI module register values go to their reset state. The SPIGCR0[0] RESET bit must be set before beginning of any SPI operation.

11.16.4.8.2 Hardware Reset Considerations

In the event of a hardware reset, the SPI module register values go to their reset state and the application software needs to reprogram the registers to the desired values.

11.16.4.9 Power Management

The SPI module can be put in local low-power mode. The SPI local low-power mode is activated by setting the SPIGCR1[8] POWERDOWN bit. Setting this bit stops the clocks to the SPI internal logic and the SPI registers. Setting the SPIGCR1[8] POWERDOWN bit causes the SPI module to enter local low-power mode and clearing the SPIGCR1[8] POWERDOWN bit causes SPI module to exit from local low-power mode. All registers are accessible during local power-down mode as any register access enables the clock to SPI module for that particular access alone.

Because entering a low-power mode has the effect of suspending all state machine activities, care must be taken when entering such mode to ensure that a valid state is entered when low-power mode is active. As a result, application software must ensure that a low-power mode is not entered during a transmission or reception of data.

11.16.4.10 Emulation Considerations

CAUTION

Viewing or otherwise reading the following SPI registers: SPIBUF, SPIFLG, SPI_INTVEC0, and SPI_INTVEC1 through the JTAG debugger will cause their contents to change, possibly invalidating the results of the debug session. Be sure to set up the debugger to avoid reading these registers.

The SPI module does not support soft or hard stop during emulation breakpoints. The SPI module will continue to run if an emulation breakpoint is encountered.

In addition, any status registers that are cleared after reading will be affected if viewed in a memory or watch window of the debugger because the emulator will read these registers to update the value displayed in the window.

11.16.5 SPI Programming Guide

11.16.5.1 Global Initialization

Table 11-3376 shows the basic steps needed for configuration of the SPI module in master or slave mode.

Table 11-3376. SPI Master and Slave Mode Initialization

Step	Register/Bit Field/Programming Model	Value	
		Master Mode	Slave Mode
1. Reset the SPI module	SPIGCR0[0] RESET	0h	0h
2. Take the SPI module out of reset	SPIGCR0[0] RESET	1h	1h
3. Configure the SPI module for master or slave mode	SPIGCR1[0] MASTER	1h	0h
	SPIGCR1[1] CLKMOD	1h	0h
4. Configure the SPI module for 3-pin or 4-pin with chip select mode	SPIPC0[11] SOMIFUN	1h	1h
	SPIPC0[10] SIMOFUN	1h	1h
	SPIPC0[9] CLKFUN	1h	1h
	SPIPC0[1-0] SCSFUN	0h for Master 3-pin mode	0h for Slave 3-pin mode
		1h, 2h, 3h for Master 4-pin with Chip Select	1h, 2h for Slave 4-pin with Chip Select
5. Choose the SPI data format register n (SPIFMTn) to be used	SPIDAT1[25-24] DFSEL	0h - SPIFMT0 is used	0h - SPIFMT0 is used
		1h - SPIFMT1 is used	1h - SPIFMT1 is used
		2h - SPIFMT2 is used	2h - SPIFMT2 is used
		3h - SPIFMT3 is used	3h - SPIFMT3 is used
6. Configure the SPI data rate, character length, shift direction, phase, polarity and other format options using SPIFMTn selected in step 5.	SPIFMTn	-h	-h[(1)]
7. In master mode, configure the master delay options	SPIDELAY	-h	NA
8. Select the error interrupt notifications	SPIINT0	-h	-h
	SPILVL	-h	-h
9. Enable the SPI communication	SPIGCR1[24] ENABLE	1h	1h
10. Setup and enable the DMA for SPI data handling and then enable the DMA servicing for the SPI data requests	SPIINT0[16] DMAREQEN	1h	1h
11. Handle SPI data transfer requests using DMA and service any SPI error conditions using the interrupt service routine.			

(1) In slave mode configure character length and shift direction only.

11.16.6 SPI Registers

Table 11-3378 lists the memory-mapped registers for the SPI module. All register offset addresses not listed in Table 11-3378 should be considered as reserved locations and the register contents should not be modified.

Table 11-3377. SPI Instances

Instance	Base Address
SPI0	2180 5400h
SPI1	2180 5800h
SPI2	2180 5C00h
SPI3	2180 6000h

Table 11-3378. SPI Registers

Offset	Acronym	Register Name	SPI0 Physical Address	SPI1 Physical Address	SPI2 Physical Address	SPI3 Physical Address	Section
0h	SPIGCR0	SPI Global Control Register 0	2180 5400h	2180 5800h	2180 5C00h	2180 6000h	Section 11.16.6.1
4h	SPIGCR1	SPI Global Control Register 1	2180 5404h	2180 5804h	2180 5C04h	2180 6004h	Section 11.16.6.2
8h	SPIINT0	SPI Interrupt Register	2180 5408h	2180 5808h	2180 5C08h	2180 6008h	Section 11.16.6.3
Ch	SPILVL	SPI Interrupt Level Register	2180 540Ch	2180 580Ch	2180 5C0Ch	2180 600Ch	Section 11.16.6.4
10h	SPIFLG	SPI Flag Register	2180 5410h	2180 5810h	2180 5C10h	2180 6010h	Section 11.16.6.5
14h	SPIPC0	SPI Pin Control Register 0 (Function)	2180 5414h	2180 5814h	2180 5C14h	2180 6014h	Section 11.16.6.6
38h	SPIDAT0	SPI Data Transmit Register 0	2180 5438h	2180 5838h	2180 5C38h	2180 6038h	Section 11.16.6.7
3Ch	SPIDAT1	SPI Data Transmit Register 1 (Data Transmit and Format Select)	2180 543Ch	2180 583Ch	2180 5C3Ch	2180 603Ch	Section 11.16.6.8
40h	SPIBUF	SPI Receive Buffer Register	2180 5440h	2180 5840h	2180 5C40h	2180 6040h	Section 11.16.6.9
44h	SPIEMU	SPI Receive Emulation Register	2180 5444h	2180 5844h	2180 5C44h	2180 6044h	Section 11.16.6.10
48h	SPIDELAY	SPI Delay Register	2180 5448h	2180 5848h	2180 5C48h	2180 6048h	Section 11.16.6.11
4Ch	SPIDEF	SPI Default Chip Select Register	2180 544Ch	2180 584Ch	2180 5C4Ch	2180 604Ch	Section 11.16.6.12
50h to 5Ch	SPIFMT0 to SPIFMT3	SPI Data Format Registers	2180 5450h to 2180 545Ch	2180 5850h to 2180 585Ch	2180 5C50h to 2180 5C5Ch	2180 6050h to 2180 605Ch	Section 11.16.6.13
60h	SPI_INTVEC0	SPI Interrupt Vector Register 0	2180 5460h	2180 5860h	2180 5C60h	2180 6060h	Section 11.16.6.14
64h	SPI_INTVEC1	SPI Interrupt Vector Register 1	2180 5464h	2180 5864h	2180 5C64h	2180 6064h	Section 11.16.6.15
1FCh	SPIREV	SPI Revision ID Register	2180 55FCh	2180 59FCh	2180 5DFCh	2180 61FCh	Section 11.16.6.16

11.16.6.1 SPIGCR0 Register (Offset = 0h) [reset = 0h]

The SPI Global Control Register 0 (SPIGCR0) is shown in Figure 11-1376 and described in Table 11-3380.

Table 11-3379. SPIGCR0 Instances

Instance	Physical Address
SPI0	2180 5400h
SPI1	2180 5800h
SPI2	2180 5C00h
SPI3	2180 6000h

Figure 11-1376. SPIGCR0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RESET
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3380. SPIGCR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	RESET	R/W	0h	Reset bit for the module. This bit needs to be set to 1 before any operation on SPI can be done. • 0 = SPI is in reset state. • 1 = SPI is out of reset state.

Table 11-3381. Register Call Summary for SPIGCR0

SPI Registers • SPI Registers: [0] • SPIGCR0 Register (Offset = 0h) [reset = 0h]: [0]
SPI Functional Description • Slave Mode Settings: [0] • Master Mode Settings: [0] • Software Reset Considerations: [0][1] • Master Mode Timing Options: [0]
SPI Programming Guide • Global Initialization: [0][1]

11.16.6.2 SPIGCR1 Register (Offset = 4h) [reset = 0h]

The SPI Global Control Register 1 (SPIGCR1) is shown in Figure 11-1377 and described in Table 11-3383.

Table 11-3382. SPIGCR1 Instances

Instance	Physical Address
SPI0	2180 5404h
SPI1	2180 5804h
SPI2	2180 5C04h
SPI3	2180 6004h

Figure 11-1377. SPIGCR1 Register

31	30	29	28	27	26	25	24
RESERVED							ENABLE
R-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED							LOOPBACK
R-0h							R/W-0h
15	14	13	12	11	10	9	8
RESERVED							POWERDOWN
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED						CLKMOD	MASTER
R-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3383. SPIGCR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24	ENABLE	R/W	0h	SPI enable. This bit enables the SPI transfers. The other SPI configuration registers except SPIINT0[16] DMAREQEN bit should be configured before writing 1 to this bit. This will prevent the SPI from responding to bus operations erroneously while it is in the process of being configured. The SPIINT0[16] DMAREQEN bit should be enabled after setting SPIGCR1[24] ENABLE bit. If SPIINT0[16] DMAREQEN bit is enabled before setting SPIGCR1[24] ENABLE bit then the first DMA request that occurs before the SPI is ready for data transfer may get dropped. When SPIGCR1[24] ENABLE bit is set to 0h, the following SPI registers get forced to their default states (to 0s except for SPIBUF[31] RXEMPTY bit): • Both TX and RX shift registers. • The SPIDAT0[15-0] TXDATA and SPIDAT1[15-0] TXDATA fields. • All the fields of the SPIFLG register. • Contents of SPIBUF and the internal RXBUF registers. • 0 = SPI is not activated for transfers. • 1 = Activates SPI.
23-17	RESERVED	R	0h	Reserved

Table 11-3383. SPIGCR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	LOOPBACK	R/W	0h	Internal loop-back test mode. The internal self-test option can be enabled by setting this bit. If the SPIm_SIMO and SPIm_SOMI pins are configured with SPI functionality, then the SPIm_SIMO pin is internally connected to the SPIm_SOMI pin. The transmit data is looped back as receive data and is stored in the receive field of the concerned buffer. Externally, during loop-back operation, the SPIm_CLK pin outputs an inactive value, SPIm_SIMO and SPIm_SOMI pins remain in high-impedance state. The SPI has to be initialized in master mode before the loop-back can be selected. If the SPI is initialized in slave mode or a data transfer is ongoing, errors may result. • 0 = Internal loop-back test mode disabled. • 1 = Internal loop-back test mode enabled.
15-9	RESERVED	R	0h	Reserved
8	POWERDOWN	R/W	0h	When active, the SPI state machine enters a power-down state. • 0 = The SPI is in active mode. • 1 = The SPI is in power-down mode.
7-2	RESERVED	R	0h	Reserved
1	CLKMOD(1)	R/W	0h	Clock mode. Selects either an internal or external clock source. This bit also determines the I/O direction of the chip select (SPIm_SCSnx) pins in functional mode. • 0 = Clock is external, SPIm_SCSnx is an input. • 1 = Clock is internal, SPIm_SCSnx is an output.
0	MASTER(1)	R/W	0h	SPIm_SIMO/SPIm_SOMI pin direction determination. Determines the direction of the SPIm_SIMO and SPIm_SOMI pins. This bit determines whether the SPI is in Master mode or Slave mode. • 0 = SPIm_SIMO pin an input, SPIm_SOMI pin an output. • 1 = SPIm_SOMI pin an input, SPIm_SIMO pin an output.

(1) Although there are two different bits which control the Master/Slave mode functions, only two of their combinations are valid. For Master mode of operation: MASTER = '1', CLKMOD = '1'. For Slave mode of operation: MASTER = '0', CLKMOD = '0'. Any other combinations of these two bits may not yield any desirable operation of the module and must be avoided.

Table 11-3384. Register Call Summary for SPIGCR1

SPI Registers • SPI Registers: [0] • SPIDAT0 Register (Offset = 38h) [reset = 0h]: [0][1] • SPIINT0 Register (Offset = 8h) [reset = 0h]: [0] • SPIGCR1 Register (Offset = 4h) [reset = 0h]: [0][1][2][3] • SPIFLG Register (Offset = 10h) [reset = 0h]: [0][1][2] • SPIDAT1 Register (Offset = 3Ch) [reset = 0h]: [0][1]
SPI Functional Description • SPI Operation: 4-Pin Mode: [0] • SPI Operation: 3-Pin Mode: [0][1][2][3] • Operation Modes: [0][1] • SPI Internal Loopback Test Mode (Master Only): [0] • Slave Mode Settings: [0][1][2][3] • Power Management: [0][1][2] • Master Mode Settings: [0][1][2][3][4]
SPI Programming Guide • Global Initialization: [0][1][2]

11.16.6.3 SPIINT0 Register (Offset = 8h) [reset = 0h]

The SPI Interrupt Register (SPIINT0) is shown in Figure 11-1378 and described in Table 11-3386.

Table 11-3385. SPIINT0 Instances

Instance	Physical Address
SPI0	2180 5408h
SPI1	2180 5808h
SPI2	2180 5C08h
SPI3	2180 6008h

Figure 11-1378. SPIINT0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							DMAREQEN
R-0h							R/W-0h
15	14	13	12	11	10	9	8
RESERVED						TXINTENA	RXINTENA
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	OVRNINTENA	RESERVED	BITERRENA	RESERVED			
R-0h	R/W-0h	R-0h	R/W-0h	R-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3386. SPIINT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	DMAREQEN	R/W	0h	DMA request enable. Enables the DMA request signal to be generated for both receive and transmit channels. Set SPIINT0[16] DMAREQEN bit only after setting the SPIGCR1[24] ENABLE bit to 1. • 0 = DMA is not used. • 1 = DMA requests will be generated. • **Note:** A transmit DMA request will be generated each time a transmit data is copied to the shift register either from TXBUF or directly from SPIDAT0/SPIDAT1 register. • **Note:** A receive DMA request will be generated each time a received data is copied to SPIBUF register either from RXBUF or directly from the shift register.
15-10	RESERVED	R	0h	Reserved
9	TXINTENA	R/W	0h	An interrupt is to be generated every time data is written to the shift register, so that a new data can be written to TXBUF. Setting this bit will generate an interrupt if the SPIFLG[9] TXINTFLG bit is set to 1. • 0 = No interrupt will be generated upon SPIFLG[9] TXINTFLG bit being set to 1. • 1 = Interrupt will be generated upon SPIFLG[9] TXINTFLG bit being set to 1.
8	RXINTENA	R/W	0h	Receive interrupt enable. An interrupt is to be generated when the SPIFLG[8] RXINTFLG bit is set. • 0 = Interrupt will not be generated. • 1 = Interrupt will be generated.
7	RESERVED	R	0h	Reserved

Table 11-3386. SPIINT0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	OVRNINTENA	R/W	0h	Overrun interrupt enable. An interrupt is to be generated when the SPIFLG[6] OVRNINTFLG bit is set. • 0 = Overrun interrupt will not be generated. • 1 = Overrun interrupt will be generated.
5	RESERVED	R	0h	Reserved
4	BITERRENA	R/W	0h	Enables interrupt on bit error. An interrupt is to be generated when the SPIFLG[4] BITERRFLG bit is set. • 0 = No interrupt asserted upon bit error. • 1 = Enables an interrupt on a bit error.
3-0	RESERVED	R	0h	Reserved

Table 11-3387. Register Call Summary for SPIINT0

SPI Registers • SPIGCR1 Register (Offset = 4h) [reset = 0h]: [0][1][2] • SPI_INTVEC0 Register (Offset = 60h) [reset = 0h]: [0] • SPI Registers: [0] • SPI_INTVEC1 Register (Offset = 64h) [reset = 0h]: [0] • SPIINT0 Register (Offset = 8h) [reset = 0h]: [0][1]
SPI Functional Description • Interrupt Support: [0] • DMA Events Support: [0][1]
SPI Programming Guide • Global Initialization: [0][1]

11.16.6.4 SPILVL Register (Offset = Ch) [reset = 0h]

The SPI Interrupt Level Register (SPILVL) is shown in Figure 11-1379 and described in Table 11-3389.

Table 11-3388. SPILVL Instances

Instance	Physical Address
SPI0	2180 540Ch
SPI1	2180 580Ch
SPI2	2180 5C0Ch
SPI3	2180 600Ch

Figure 11-1379. SPILVL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						TXINTLVL	RXINTLVL
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	OVRNINTLVL	RESERVED	BITERRLVL	RESERVED			
R-0h	R/W-0h	R-0h	R/W-0h	R-0h			

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3389. SPILVL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	TXINTLVL	R/W	0h	Transmit interrupt level. • 0 = Transmit interrupt is mapped to interrupt line INT0. • 1 = Transmit interrupt is mapped to interrupt line INT1.
8	RXINTLVL	R/W	0h	Receive interrupt level. • 0 = Receive interrupt is mapped to interrupt line INT0. • 1 = Receive interrupt is mapped to interrupt line INT1.
7	RESERVED	R	0h	Reserved
6	OVRNINTLVL	R/W	0h	Receive overrun interrupt level. • 0 = Receive overrun interrupt is mapped to interrupt line INT0. • 1 = Receive overrun interrupt is mapped to interrupt line INT1.
5	RESERVED	R	0h	Reserved
4	BITERRLVL	R/W	0h	Bit error interrupt level. • 0 = Bit error interrupt is mapped to interrupt line INT0. • 1 = Bit error interrupt is mapped to interrupt line INT1.
3-0	RESERVED	R	0h	Reserved

Table 11-3390. Register Call Summary for SPILVL

SPI Registers • SPI Registers: [0] • SPILVL Register (Offset = Ch) [reset = 0h]: [0]
SPI Functional Description • Interrupt Support: [0]

Table 11-3390. Register Call Summary for SPILVL (continued)

SPI Programming Guide
• Global Initialization: [0]

11.16.6.5 SPIFLG Register (Offset = 10h) [reset = 0h]

The SPI Flag Register (SPIFLG) is shown in Figure 11-1380 and described in Table 11-3392.

Table 11-3391. SPIFLG Instances

Instance	Physical Address
SPI0	2180 5410h
SPI1	2180 5810h
SPI2	2180 5C10h
SPI3	2180 6010h

Figure 11-1380. SPIFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						TXINTFLG	RXINTFLG
R-0h						R-0h	R/W1C-0h
7	6	5	4	3	2	1	0
RESERVED	OVRNINTFLG	RESERVED	BITERRFLG	RESERVED			
R-0h	R/W1C-0h	R-0h	R/W1C-0h	R-0h			

LEGEND: R = Read Only; R/W1C = Read/Write 1 to Clear Bit; -*n* = value after reset

Table 11-3392. SPIFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	TXINTFLG	R	0h	Transmitter empty interrupt flag. Serves as an interrupt flag indicating that the transmit buffer (TXBUF) is empty and a new data can be written to it. This flag is set when a data is copied to the shift register either directly or from the TXBUF register. This bit is cleared by one of following ways: • Writing a new data to either SPIDAT0 or SPIDAT1 registers. • Writing a 0 to SPIGCR1[24] ENABLE bit. • 0 = Transmit buffer is now full. No interrupt pending for transmitter empty. • 1 = Transmit buffer is empty. An interrupt is pending to fill the transmitter.

Table 11-3392. SPIFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	RXINTFLG	R/W1C	0h	Receiver full interrupt flag. This flag is set when a word is received and copied into the buffer register (SPIBUF). This bit is cleared under the following ways: • Reading the SPIBUF register. During emulation mode, however, a read to the emulation register (SPIEMU) does not clear this flag bit. • Reading SPI_INTVEC0 or SPI_INTVEC1 register when there is a receive buffer full interrupt. • Writing a 1 to this bit. • Writing a 0 to SPIGCR1[24] ENABLE bit. • System reset. • 0 = No new received data pending. Receive buffer is empty. • 1 = A newly received data is ready to be read. Receive buffer is full. • **Note:** Clearing SPIFLG[8] RXINTFLG bit by writing a 1 before reading the SPIBUF register sets the SPIBUF[31] RXEMPTY bit too. This way, one can ignore a received data. However, if the internal RXBUF is already full, the data from RXBUF will be copied to SPIBUF register and the SPIBUF[31] RXEMPTY bit will be cleared again. The SPIBUF register contents should be read first if this situation needs to be avoided.
7	RESERVED	R	0h	Reserved
6	OVRNINTFLG	R/W1C	0h	Receiver overrun flag. The bit is set when a receive operation completes before the previous character has been read from the receive buffer. The bit indicates that the last received character has been overwritten and therefore lost. This bit is cleared under the following conditions: • Reading SPI_INTVEC0 or SPI_INTVEC1 register when there is a receive buffer overrun interrupt. • Writing a 1 to this bit. • 0 = Overrun condition did not occur. • 1 = Overrun condition has occurred. • **Note:** Reading SPIBUF register does not clear the SPIFLG[6] OVRNINTFLG bit. If an overrun interrupt is detected, then the SPIBUF register may need to be read twice to get to the overrun buffer. This is due to the fact that the overrun will always occur to the internal RXBUF. Each read to the SPIBUF register will result in RXBUF contents (if it is full) getting copied to SPIBUF register. • **Note:** A special condition under which SPIFLG[6] OVRNINTFLG bit gets set. If both SPIBUF register and RXBUF are already full and while another buffer receive is underway, if any errors like TIMEOUT, BITERR and DLENERR occur, then SPIFLG[6] OVRNINTFLG bit will be set to indicate that the status flags are getting overwritten by the new transfer. This overrun should be treated like a normal receiver overrun.
5	RESERVED	R	0h	Reserved
4	BITERRFLG	R/W1C	0h	This bit is set when a mismatch of internal transmit data and transmitted data is detected. The SPI samples the signal of the transmit pin (master: SPIm_SIMO, slave: SPIm_SOMI) at the receive point (half clock cycle after transmit point). If the sampled value differs from the transmitted value a bit error is detected and the flag is set. A possible reason for a bit error can be a too high bit rate/capacitive load or another master/slave trying to transmit at the same time. This flag can be cleared by one of the following ways: • Write a 1 to this bit. • Set SPIGCR1[24] ENABLE bit to 0. • 0 = No bit error occurred. • 1 = A bit error occurred.
3-0	RESERVED	R	0h	Reserved

Table 11-3393. Register Call Summary for SPIFLG

SPI Registers • SPI_INTVEC0 Register (Offset = 60h) [reset = 0h]: [0][1][2][3] • SPI Registers: [0] • SPI_INTVEC1 Register (Offset = 64h) [reset = 0h]: [0][1][2][3] • SPIBUF Register (Offset = 40h) [reset = 80000000h]: [0][1][2] • SPIINT0 Register (Offset = 8h) [reset = 0h]: [0][1][2][3][4][5] • SPIGCR1 Register (Offset = 4h) [reset = 0h]: [0] • SPIFLG Register (Offset = 10h) [reset = 0h]: [0][1][2][3][4]
SPI Functional Description • SPI Transmission Continuous Self-Test: [0] • Emulation Considerations: [0] • Interrupt Support: [0][1]

11.16.6.6 SPIPC0 Register (Offset = 14h) [reset = 0h]

The SPI Pin Control Register 0 (SPIPC0) is shown in Figure 11-1381 and described in Table 11-3395.

Table 11-3394. SPIPC0 Instances

Instance	Physical Address
SPI0	2180 5414h
SPI1	2180 5814h
SPI2	2180 5C14h
SPI3	2180 6014h

Figure 11-1381. SPIPC0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				SOMIFUN	SIMOFUN	CLKFUN	RESERVED
R-0h				R/W-0h	R/W-0h	R/W-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED						SCSFUN	
R-0h						R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3395. SPIPC0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	SOMIFUN	R/W	0h	Slave out, master in pin function. This bit determines whether the SPIm_SOMI pin is Reserved or a SPI functional pin. • 0 = SPIm_SOMI pin is reserved. • 1 = SPIm_SOMI pin is a SPI functional pin.
10	SIMOFUN	R/W	0h	Slave in, master out pin function. This bit determines whether the SPIm_SIMO pin is Reserved or a SPI functional pin. • 0 = SPIm_SIMO pin is reserved. • 1 = SPIm_SIMO pin is a SPI functional pin.
9	CLKFUN	R/W	0h	SPI clock pin function. This bit determines whether the SPIm_CLK pin is Reserved or a SPI functional pin. • 0 = SPIm_CLK pin is reserved. • 1 = SPIm_CLK pin is a SPI functional pin.
8-2	RESERVED	R	0h	Reserved
1-0	SCSFUN	R/W	0h	SPI chip select pin function. The bit in this field determines whether the SPIm_SCSnx pin is Reserved or a SPI functional pin. For more information about supported number of the chip select pins, see Section 11.16.1, *SPI Overview*. • 0 = The corresponding SPIm_SCSnx pin is Reserved. • 1 = The corresponding SPIm_SCSnx pin is a SPI functional pin. • **Note:** In slave mode, only one chip select pin can be used.

Table 11-3396. Register Call Summary for SPIPC0

SPI Registers • SPIPC0 Register (Offset = 14h) [reset = 0h]: [0] • SPI Registers: [0]
SPI Functional Description • Slave Mode Settings: [0][1][2][3] • SPI Internal Loopback Test Mode (Master Only): [0] • Multiple Chip Selects: [0] • Master Mode Settings: [0][1][2][4]
SPI Programming Guide • Global Initialization: [0][1][2][3]

11.16.6.7 SPIDAT0 Register (Offset = 38h) [reset = 0h]

The SPI Transmit Data Register 0 (SPIDAT0) is shown in Figure 11-1382 and described in Table 11-3398.

Table 11-3397. SPIDAT0 Instances

Instance	Physical Address
SPI0	2180 5438h
SPI1	2180 5838h
SPI2	2180 5C38h
SPI3	2180 6038h

Figure 11-1382. SPIDAT0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																TXDATA															
R-0h																R/W-0h															

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3398. SPIDAT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	TXDATA	R/W	0h	SPI transmit data (value = 0-FFFFh). When written, these bits will be copied to the shift register if it is empty. If the shift register is not empty, the TXBUF will hold the written values. SPIGCR1[24] ENABLE bit must be set to 1 before this register can be written to. Writing a 0 to the SPIGCR1[24] ENABLE bit forces the SPIDAT0[15-0] TXDATA field to 0. • **Note:** Irrespective of the character length, the transmit data should be right-justified before writing to SPIDAT0 register. • **Note:** The default data format control register for SPIDAT0 is SPIFMT0 register. However, it is possible to reprogram the SPIDAT1[25-24] DFSEL field before using SPIDAT0 register, to select a different SPIFMTn register.

Table 11-3399. Register Call Summary for SPIDAT0

SPI Registers • SPI_INTVEC0 Register (Offset = 60h) [reset = 0h]: [0] • SPI Registers: [0] • SPIDAT0 Register (Offset = 38h) [reset = 0h]: [0][1][2][3][4] • SPIEMU Register (Offset = 44h) [reset = 80000000h]: [0][1] • SPIBUF Register (Offset = 40h) [reset = 80000000h]: [0][1][2][3] • SPIINT0 Register (Offset = 8h) [reset = 0h]: [0] • SPIGCR1 Register (Offset = 4h) [reset = 0h]: [0] • SPIFLG Register (Offset = 10h) [reset = 0h]: [0] • SPI_INTVEC1 Register (Offset = 64h) [reset = 0h]: [0]
SPI Functional Description • SPI Operation: 4-Pin Mode: [0] • SPI Operation: 3-Pin Mode: [0]
Serial Peripheral Interface (SPI) • SPI Overview: [0]

11.16.6.8 SPIDAT1 Register (Offset = 3Ch) [reset = 0h]

The SPI Transmit Data Register (SPIDAT1) is shown in Figure 11-1383 and described in Table 11-3401.

Table 11-3400. SPIDAT1 Instances

Instance	Physical Address
SPI0	2180 543Ch
SPI1	2180 583Ch
SPI2	2180 5C3Ch
SPI3	2180 603Ch

Figure 11-1383. SPIDAT1 Register

31	30	29	28	27	26	25	24
RESERVED			CSHOLD	RESERVED	WDEL	DFSEL	
R-0h			R/W-0h	R-0h	R/W-0h	R/W-0h	
23	22	21	20	19	18	17	16
CSNR							
R/W-0h							
15	14	13	12	11	10	9	8
TXDATA							
R/W-0h							
7	6	5	4	3	2	1	0
TXDATA							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3401. SPIDAT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved
28	CSHOLD	R/W	0h	Chip select hold mode. The SPIDAT1[28] CSHOLD bit is supported in master mode only. In slave mode, this bit is ignored. SPIDAT1[28] CSHOLD bit defines the behavior of the chip select lines at the end of a data transfer. • 0 = The SPIm_SCSnx pins are restored to SPIDEF[1-0] CSDEF field at the end of a transfer after the T2CDELAY time has passed. • 1 = The SPIm_SCSnx pins are continued as SPIDAT1[23-16] CSNR field at the end of a transfer until a control field with new data and control information is loaded into SPIDAT1 register. The setup and hold delays in SPIDELAY register (SPIDELAY[31-24] C2TDELAY and SPIDELAY[23-16] T2CDELAY fields) are not applied between transactions which have SPIDAT1[28] CSHOLD = 1. • **Note:** If a transfer with SPIDAT1[28] CSHOLD = 0 is followed by a transfer with SPIDAT1[28] CSHOLD = 1, then SPIDELAY[31-24] C2TDELAY field will be applied at the beginning of the second transfer. To do multiple transactions to a slave without deactivating chip select between transfers, maintain SPIDAT1[28] CSHOLD of 1 and same SPIDAT1[23-16] CSNR field information to keep the same slave selected in the transfers.
27	RESERVED	R	0h	Reserved

Table 11-3401. SPIDAT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	WDEL	R/W	0h	Enable the delay counter at the end of the current transaction. The SPIDAT1[26] WDEL bit is supported in master mode only. In slave mode, this bit is ignored. • 0 = No delay will be inserted. However, SPIm_SCSnx pin will still be deactivated for at least 2 SPI module clock cycles if SPIDAT1[28] CSHOLD = 0. • 1 = After a transaction, SPIFMTn[29-24] WDELAY field of the selected data format will be loaded into the delay counter. No transaction will be performed until the SPIFMTn[29-24] WDELAY counter overflows. The SPIm_SCSnx pin will be deactivated for at least (WDELAY + 2) × SPI module clock period.
25-24	DFSEL	R/W	0h	Data word format select: • 0h = Data word format 0 is selected. • 1h = Data word format 1 is selected. • 2h = Data word format 2 is selected. • 3h = Data word format 3 is selected.
23-16	CSNR	R/W	0h	Chip select number (value = 0-FFh). The SPIDAT1[23-16] CSNR field defines the chip select that will be activated during the data transfer. In slave mode, this field must be written as 00h; in master mode, the value of this field is driven on the chip select pins. Bits [23-18] are not used. Bits [17-16] are put out on SPIm_SCSn[1-0] during a transfer. For more information about supported number of the chip select pins, see Section 11.16.1, *SPI Overview*.
15-0	TXDATA	R/W	0h	Transfer data (value = 0-FFFFh). When written, these bits will be copied to the shift register if it is empty. If the shift register is not empty, the TXBUF will hold the written values. SPIGCR1[24] ENABLE bit must be set to 1 before this register can be written to. Writing a 0 to the SPIGCR1[24] ENABLE bit forces the lower 16 bits of the SPIDAT1 register to 0. • **Note:** Irrespective of the character length, the transmit data should be right-justified before writing to SPIDAT1 register.

Table 11-3402. Register Call Summary for SPIDAT1

SPI Registers • SPI_INTVEC0 Register (Offset = 60h) [reset = 0h]: [0] • SPI Registers: [0] • SPIDAT0 Register (Offset = 38h) [reset = 0h]: [0] • SPIBUF Register (Offset = 40h) [reset = 80000000h]: [0][1][2][3] • SPIINT0 Register (Offset = 8h) [reset = 0h]: [0] • SPIFLG Register (Offset = 10h) [reset = 0h]: [0] • SPIGCR1 Register (Offset = 4h) [reset = 0h]: [0] • SPIFMT0 to SPIFMT3 Registers (Offset = 50h to 5Ch) [reset = 0h]: [0] • SPIDAT1 Register (Offset = 3Ch) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14] • SPI_INTVEC1 Register (Offset = 64h) [reset = 0h]: [0] • SPIEMU Register (Offset = 44h) [reset = 80000000h]: [0][1]
SPI Functional Description • SPI Operation: 4-Pin Mode: [0] • SPI Operation: 3-Pin Mode: [0][1][2][3] • Operation Modes: [0] • Multiple Chip Selects: [0][1] • Master Mode Timing Options: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16] • DMA Events Support: [0]
SPI Programming Guide • Global Initialization: [0]

Table 11-3402. Register Call Summary for SPIDAT1 (continued)

SPI Environment • Clock Phase and Polarity: [0] • Data Formats: [0][1] • Character Length: [0]
Serial Peripheral Interface (SPI) • SPI Overview: [0]

11.16.6.9 SPIBUF Register (Offset = 40h) [reset = 80000000h]

The SPI Receive Buffer register (SPIBUF) is shown in Figure 11-1384 and described in Table 11-3404.

Table 11-3403. SPIBUF Instances

Instance	Physical Address
SPI0	2180 5440h
SPI1	2180 5840h
SPI2	2180 5C40h
SPI3	2180 6040h

Figure 11-1384. SPIBUF Register

31	30	29	28	27	26	25	24
RXEMPTY	RXOVR	TXFULL	BITERR	RESERVED			
RS-1h	RC-0h	R-0h	RC-0h	R-0h			
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RXDATA							
R-0h							
7	6	5	4	3	2	1	0
RXDATA							
R-0h							

LEGEND: R = Read Only; RC = Read to Clear; RS = Read to Set; *-n* = value after reset

Table 11-3404. SPIBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RXEMPTY	RS	1h	Receive data buffer empty. When the host reads the SPIBUF field or the whole SPIBUF register, this will automatically set the SPIBUF[31] RXEMPTY bit. When a data transfer is completed, the received data is copied into SPIBUF register, the SPIBUF[31] RXEMPTY bit is cleared. This flag is set to 1 under following conditions: • Reading the SPIBUF[15-0] RXDATA field. • Writing 1 to clear the SPIFLG[8] RXINTFLG bit. • 0 = New data has been received and copied into the SPIBUF register. • 1 = No data received since last reading of SPIBUF register. Write-clearing the SPIFLG[8] RXINTFLG bit before reading the SPIBUF register indicates the received data is being ignored. Conversely, SPIFLG[8] RXINTFLG bit can be cleared by reading the SPIBUF[15-0] RXDATA field or the entire SPIBUF register.

Table 11-3404. SPIBUF Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
30	RXOVR	RC	0h	Receive data buffer overrun. When a data transfer is completed and the received data is copied into the RXBUF while it is already full, SPIBUF[30] RXOVR bit is set. An overrun always occurs to the RXBUF, and SPIBUF register contents never get overwritten until after it is read by the CPU/DMA. Reading SPIBUF register does not clear the SPIBUF[30] RXOVR bit. If an overrun interrupt is detected, then the SPIBUF register may need to be read twice to get to the overrun buffer. This is due to the fact that the overrun will always occur to the internal RXBUF. Each read to the SPIBUF register will result in RXBUF contents (if it is full) getting copied to SPIBUF register. • **Note:** A special condition under which SPIBUF[30] RXOVR bit gets set. If both SPIBUF register and RXBUF are already full and while another buffer receive is underway, if any errors like TIMEOUT, BITERR and DLENERR occur, then SPIBUF[30] RXOVR bit will be set to indicate that the status flags are getting overwritten by the new transfer. This overrun should be treated like a normal receiver overrun. • 0 = No receive data overrun condition occurred since last time reading the data field. • 1 = A receive data overrun condition occurred since last time reading the data field.
29	TXFULL	R	0h	Transmit data buffer full. This flag is a read-only flag. Writing into SPIDAT0 or SPIDAT1 field while the TX shift register is full will automatically set the SPIBUF[29] TXFULL bit. Once the data is copied to the shift register, the SPIBUF[29] TXFULL bit will be cleared. Writing to the SPIDAT0/SPIDAT1 register when both TXBUF and the TX shift register are empty does not set the SPIBUF[29] TXFULL bit. • 0 = The transmit buffer is empty; SPIDAT0/SPIDAT1 register is ready to accept a new data. • 1 = The transmit buffer is full; SPIDAT0/SPIDAT1 register is not ready to accept a new data.
28	BITERR	RC	0h	Bit error. There was a mismatch of internal transmit data and transmitted data. The SPI samples the signal of the transmit pin (master: SIMO, slave: SOMI) at the receive point (half clock cycle after transmit point). If the sampled value differs from the transmitted value, a bit error is detected and the flag BITERR is set. A possible reason for a bit error can be noise, a too-high bit rate/capacitive load, or another master/slave trying to transmit at the same time. • **Note:** This flag is set to 0h when the SPIBUF[15-0] RXDATA field is read. • 0 = No bit error occurred. • 1 = A bit error occurred.
27-16	RESERVED	R	0h	Reserved
15-0	RXDATA	R	0h	SPI receive data (value = 0-FFFFh). This is the received data, transferred from the receive shift-register at the end of a transfer completion. Irrespective of the programmed character length and the direction of shifting, the received data is stored right-justified in the register.

Table 11-3405. Register Call Summary for SPIBUF

SPI Registers
- SPI Registers: [0]
- SPIEMU Register (Offset = 44h) [reset = 80000000h]: [0][1][2][3]
- SPIBUF Register (Offset = 40h) [reset = 80000000h]: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25]
- SPIINT0 Register (Offset = 8h) [reset = 0h]: [0]
- SPIGCR1 Register (Offset = 4h) [reset = 0h]: [0][1]
- SPIFLG Register (Offset = 10h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9][10][11]

Table 11-3405. Register Call Summary for SPIBUF (continued)

SPI Functional Description • SPI Transmission Continuous Self-Test: [0] • Emulation Considerations: [0] • SPI Operation: 3-Pin Mode: [0][1] • DMA Events Support: [0]
SPI Environment • Character Length: [0]
Serial Peripheral Interface (SPI) • SPI Overview: [0]

11.16.6.10 SPIEMU Register (Offset = 44h) [reset = 80000000h]

The SPI Emulation Register (SPIEMU) is shown in Figure 11-1385 and described in Table 11-3407.

NOTE: All the fields of SPIEMU register are Read-Only. Read operation on this register under any mode will not have any impact on the status of this or any other registers.

Table 11-3406. SPIEMU Instances

Instance	Physical Address
SPI0	2180 5444h
SPI1	2180 5844h
SPI2	2180 5C44h
SPI3	2180 6044h

Figure 11-1385. SPIEMU Register

31	30	29	28	27	26	25	24
RXEMPTY	RXOVR	TXFULL	BITERR	RESERVED			
R-1h	R-0h	R-0h	R-0h	R-0h			
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RXDATA							
R-0h							
7	6	5	4	3	2	1	0
RXDATA							
R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3407. SPIEMU Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RXEMPTY	R	1h	Receive data buffer empty. • 0 = New data has been received and copied into the SPIBUF register. • 1 = No data has been received since the last time the SPIBUF register was read.
30	RXOVR	R	0h	Receive data buffer overrun. • 0 = No receive data overrun condition occurred since last time the data field was read. • 1 = A receive data overrun condition occurred since the last time the data field was read.
29	TXFULL	R	0h	Transmit data buffer full. • 0 = The transmit buffer is empty; SPIDAT0/SPIDAT1 register is ready to accept new data. • 1 = The transmit buffer is full; SPIDAT0/SPIDAT1 register is not ready to accept new data.
28	BITERR	R	0h	Mismatch of internal transmit data and transmitted data. • 0 = No bit error occurred. • 1 = A bit error occurred.
27-16	RESERVED	R	0h	Reserved

Table 11-3407. SPIEMU Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-0	RXDATA	R	0h	SPI receive data (value = 0-FFFFh). SPI emulation is a mirror of the SPIBUF register. The only difference between SPIEMU and SPIBUF registers is that a read from SPIEMU register does not clear any of the status flags.

Table 11-3408. Register Call Summary for SPIEMU

SPI Registers • SPI Registers: [0] • SPIFLG Register (Offset = 10h) [reset = 0h]: [0] • SPIEMU Register (Offset = 44h) [reset = 80000000h]: [0][1][2][3]
Serial Peripheral Interface (SPI) • SPI Overview: [0]

11.16.6.11 SPIDELAY Register (Offset = 48h) [reset = 0h]

The SPI Delay Register (SPIDELAY) is shown in Figure 11-1386 and described in Table 11-3410.

Table 11-3409. SPIDELAY Instances

Instance	Physical Address
SPI0	2180 5448h
SPI1	2180 5848h
SPI2	2180 5C48h
SPI3	2180 6048h

Figure 11-1386. SPIDELAY Register

31	30	29	28	27	26	25	24
C2TDELAY							
R/W-0h							
23	22	21	20	19	18	17	16
T2CDELAY							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3410. SPIDELAY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	C2TDELAY	R/W	0h	Chip-select-active-to-transmit-start delay (value = 0-FFh). SPIDELAY[31-24] C2TDELAY bit is used in master mode only. It defines a setup time for the slave device that delays the data transmission from the chip select active edge by a multiple of SPI module clock cycles. C2TDELAY can be configured between 2 and 257 SPI module clock cycles (See Section 11.16.4.2.3.1.1). The setup time value is calculated as follows: $t_{C2TDELAY}$ = (C2TDELAY + 2) × SPI module clock period • **Note:** If C2TDELAY = 0, then $t_{C2TDELAY}$ = 0. • ***Example:*** SPI module clock = 25 MHz -> SPI module clock period = 40 ns; C2TDELAY = 06h; then $t_{C2TDELAY}$ = 320 ns; When the chip select signal becomes active, the slave has to prepare for data transfer within 320 ns. • **Note:** If phase = 1, the delay between SPIm_SCSnxfalling edge to the first edge of SPIm_SCSnx will have an additional 0.5 SPIm_CLK period delay. This delay is as per the SPI protocol.

Table 11-3410. SPIDELAY Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-16	T2CDELAY	R/W	0h	Transmit-end-to-chip-select-inactive delay (value = 0-FFh). SPIDELAY[23-16] T2CDELAY is used in master mode only. It defines a hold time for the slave device that delays the chip select deactivation by a multiple of SPI module clock cycles after the last bit is transferred. T2CDELAY can be configured between 2 and 256 SPI module clock cycles (See Section 11.16.4.2.3.1.2). The hold time value is calculated as follows: $t_{T2CDELAY}$ = (T2CDELAY + 1) × SPI module clock period Note: If T2CDELAY = 0, then $t_{T2CDELAY}$ = 0 • ***Example:*** VBUSPCLK = 25 MHz -> VBUSPCLK period = 40 ns; T2CDELAY = 03h; then $t_{T2CDELAY}$ = 160 ns; After the last data bit (or parity bit) is being transferred the chip select signal is held active for 160 ns. • **Note:** If phase = 0, then between the last edge of SPIm_CLK and rise-edge of SPIm_SCSnx there will be an additional delay of 0.5 SPIm_CLK period. This is as per the SPI protocol.
15-0	RESERVED	R	0h	Reserved

Table 11-3411. Register Call Summary for SPIDELAY

SPI Registers • SPIDELAY Register (Offset = 48h) [reset = 0h]: [0][1][2] • SPI Registers: [0] • SPIDAT1 Register (Offset = 3Ch) [reset = 0h]: [0][1][2][3]
SPI Functional Description • Master Mode Settings: [0][1][2][3] • Master Mode Timing Options: [0][1][2][3][4][5][6][7]
SPI Programming Guide • Global Initialization: [0]

11.16.6.12 SPIDEF Register (Offset = 4Ch) [reset = 1h]

The SPI Default Chip Select Register (SPIDEF) is shown in Figure 11-1387 and described in Table 11-3413.

Table 11-3412. SPIDEF Instances

Instance	Physical Address
SPI0	2180 544Ch
SPI1	2180 584Ch
SPI2	2180 5C4Ch
SPI3	2180 604Ch

Figure 11-1387. SPIDEF Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						CSDEF	
R-0h						R/W-1h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3413. SPIDEF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	CSDEF	R/W	1h	Chip Select Default pattern. In master mode, the bit in this field determines the SPIm_SCSnx pin state when no transmissions are currently performed. It allows the user to set a chip select pattern which deselects all the SPI slaves. For more information about supported number of the chip select pins, see Section 11.16.1, *SPI Overview*. • 0 = The corresponding SPIm_SCSnx pin is set to 0 when no transfer occurs. • 1 = The corresponding SPIm_SCSnx pin is set to 1 when no transfer occurs.

Table 11-3414. Register Call Summary for SPIDEF

SPI Registers • SPIDEF Register (Offset = 4Ch) [reset = 1h]: [0] • SPI Registers: [0] • SPIDAT1 Register (Offset = 3Ch) [reset = 0h]: [0]
SPI Functional Description • Multiple Chip Selects: [0] • Master Mode Timing Options: [0]

11.16.6.13 SPIFMT0 to SPIFMT3 Register (Offset = 50h to 5Ch) [reset = 0h]

The SPI Data Format Registers (SPIFMTn, where n = 0 to 3) are shown in Figure 11-1388 and described in Table 11-3416.

Table 11-3415. SPIFMT0 to SPIFMT3 Instances

Instance	Physical Address
SPI0	2180 5450h to 2180 545Ch
SPI1	2180 5850h to 2180 585Ch
SPI2	2180 5C50h to 2180 5C5Ch
SPI3	2180 6050h to 2180 605Ch

Figure 11-1388. SPIFMT0 to SPIFMT3 Register

31	30	29	28	27	26	25	24
RESERVED			WDELAY				
R-0h			R/W-0h				
23	22	21	20	19	18	17	16
RESERVED			SHIFTDIR	RESERVED	DISCSTIMERS	POLARITY	PHASE
R-0h			R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
PRESCALE							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED			CHARLEN				
R-0h			R/W-0h				

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3416. SPIFMT0 to SPIFMT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29-24	WDELAY	R/W	0h	Delay in between transmissions. Idle time that will be applied at the end of the current transmission if the SPIDAT1[26] WDEL bit is set in the current buffer. The delay to be applied is equal to: WDELAY × $P_{SPI\ module\ clock}$ + 2 × $P_{SPI\ module\ clock}$ $P_{SPI\ module\ clock}$ -> Period of SPI module clock
23-21	RESERVED	R	0h	Reserved
20	SHIFTDIR	R/W	0h	Shift direction. • 0 = Most significant bit is shifted out first. • 1 = Least significant bit is shifted out first. • **Note:** Shift direction must be programmed in both master mode and slave mode.
19	RESERVED	R	0h	Reserved
18	DISCSTIMERS	R/W	0h	Disable chip select timers for this format register. The C2TDELAY and T2CDELAY timers are by default enabled for all the data format registers. Using this bit, these timers can be disabled for a particular data format if not required. When a master is handling multiple slaves, with varied set-up hold requirement, the application can selectively choose to include or not include the chip select delay timers for any slaves. • 0 = Both C2TDELAY and T2CDELAY counts are inserted for the chip selects. • 1 = No C2TDELAY or T2CDELAY is inserted in the chip select timings.

Table 11-3416. SPIFMT0 to SPIFMT3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	POLARITY	R/W	0h	SPI clock polarity. • 0 = SPI clock signal is low-inactive (before and after data transfer the clock signal is low). • 1 = SPI clock signal is high-inactive (before and after data transfer the clock signal is high).
16	PHASE	R/W	0h	SPI clock delay. • 0 = SPI clock signal is not delayed versus the transmit/receive data stream. The first data bit is transmitted with the first clock edge and the first bit is received with the second (inverse) clock edge. • 1 = SPI clock signal is delayed by a half SPI clock cycle versus the transmit/receive data stream. The first transmit bit has to output prior to the first clock edge. The master and slave receive the first bit with the first edge.
15-8	PRESCALE	R/W	0h	SPI prescaler (value = 0-FFh). It determines the bit transfer rate if the SPI is the network master and is directly derived from the SPI module clock. If the SPI is configured as slave, SPIFMTn[15-8] PRESCALE does not need to be configured. The clock rate can be calculated as: SPI clock frequency = SPI module clock/(PRESCALE + 1) • **Note:** When PRESCALEx is set to 0h, the SPI clock rate defaults to SPI module clock/2.
7-5	RESERVED	R	0h	Reserved
4-0	CHARLEN	R/W	0h	SPI data word length (value = 0-1Fh). Legal values are 2h (data word length = 2 bit) to 10h (data word length = 16). Illegal values, such as 0 or 1Fh are not detected and their effect is indeterminate. • **Note:** Data word length must be programmed in both master mode and slave mode.

Table 11-3417. Register Call Summary for SPIFMT0

SPI Registers
- SPI Registers: [0]
- SPIDAT0 Register (Offset = 38h) [reset = 0h]: [0][1]

11.16.6.14 SPI_INTVEC0 Register (Offset = 60h) [reset = 0h]

The SPI Interrupt Vector Register 0 (SPI_INTVEC0) is shown in Figure 11-1389 and described in Table 11-3419.

Table 11-3418. SPI_INTVEC0 Instances

Instance	Physical Address
SPI0	2180 5460h
SPI1	2180 5860h
SPI2	2180 5C60h
SPI3	2180 6060h

Figure 11-1389. SPI_INTVEC0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		INTVECT0					RESERVED
R-0h		R-0h					R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3419. SPI_INTVEC0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5-1	INTVECT0	R	0h	Interrupt vector for interrupt line INT0. SPI_INTVEC0[5-1] INTVECT0 field returns the vector of the pending interrupt at interrupt line INT0. If more than one interrupt is pending, SPI_INTVEC0[5-1] INTVECT0 field always references the highest priority interrupt source first. The interrupts available for SPI in the descending order of their priorities are as given below. • Transmission error Interrupt. • Receive buffer overrun interrupt. • Receive buffer full interrupt. • Transmit buffer empty interrupt. The SPI_INTVEC0[5-1] INTVECT0 field just reflects the status of SPIFLG register in a vectorized format. So, any updates to SPIFLG register will automatically reflect in the vector value in this register. Vectors for each of these interrupts will be reflected on the SPI_INTVEC0[5-1] INTVECT0 field, when they occur. Reading the vectors for the receive buffer overrun and receive buffer full interrupts will automatically clear the respective flags in the SPIFLG register. Reading the vector register when transmitter empty is indicated does not clear the SPIFLG[9] TXINTFLG bit. Writing a new data to SPIDAT0/SPIDAT1 register clears the transmitter empty interrupt. On reading the SPI_INTVEC0[5-1] INTVECT0 field, the vector of the next highest priority interrupt (if any) will be then reflected on the INTVECT0 bits. If two or more interrupts occur simultaneously, the vector for the highest priority interrupt will be reflected on the SPI_INTVEC0[5-1] INTVECT0 field. The following are the SPI interrupt vectors for line INT0: • 0h = No interrupt pending. • 1h-10h = Reserved. • 11h = Error interrupt pending. See the lower halfword of SPIINT0 register to determine more details about the type of error. • 12h = The pending interrupt is receive buffer full interrupt. • 13h = The pending interrupt is receive buffer overrun interrupt. • 14h = The pending interrupt is transmit buffer empty interrupt. • 15h-1Fh = Reserved.
0	RESERVED	R	0h	Reserved

Table 11-3420. Register Call Summary for SPI_INTVEC0

SPI Registers • SPI_INTVEC0 Register (Offset = 60h) [reset = 0h]: [0][1][2][3][4][5][6] • SPI Registers: [0] • SPIFLG Register (Offset = 10h) [reset = 0h]: [0][1] • SPI_INTVEC1 Register (Offset = 64h) [reset = 0h]: [0]
SPI Functional Description • Emulation Considerations: [0] • Interrupt Support: [0]

11.16.6.15 SPI_INTVEC1 Register (Offset = 64h) [reset = 0h]

The SPI Interrupt Vector Register 1 (SPI_INTVEC1) is shown in Figure 11-1390 and described in Table 11-3422.

Table 11-3421. SPI_INTVEC1 Instances

Instance	Physical Address
SPI0	2180 5464h
SPI1	2180 5864h
SPI2	2180 5C64h
SPI3	2180 6064h

Figure 11-1390. SPI_INTVEC1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		INTVECT1					RESERVED
R-0h		R-0h					R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3422. SPI_INTVEC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5-1	INTVECT1	R	0h	Interrupt vector for interrupt line INT1. SPI_INTVEC1[5-1] INTVECT1 field returns the vector of the pending interrupt at interrupt line INT1. If more than one interrupt is pending, SPI_INTVEC1[5-1] INTVECT1 field always references the highest priority interrupt source first. The interrupts available for SPI in the descending order of their priorities are as given below. • Transmission error Interrupt. • Receive buffer overrun interrupt. • Receive buffer full interrupt. • Transmit buffer empty interrupt. The SPI_INTVEC1[5-1] INTVECT1 field just reflects the status of SPIFLG in a vectorized format. So, any updates to SPIFLG register will automatically reflect in the vector value in this register. Vectors for each of these interrupts will be reflected on the SPI_INTVEC0[5-1] INTVECT0 field, when they occur. Reading the vectors for the receive buffer overrun and receive buffer full interrupts will automatically clear the respective flags in the SPIFLG register. Reading the vector register when transmitter empty is indicated does not clear the SPIFLG[9] TXINTFLG bit. Writing a new data to SPIDAT0/SPIDAT1 register clears the transmitter empty interrupt. On reading the SPI_INTVEC1[5-1] INTVECT1 field, the vector of the next highest priority interrupt (if any) will be then reflected on the SPI_INTVEC1[5-1] INTVECT1 field. If two or more interrupts occur simultaneously, the vector for the highest priority interrupt will be reflected on the SPI_INTVEC1[5-1] INTVECT1 field. The following are the SPI interrupt vectors for line INT1: • 0h = No interrupt pending. • 1h-10h = Reserved. • 11h = Error interrupt pending. See the lower halfword of SPIINT0 register to determine more details about the type of error. • 12h = The pending interrupt is receive buffer full interrupt. • 13h = The pending interrupt is receive buffer overrun interrupt. • 14h = The pending interrupt is transmit buffer empty interrupt. • 15h-1Fh = Reserved.
0	RESERVED	R	0h	Reserved

Table 11-3423. Register Call Summary for SPI_INTVEC1

SPI Registers • SPI Registers: [0] • SPIFLG Register (Offset = 10h) [reset = 0h]: [0][1] • SPI_INTVEC1 Register (Offset = 64h) [reset = 0h]: [0][1][2][3][4][5][6]
SPI Functional Description • Emulation Considerations: [0] • Interrupt Support: [0]

11.16.6.16 SPIREV Register (Offset = 1FCh) [reset = 4A050300h]

SPIREV is shown in Figure 11-1391 and described in Table 11-3425.

Table 11-3424. SPIREV Instances

Instance	Physical Address
SPI0	2180 55FCh
SPI1	2180 59FCh
SPI2	2180 5DFCh
SPI3	2180 61FCh

Figure 11-1391. SPIREV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-4A050300h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3425. SPIREV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4A050300h	TI internal data. Identifies revision of peripheral.

Table 11-3426. Register Call Summary for SPIREV

SPI Registers
- SPI Registers: [0]
- SPIREV Register (Offset = 1FCh) [reset = 4A050300h]: [0]

11.17 Timers

This chapter describes the Timer modules for the device.

11.17.1 Timers Overview

There are total of 7 chip-level timers.

The device includes several types of timers used by the system software, including general-purpose (GP) timers, watchdog timers, and a wake-up timer, as it follows:

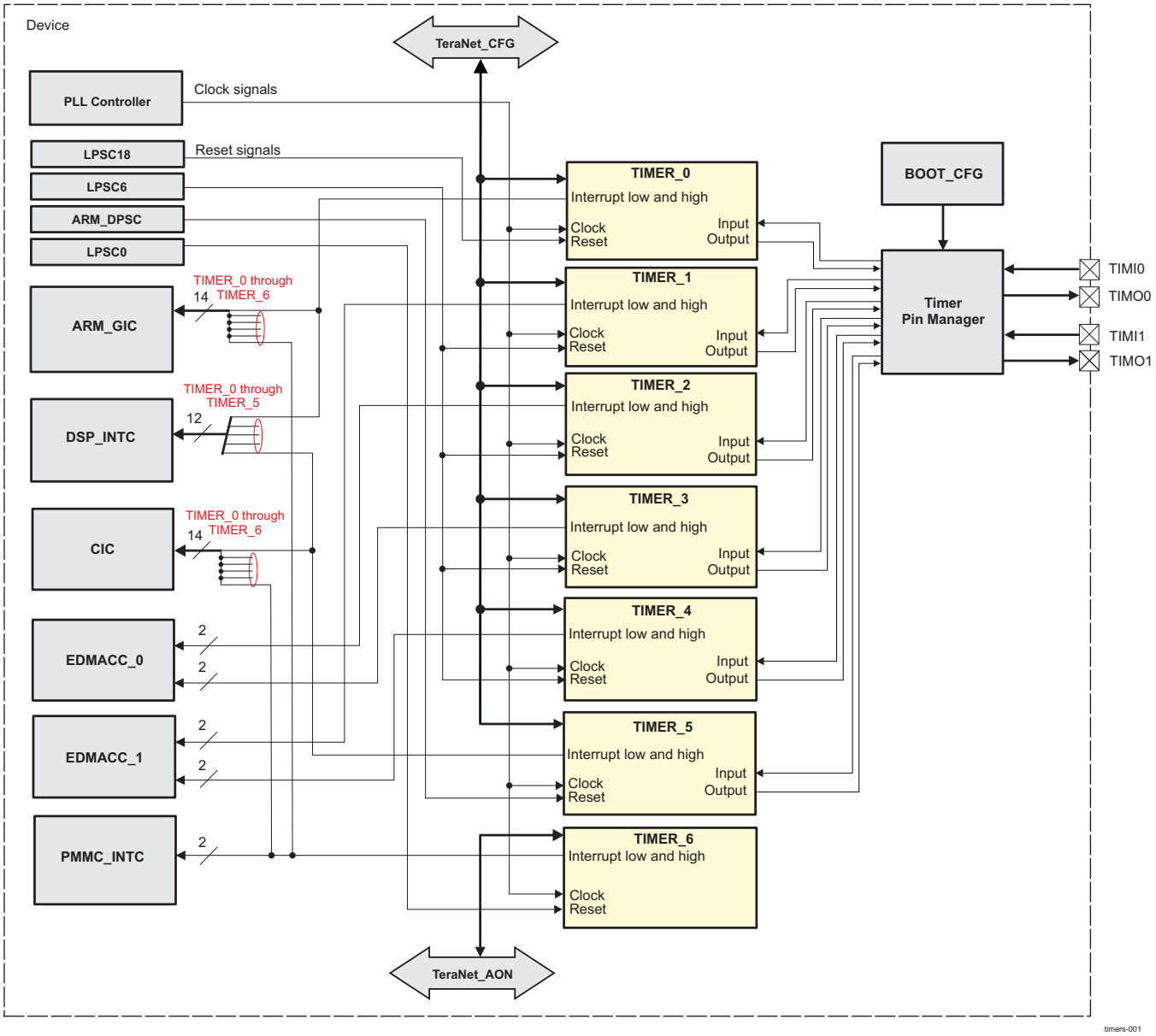
- TIMER_0 is dedicated/tightly coupled for C66x CorePac. TIMER_0 can be used as general-purpose timer or watchdog timer
- TIMER_1 through TIMER_4 are general-purpose timers
- TIMER_5 is dedicated/tightly coupled for the Arm core 0. TIMER_5 can be used as general-purpose timer or watchdog timer
- TIMER_6 is dedicated as device wake-up timer by interrupting PMMC CPU. TIMER_6 cannot be used by high-level software as a general-purpose timer or watchdog. TIMER_6 is neither connected to Timer pin manager block nor to Timer IOs.

Each timer has two input pins (TINPL and TINPH) and two output pins (TOUTL and TOUTH).

At the chip level there are 4 timer pins — two input pins (TIMI[1:0]) and two output pins (TIMO[1:0]). Each of TIMER_0 through TIMER_5 input can be configured to be driven by the timer input pins. Each of TIMO[1:0] output pin can be driven by any of the timer outputs. The selection of timer inputs and outputs is controlled by Timer pin manager. The Timer pin manager block is controlled by registers in BOOT_CFG module. For more information see Section 5.1.3.1.9, *Timer Pin Manager Control Registers*.

Figure 11-1392 shows a high-level block diagram of the timer circuitry.

Figure 11-1392. Timers Overview



The timer can be driven by an external clock at the timer input pin (TINPL) or by the divide-down clock rate of the internal clock. TINPH can be used to drive the higher 32-bit timer, TIMHI, in unchained mode. The internal clock is generated by the PLL Controller and is a divided-down version of the CHIP_CLK1 clock. For more information, see Section 5.4, *Clock Management*.

11.17.1.1 Timers Features

Timer modules are identical instances of hardware supporting three modes. The modes are selected in TIMER_TGCR[3-2] TIMMODE bits:

- 64-bit general-purpose (GP) timer
- Dual 32-bit timers (TIMLO and TIMHI in chain mode or unchained mode)
- Watchdog timer

Timer integration in this device implies timer usage as described in Section 11.17.1, *Timers Overview*.

11.17.2 Timers Environment

11.17.2.1 Timers External System Interface

Figure 11-1393 shows the external system interface for the timers, and Table 11-3427 describes the GP timer inputs and outputs.

NOTE: Software control (Timer pin manager) must ensure that MUX mode is configured to select the TIMER_i (where i = 0 to 5) signal on four different pads.

Note that TIMER_6 is not connected to Timer pin manager block nor to Timer IOs.

For more information about the configuration of the TIMER_i I/O pads, see Section 5.1.3.1.1, *Pad Configuration Registers*.

Figure 11-1393. Timers External System Interface

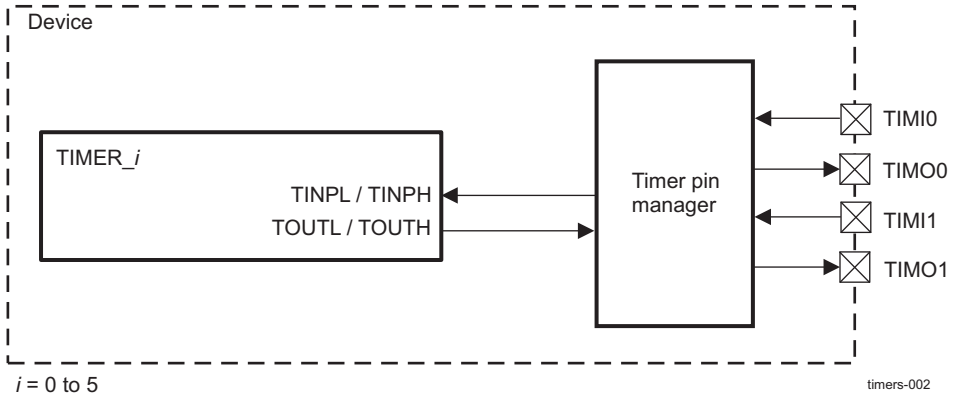


Table 11-3427. Input/Output Description

Pin Name	Type [1]	Reset Value	Signal Name	Description
TIMI0	I		TINPL / TINPH	TIMER_0 input low/high or TIMER_1 input low/high or TIMER_2 input low/high or TIMER_3 input low/high or TIMER_4 input low/high or TIMER_5 input low/high
TIMO0	O	0	TOUTL / TOUTH	TIMER_0 output low/high or TIMER_1 output low/high or TIMER_2 output low/high or TIMER_3 output low/high or TIMER_4 output low/high or TIMER_5 output low/high
TIMI1	I		TINPL / TINPH	TIMER_0 input low/high or TIMER_1 input low/high or TIMER_2 input low/high or TIMER_3 input low/high or TIMER_4 input low/high or TIMER_5 input low/high
TIMO1	O	0	TOUTL / TOUTH	TIMER_0 output low/high or TIMER_1 output low/high or TIMER_2 output low/high or

(1) When configured for that function; I = Input, O = Output

Table 11-3427. Input/Output Description (continued)

Pin Name	Type [1]	Reset Value	Signal Name	Description
				TIMER_3 output low/high or
				TIMER_4 output low/high or
				TIMER_5 output low/high

11.17.3 Timers Integration

This section describes module integration in the device, including information about clocks, resets, and hardware requests.

Figure 11-1394 shows the integration of the timers in the device.

Figure 11-1394. Timers Integration

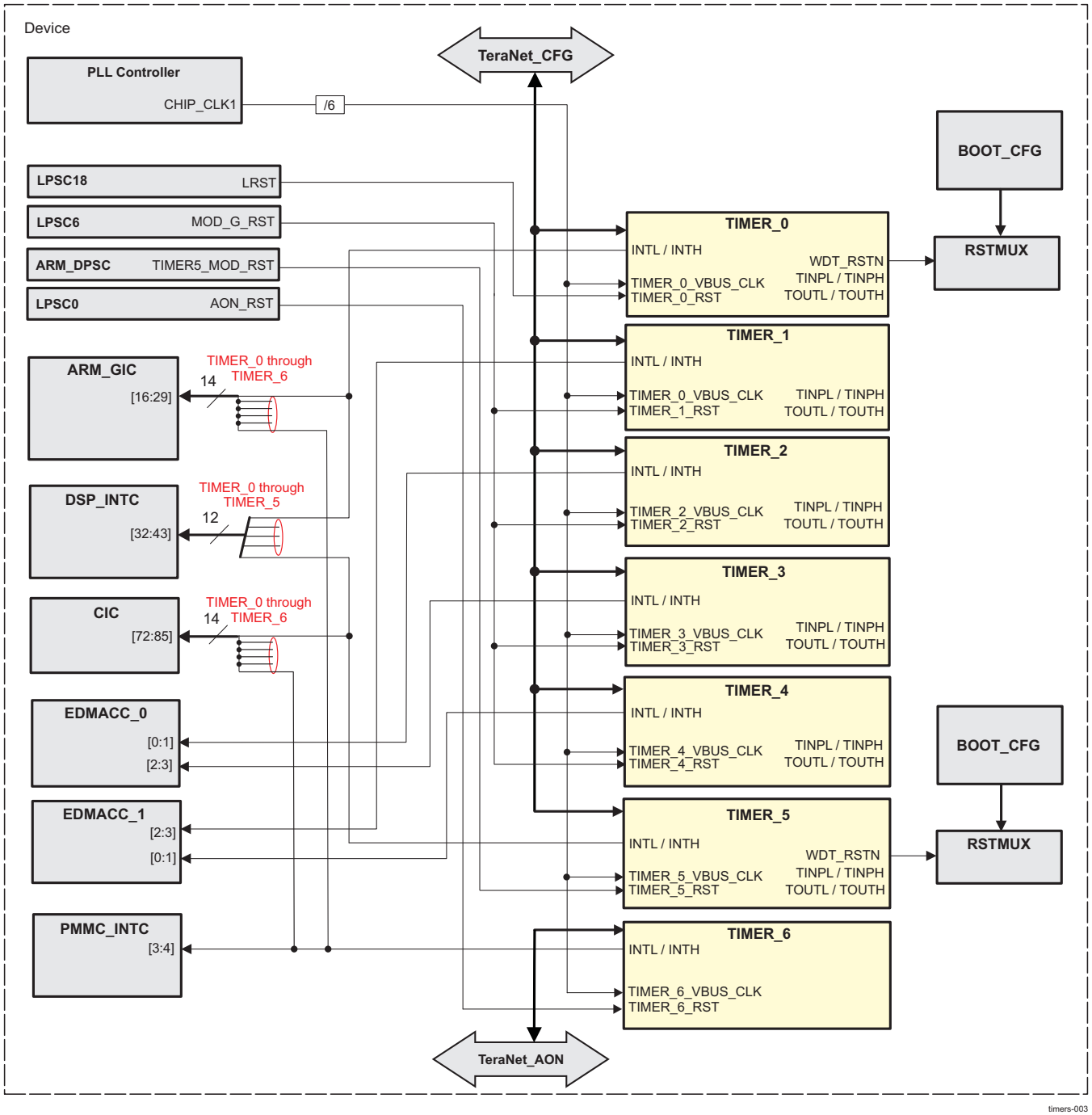


Table 11-3428 through Table 11-3430 summarize the integration of the module in the device.

Table 11-3428. Timers Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
TIMER_0	PD8	LPSC18	TeraNet_CFG
TIMER_1	PD5	LPSC6	TeraNet_CFG
TIMER_2	PD5	LPSC6	TeraNet_CFG
TIMER_3	PD5	LPSC6	TeraNet_CFG
TIMER_4	PD5	LPSC6	TeraNet_CFG
TIMER_5	PD9	ARM_DPSC	TeraNet_CFG
TIMER_6	PD0	LPSC0	TeraNet_AON

Table 11-3429. Timers Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
TIMER_0	TIMER_0_VBUS_CLK	CHIP_CLK1/6	PLL Controller	TIMER_0 Inteface and Functional clock
TIMER_1	TIMER_1_VBUS_CLK	CHIP_CLK1/6	PLL Controller	TIMER_1 Inteface and Functional clock
TIMER_2	TIMER_2_VBUS_CLK	CHIP_CLK1/6	PLL Controller	TIMER_2 Inteface and Functional clock
TIMER_3	TIMER_3_VBUS_CLK	CHIP_CLK1/6	PLL Controller	TIMER_3 Inteface and Functional clock
TIMER_4	TIMER_4_VBUS_CLK	CHIP_CLK1/6	PLL Controller	TIMER_4 Inteface and Functional clock
TIMER_5	TIMER_5_VBUS_CLK	CHIP_CLK1/6	PLL Controller	TIMER_5 Inteface and Functional clock
TIMER_6	TIMER_6_VBUS_CLK	CHIP_CLK1/6	PLL Controller	TIMER_6 Inteface and Functional clock
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
TIMER_0	TIMER_0_RST	LRST	LPSC18	TIMER_0 Module Reset
TIMER_1	TIMER_1_RST	MOD_G_RST	LPSC6	TIMER_1 Module Reset
TIMER_2	TIMER_2_RST	MOD_G_RST	LPSC6	TIMER_2 Module Reset
TIMER_3	TIMER_3_RST	MOD_G_RST	LPSC6	TIMER_3 Module Reset
TIMER_4	TIMER_4_RST	MOD_G_RST	LPSC6	TIMER_4 Module Reset
TIMER_5	TIMER_5_RST	TIMER5_MOD_RST	ARM_DPSC	TIMER_5 Module Reset
TIMER_6	TIMER_6_RST	AON_RST	LPSC0	TIMER_6 Module Reset

Table 11-3430. Timers Hardware Requests

Interrupt Requests						
Module Instance	**Event Name**	**Mapped To Input Event [Number]**				**Description**
		ARM GIC	**CIC**	**DSP INTC**	**PMMC INTC**	
TIMER_0[(1)]	TIMER_0_INTL	[16]	[72]	[32]	-	TIMER_0 Interrupt low
	TIMER_0_INTH	[17]	[73]	[33]	-	TIMER_0 Interrupt high
TIMER_1	TIMER_1_INTL	[18]	[74]	[34]	-	TIMER_1 Interrupt low
	TIMER_1_INTH	[19]	[75]	[35]	-	TIMER_1 Interrupt high
TIMER_2	TIMER_2_INTL	[20]	[76]	[36]	-	TIMER_2 Interrupt low
	TIMER_2_INTH	[21]	[77]	[37]	-	TIMER_2 Interrupt high
TIMER_3	TIMER_3_INTL	[22]	[78]	[38]	-	TIMER_3 Interrupt low
	TIMER_3_INTH	[23]	[79]	[39]	-	TIMER_3 Interrupt high
TIMER_4	TIMER_4_INTL	[24]	[80]	[40]	-	TIMER_4 Interrupt low
	TIMER_4_INTH	[25]	[81]	[41]	-	TIMER_4 Interrupt high

(1) Because TIMER_0 is used as DSP watchdog timer additionally to TIMER_0_INTL and TIMER_0_INTH there is WDT_RSTN signal. This watchdog timer signal is controlled by the registers in the RSTMUX. For more information about RSTMUX registers, see Section 5.1.3.1.8, *Reset Mux Control Registers*.

Table 11-3430. Timers Hardware Requests (continued)

TIMER_5[(2)]	TIMER_5_INTL	[26]	[82]	[42]	-	TIMER_5 Interrupt low
	TIMER_5_INTH	[27]	[83]	[43]	-	TIMER_5 Interrupt high
TIMER_6	TIMER_6_INTL	[28]	[84]	-	[3]	TIMER_6 Interrupt low
	TIMER_6_INTH	[29]	[85]	-	[4]	TIMER_6 Interrupt high

DMA Requests

Module Instance	Event Name	Mapped To Input Event [Number]		Description
		EDMACC_0	EDMACC_1	
TIMER_1	TIMER_1_INTL	-	[2]	TIMER_1 Interrupt low
	TIMER_1_INTH	-	[3]	TIMER_1 Interrupt high
TIMER_2	TIMER_2_INTL	[0]	-	TIMER_2 Interrupt low
	TIMER_2_INTH	[1]	-	TIMER_2 Interrupt high
TIMER_3	TIMER_3_INTL	[2]	-	TIMER_3 Interrupt low
	TIMER_3_INTH	[3]	-	TIMER_3 Interrupt high
TIMER_4	TIMER_4_INTL	-	[0]	TIMER_4 Interrupt low
	TIMER_4_INTH	-	[1]	TIMER_4 Interrupt high

(2) Because TIMER_5 is used as Arm watchdog timer additionally to TIMER_5_INTL and TIMER_5_INTH there is WDT_RSTN signal. This watchdog timer signal is controlled by the registers in the RSTMUX. For more information about RSTMUX registers, see Section 5.1.3.1.8, *Reset Mux Control Registers*.

NOTE: For the description of the interrupt source, see Section 11.17.4.9, *Timer Interrupt Rate*.

11.17.3.1 Watchdog Timer Integration

When in watchdog timer mode, an extra active-low signal called WDT_RSTN is provided to ease the chip-level integration. This signal is an exact duplication and inversion of the TOUTL pin but only valid in watchdog timer mode. In GP timer mode, this signal is inactive (stays high).

In Aem, watchdog event for TIMER_5 (Arm dedicated) is routed to the chip level Arm interrupt controller (ARM_INTC) and also to DSP interrupt controller (via INTC) to enable multiple options to handle watchdog timeout exception. As Arm does not support a local CPU reset input like the DSP do, the only options are to do a device reset or in certain situations DSP core can power cycle the errant Arm core.

The first option is to enable device reset on watchdog reset which can be configured using RSTMUX8. Here no interrupt handling or other software intervention after watchdog timeout event is required.

In the second option the DSP software can power cycle Arm core (dynamic power down) or entire Arm domain when watchdog times out, based on device level interrupt to DSP (either directly or via INTC).

For more information about Watchdog Timer Mode refer to Section 11.17.4.12, *Watchdog Timer Mode*.

11.17.3.1.1 RSTMUX

Timer can be configured to act as a watchdog timer or a general purpose timer. When configured as a watchdog timer, the software periodically writes the appropriate value in the specified register so that the watchdog timer does not time out. The timer output is routed to the reset input of the DSP LPSC through the RSTMUX block.

In order for the timers to reset the CPU, there needs to be a register that controls the routing of the events to the CPU. This is done in the RSTMUX registers. RSTMUX0 register correspond to DSP_INTC. RSTMUX8 register correspond to ARM_GIC.

For more information about RSTMUX block and registers refer to Section 5.1.3.1.8, *Reset Mux Control Registers*.

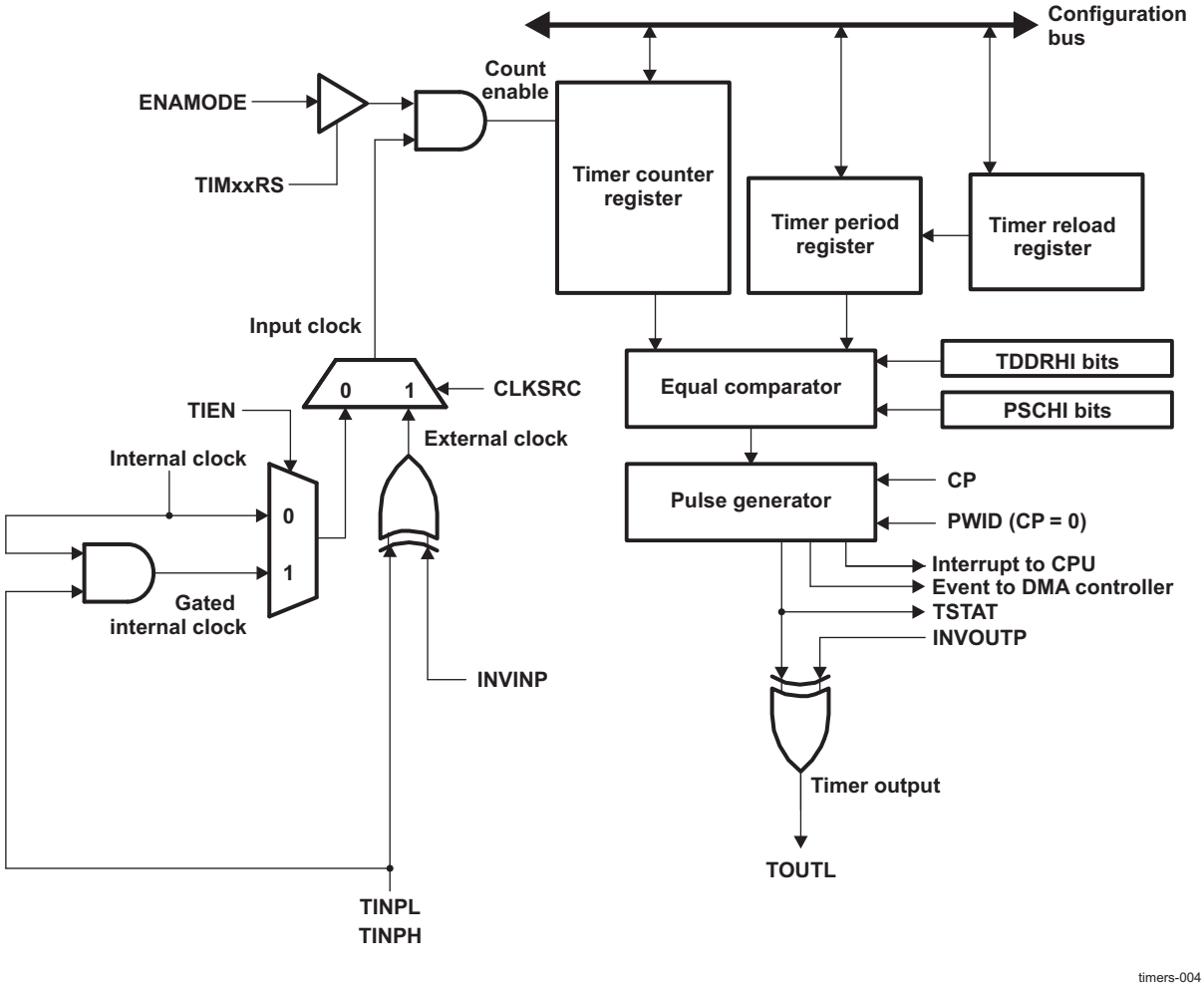
11.17.4 Timers Functional Description

This section describes the functional description of the timers.

11.17.4.1 Timers Block Diagram

Figure 11-1395 is a block diagram of the timers in the device.

Figure 11-1395. Timers Block Diagram



11.17.4.2 Timer Modes

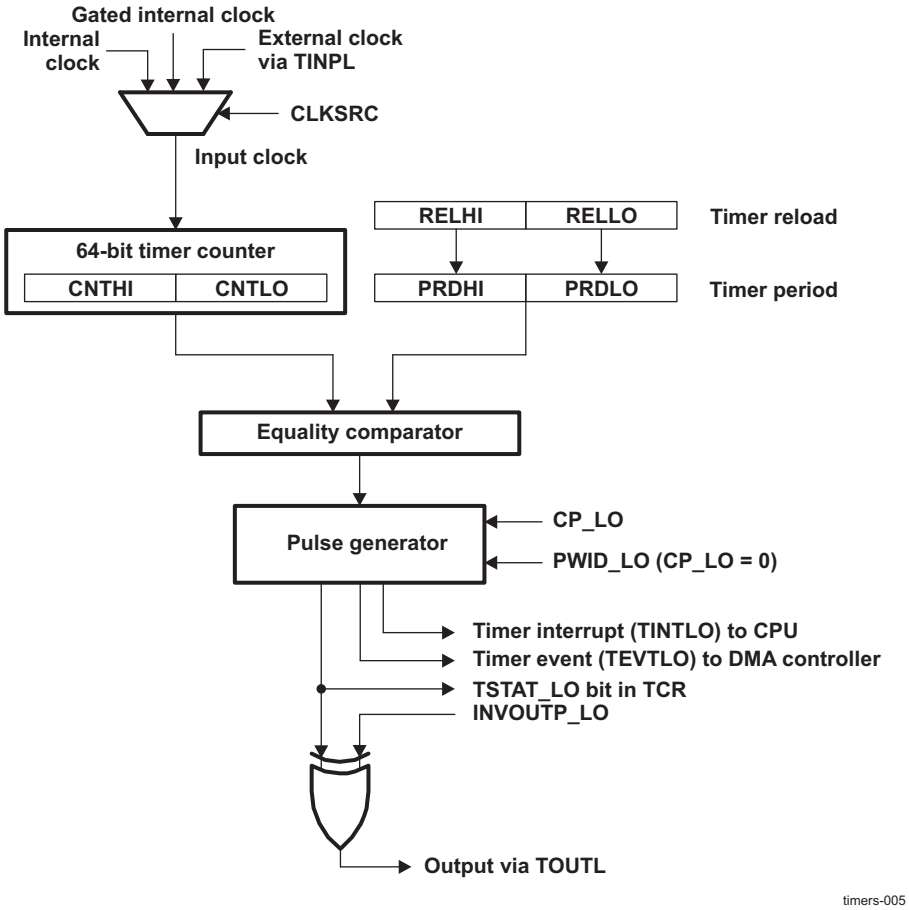
11.17.4.2.1 64-Bit Timer Mode

The timer can be configured as a 64-bit general-purpose (GP) timer, using the TIMER_TGCR[3-2] TIMMODE bit. At reset, the timer is in 64-bit GP timer mode.

In this mode, the timer operates as a 64-bit up-counter, as shown in Figure 11-1396. The counter registers (TIMER_CNTLO, TIMER_CNTHI) and the period registers (TIMER_PRDLO, TIMER_PRDHI) form a 64-bit timer counter register and a 64-bit timer period register, respectively. When the timer is enabled (see Table 11-3433), the timer counter starts incrementing by 1 at every timer input clock cycle. When the timer counter matches the timer period, it generates a maskable timer interrupt (TINTLO), a timer event (TEVTLO), and an output signal on the timer output pin, TOUTL.

When in pulse mode (CP_LO = 0), the timer output pin (TOUTL) asserts a pulse that is 1, 2, 3, or 4 timer clock cycles wide, depending on the setting of the pulse width bits TIMER_TCR[5-4] PWID_LO. When the timer is configured in continuous mode, the timer counter is reset to 0 on the cycle after the timer counter reaches the timer period. When the timer counter matches the timer period, the period registers will be loaded with the values in the reload registers (TIMER_RELLO, TIMER_RELHI) if TIMER_TCR[7-6] ENAMODE_LO = 3h. The timer can be stopped, restarted, reset, or disabled using the bits of the timer control register.

Figure 11-1396. 64-Bit Timer Mode Block Diagram



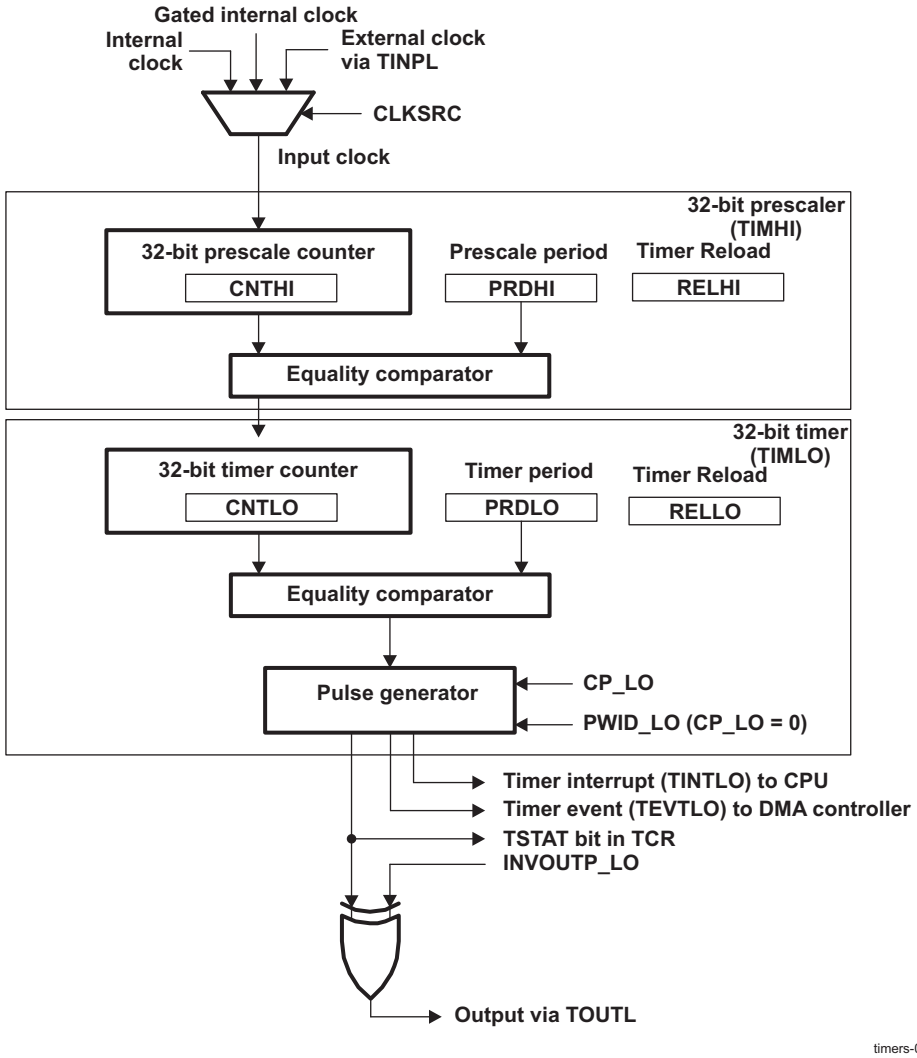
11.17.4.2.2 Dual 32-bit Timer Mode

The timer can be broken down into two 32-bit timers, using the TIMER_TGCR[3-2] TIMMODE. In this mode, the two 32-bit timers can be operated in conjunction with each other (chained mode) or independently.

11.17.4.2.2.1 Chained Mode

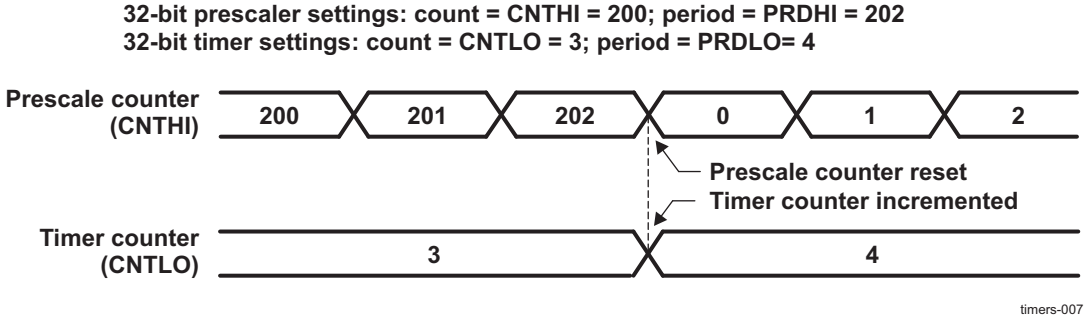
In the chained mode, shown in Figure 11-1397, one 32-bit timer (TIMHI) is used as a 32-bit prescaler to a second timer (TIMLO).

Figure 11-1397. Dual 32-Bit Timers Chained Mode Block Diagram



The 32-bit prescaler (TIMHI) uses the counter register (TIMER_CNTHI) and the period register (TIMER_PRDHI) to form a 32-bit prescale counter register and a 32-bit prescale period register, respectively. When the timer is enabled, the prescale counter starts incrementing by 1 at every timer input clock cycle. On the cycle after the prescale counter matches the prescale period, a clock signal is generated and the prescale counter register is reset to 0 (see the example in Figure 11-1398).

Figure 11-1398. Dual 32-Bit Timers Chained Mode Example



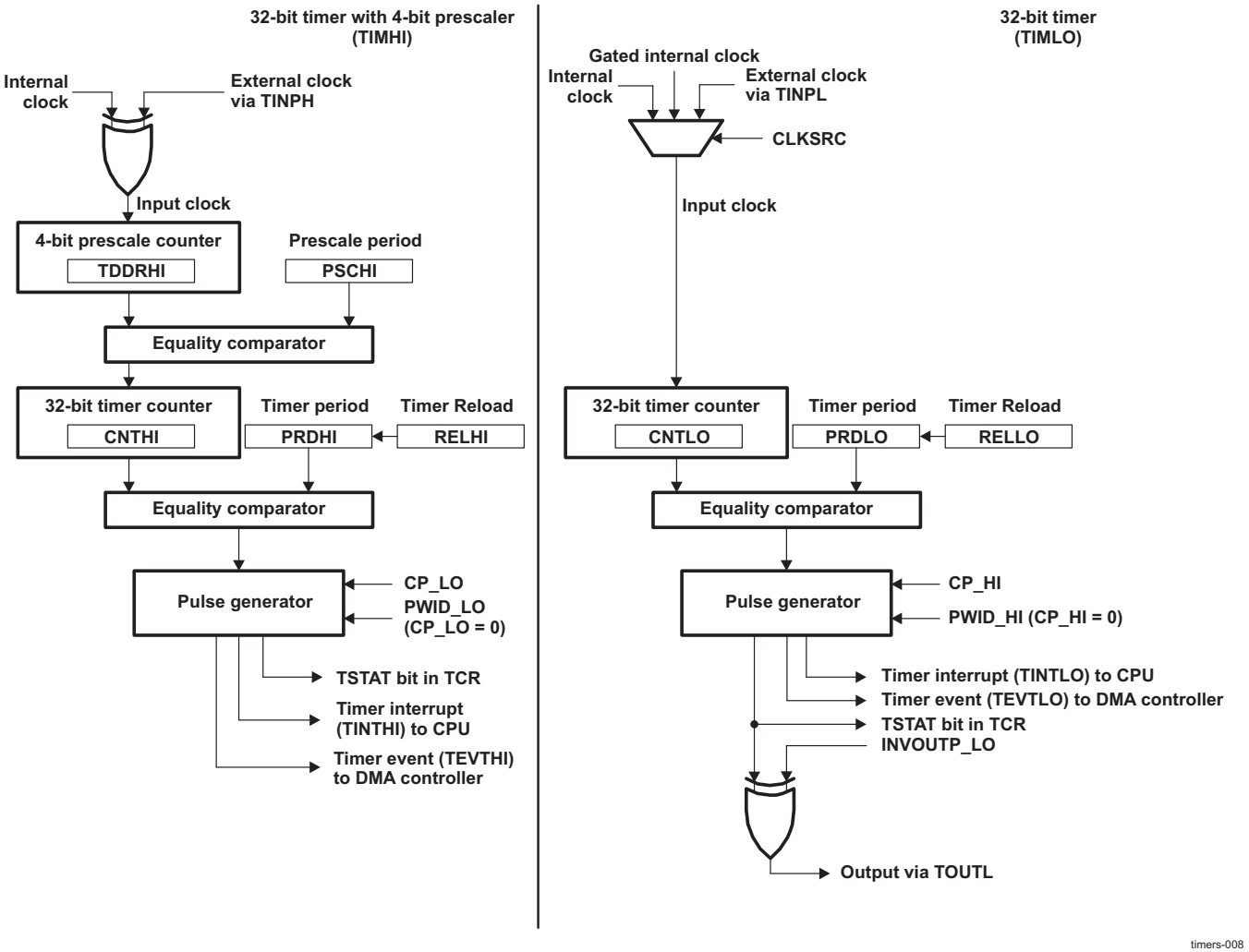
The other 32-bit timer (TIMLO) uses the counter register (TIMER_CNTLO) and the period register (TIMER_PRDLO) to form a 32-bit timer counter register and a 32-bit timer period register, respectively. This timer is clocked by the output clock from the prescaler (see the example in Figure 11-1398). The timer counter increments by 1 at every prescaler output clock cycle. When the timer counter matches the timer period, a maskable timer interrupt (TINTLO), a timer DMA event (TEVTLO), and an output signal are generated.

When in pulse mode (CP_LO = 0), the timer output (TOUTL) asserts a pulse that is 1, 2, 3, or 4 timer clock cycles wide, depending on the setting of the pulse width bits TIMER_TCR[5-4] PWID_LO. When the timer is configured in continuous mode, the timer counter is reset to 0 on the cycle after the timer counter reaches the timer period. The timer can be stopped, restarted, reset, or disabled using the bits of the timer control register. The timer control register (TIMER_TCR) does not control the TIMHI in this mode.

11.17.4.2.2.2 Unchained Mode

In the unchained mode, shown in Figure 11-1399, the timer can operate as two independent 32-bit timers. One 32-bit timer (TIMHI) can be configured as a 32-bit timer being clocked by a 4-bit prescaler. The other (TIMLO) can be used as a 32-bit timer.

Figure 11-1399. Dual 32-Bit Timers Unchained Mode Block Diagram



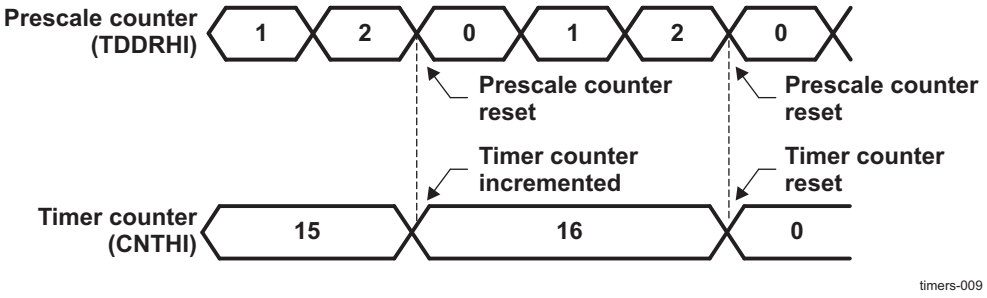
11.17.4.2.2.1 32-Bit Timer With a 4-Bit Prescaler (TIMHI)

In the unchained mode, the 4-bit prescaler can be clocked by either the internal clock or the external clock source (TINPH) for TIMHI. The 4-bit prescaler uses the timer divide-down ratio bits TIMER_TGCR[15-12] TDDRHI and the prescale counter bits TIMER_TCR[11-8] PSCHI to form a 4-bit prescale counter register and a 4-bit prescale period register, respectively.

The 32-bit timer uses the counter register (TIMER_CNTHI) and the period register (TIMER_PRDHI) to form a 32-bit timer counter register and a 32-bit timer period register, respectively. The 32-bit timer is clocked by the output clock from the 4-bit prescaler (see the example in Figure 11-1400). When the timer is enabled, the timer counter increments by 1 at every prescaler output clock cycle. When the timer counter matches the timer period, a maskable timer interrupt (TINTHI) and a timer DMA event (TEVTHI) are generated. The state of the output signal is read in the timer status TIMER_TCR[16] TSTAT_HI bit. When in pulse mode (CP_HI = 0), TIMER_TCR[16] TSTAT_HI stays high or low for 1, 2, 3, or 4 timer clock cycles. The pulse width depends on the setting of the pulse width TIMER_TCR[21-20] PWID_HI bits. When in clock mode (CP_HI = 1), the TSTAT bit changes state (high-to-low or low-to-high) every time the timer counter matches the timer period. When the timer is configured in continuous mode, the timer counter is reset to 0 on the cycle after the timer counter reaches the timer period (see the example in Figure 11-1400). Note that when the timer counter matches the timer period, the period register (TIMER_PRDHI) will be loaded with the value in the reload register (TIMER_RELHI) if TIMER_TCR[23-22] ENAMODE_HI = 3h.The timer can be stopped, restarted, reset, or disabled using TIMER_TCR and the timer global control register (TIMER_TGCR).

Figure 11-1400. Dual 32-Bit Timers Unchained Mode Example

4-bit prescaler settings: count = TDDRHI = 1; period = PSCHI = 2
32-bit timer settings: count = CNTHI= 15; period = PRDHI = 16



11.17.4.2.2.2 32-Bit Timer (TIMLO)

The other 32-bit timer (TIMLO) uses the counter register (TIMER_CNTLO) and the period register (TIMER_PRDLO) to form a 32-bit timer counter register and a 32-bit timer period register, respectively. When the timer is enabled, the timer counter increments by 1 at every timer input clock cycle. When the timer counter matches the timer period, a maskable timer interrupt (TINTLO), a timer DMA event (TEVTLO), and an output signal (TOUTL) are generated. The state of the output signal is also read in the timer status (TSTAT) bit of the timer control register (TIMER_TCR). When in pulse mode (CP_LO = 0) and depending on the timer output inverter control (INVOUTP) bit in TIMER_TCR, the timer output pin (TOUTL) stays high or low for 1, 2, 3, or 4 timer clock cycles.

The pulse width depends on the setting of the pulse width TIMER_TCR[5-4] PWID_LO bits. When in clock mode (CP_LO = 1), the timer output and the TIMER_TCR[0] TSTAT_LO bit change state (high-to-low or low-to-high) every time the timer counter matches the timer period. When the timer is configured in continuous mode, the timer counter is reset to 0 on the cycle after the timer counter reaches the timer period. When the timer counter matches the timer period, the period register will be loaded with the value in the reload register (TIMER_RELLO) if TIMER_TCR[7-6] ENAMODE_LO = 3h. The timer can be stopped, restarted, reset, or disabled using TIMER_TCR and the timer global control register (TIMER_TGCR).

11.17.4.2.3 Counter, Reload and Period Registers Used in GP Timer Modes

Table 11-3431 summarizes the counter registers (TIMER_CNTLO and TIMER_CNTHI), reload registers (TIMER_RELLO and TIMER_RELHI) and period registers (TIMER_PRDLO and TIMER_PRDHI) used in each GP timer mode.

Table 11-3431. Counter and Period Registers Used in GP Timer Modes

Timer Mode		Counter Registers	Reload Registers	Period Registers
64-bit general-purpose		TIMER_CNTHI: TIMER_CNTLO	TIMER_RELHI: TIMER_RELLO	TIMER_PRDHI: TIMER_PRDLO
Dual 32-bit chained	Prescaler (TIMHI)	TIMER_CNTHI	-	TIMER_PRDHI
	Timer (TIMLO)	TIMER_CNTLO	-	TIMER_PRDLO
Dual 32-bit unchained	Timer (TIMLO)	TIMER_CNTLO	TIMER_RELLO	TIMER_PRDLO
	Timer with prescaler (TIMHI)	PSCHI bits and TIMER_CNTHI	TIMER_RELHI	TDDRHI bits and TIMER_PRDHI

11.17.4.2.4 Timer Plus/Additional Modes

The Timer supports the following additional features:

- Period reload
- External event capture mode
- Timer counter register read reset mode

By default, period reload, external event capture mode, timer counter register read reset mode, timer counter capture registers, and interrupt/DMA/TOUTL generation control and status are not available. Setting the TIMER_TGCR[4] PLUSEN bit enables these features.

11.17.4.2.4.1 Timer Capture Registers

When the timer has a timeout due to a normal expiration of timer, external input event in Event Capture Mode, or read of timer counter registers in Read Reset Mode, the values of the timer counter registers (TIMER_CNTLO and TIMER_CNTHI) are copied onto the timer counter capture registers (TIMER_CAPLO and TIMER_CAPHI). Note that the value in TDDR is not captured when a read of TIMER_CNTHI happens.

11.17.4.2.4.2 Event Capture Mode

When the TIMER_TGCR[4] PLUSEN bit is set, Event Capture Mode is available for TIMER_CNTLO when the timer is configured in 32-bit unchained mode. When Event Capture Mode is enabled, the timer cycle is restarted when an external input event occurs on pin TINPL. In particular, when an external input event occurs, the timer stops counting, generates output CPU interrupts and DMA events, copies values from the timer counter register TIMER_CNTLO to the timer capture register TIMER_CAPLO, reloads the timer period register TIMER_PRDLO if in continuous mode with period reload (ENAMODE = 3h), resets the timer counter register TIMER_CNTLO and then restarts counting in continuous mode. Event Capture Mode is available only when the timer clock source is the internal timer (CLKSRC = 0) and the timer is in continuous mode (ENAMODE = 2h or 3h). Capture mode is enabled using the Capture mode enable bit TIMER_TCR[11] CAPMODE_LO. The type of input event is selected by the capture event mode bit TIMER_TCR[13-12] CAPEVTMODE_LO. All of the following input event types are available:

- Rising edge of input signal
- Falling edge of input signal
- Rising or falling edge of input signal

11.17.4.2.4.3 Timer Counter Register Read Reset Mode

Read Reset Mode is available when the TIMER_TGCR[4] PLUSEN bit is set and the timer is configured in 32-bit unchained mode. When Read Reset Mode is enabled, the timer cycle will restart when the timer counter registers are read (TIMER_CNTLO and/or TIMER_CNTHI). In particular, when the timer registers are read, the timer stops counting, copies values from the timer counter registers (TIMER_CNTLO and/or TIMER_CNTHI) to the timer capture registers (TIMER_CAPLO and/or TIMER_CAPHI), reloads the timer period registers (TIMER_PRDLO and/or TIMER_PRDHI) if in continuous mode with period reload (ENAMODE = 3h), and then restarts counting in continuous mode. Timer output events (TINTn, TEVTn, and TOUTn) are not generated during this process. Read Reset Mode is enabled using the read reset mode enable bit (READRSTMODE) in the timer control register (TIMER_TCR).

11.17.4.3 Timer Operation

The following sections describe the overall timer operation. For specific details on the watchdog timer operation, see Section 11.17.4.12.

11.17.4.3.1 Timer Mode Selection

The timer can be configured as a 64-bit general-purpose timer or dual 32-bit timers (chained or unchained), or as a watchdog timer using the timer mode (TIMMODE) bits in timer global control register (TIMER_TGCR) (see Table 11-3432). At reset, the timer is configured as a 64-bit GP timer by default. These bits can be written to select the respective function as shown in Table 11-3432.

Table 11-3432. Timer Mode Selection

TIMMODE Bits		Timer Mode
Bit 3	Bit 2	
0	0	64-bit general-purpose timer (default)
0	1	Dual 32-bit timers (unchained)
1	0	64-bit watchdog timers
1	1	Dual 32-bit timers (chained)

11.17.4.3.2 Timer Enabling

In the 64-bit timer mode or the dual 32-bit timers chained mode, the timer can be enabled by setting the TIMER_TGCR[0] TIMLORS and TIMER_TGCR[1] TIMHIRS bits to 1 and setting the TIMER_TCR[7-6] ENAMODE_LO bits to 1h, 2h or 3h.

In the dual 32-bit timers unchained mode, the 32-bit timer (TIMLO) can be enabled by setting the TIMER_TGCR[0] TIMLORS bit to 1 and the TIMER_TCR[7-6] ENAMODE_LO bits to 1h, 2h or 3h. The 32-bit timer with prescaler (TIMHI) can be enabled by setting the TIMER_TGCR[1] TIMHIRS to 1 and the TIMER_TCR[23-22] ENAMODE_HI to 1h, 2h or 3h.

Table 11-3433 is a summary of timer enabling.

Table 11-3433. Timer Enabling

Timer Mode	TIMER_TCR ENAMODE bits				TIMER_TGCR		Timer Status
	Bit 23	Bit 22	Bit 7	Bit 6	TIMHIRS	TIMLORS	
64-bit general-purpose	X	X	0	0	X	X	Disabled (default)
	X	X	0	1	1	1	Enabled one time
	X	X	1	0	1	1	Enabled continuously
	X	X	1	1	1	1	Enabled continuously with period reload
Dual 32-bit chained	X	X	0	0	X	X	Disabled (default)
	X	X	0	1	1	1	Enabled one time
	X	X	1	0	1	1	Enabled continuously
	X	X	1	1	1	1	Reserved

Table 11-3433. Timer Enabling (continued)

Timer Mode	TIMER_TCR ENAMODE bits				TIMER_TGCR		Timer Status
	Bit 23	Bit 22	Bit 7	Bit 6	TIMHIRS	TIMLORS	
Dual 32-bit unchained	0	0	0	0	X	X	Both timers disabled (default)
• 32-bit timer	X	X	0	1	1	1	32-bit timer enabled one time
	1	0	1	0	1	1	32-bit timer enabled continuously
	1	1	1	1	1	1	Enabled continuously with period reload
• 32-bit timer with prescaler	0	1	X	X	1	X	32-bit timer enabled one time
	1	0	X	X	1	X	32-bit timer enabled continuously
	1	1	X	X	1	X	Enabled continuously with period reload

11.17.4.3.3 Timer Clock Source Selection

As shown in Table 11-3434 and Figure 11-1401, the timer clock sources for TIMLO and TIMHI is selected using the clock source (CLKSRC) bit and timer input enable TIMER_TCR[9] TIEN_LO bit.

Three clock sources are available to drive the timer clock:

- Internal clock, by setting TIMER_TCR[8] CLKSRC_LO = 0 and TIMER_TCR[9] TIEN_LO = 0.
- Internal clock gated by the timer input signal, by setting TIMER_TCR[8] CLKSRC_LO = 0 and TIMER_TCR[9] TIEN_LO = 1.
- External clock on the timer input pins (TINPL and TINPH), by setting TIMER_TCR[8] CLKSRC_LO = 1. This input signal is synchronized internally and can be inverted by setting the timer inverter control bit TIMER_TCR[2] INVINP_LO to 1.

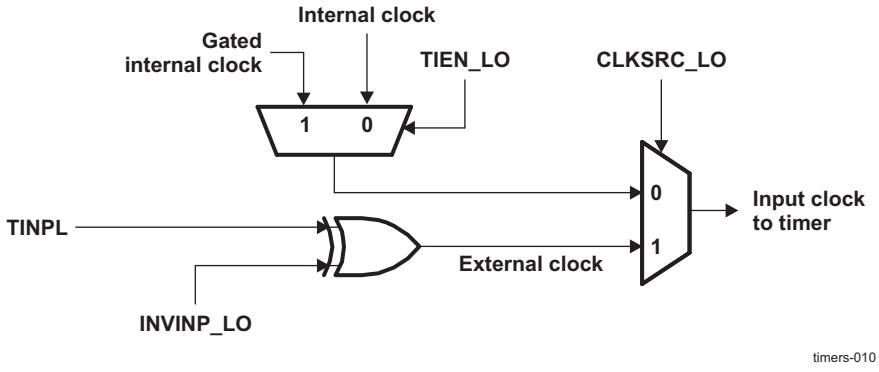
At reset, the clock source is the internal clock. The internal clock is derived from the DSP clock generator.

When the clock source is the gated internal clock, the timer starts counting when the timer input transitions from low to high and the timer stops counting when the timer input transitions from high to low.

Table 11-3434. Timer Clock Source Selection

CLKSRC_LO	TIEN_LO	Input Clock
0	0	Internal clock (default)
0	1	Gated internal clock
1	X	External clock on timer input (TINPL and TINPH)

Figure 11-1401. Timer Clock Source Block Diagram



11.17.4.4 Timer Output Mode Selection

The two basic timer output modes are pulse mode and clock mode. The timer output mode is selected using the clock/pulse mode bits TIMER_TCR[3] CP_LO and TIMER_TCR[19] CP_HI.

When in the pulse mode (TIMER_TCR[3] CP_LO or TIMER_TCR[19] CP_HI = 0), the pulse width bits (TIMER_TCR[5-4] PWID_LO or TIMER_TCR[21-20] PWID_HI) set the pulse width to 1, 2, 3, or 4 timer clock cycles. This pulse can be inverted by setting the timer output inverter control bits (TIMER_TCR[1] INVOUTP_LO or TIMER_TCR[17] INVOUTP_HI) to 1.

When in the clock mode (TIMER_TCR[3] CP_LO or TIMER_TCR[19] CP_HI = 1), the timer output signal has a 50% duty cycle. The signal toggles (from high-to-low or from low-to-high) each time the timer counter reaches the timer period.

The output signal of TIMLO is driven on both TIMER_TCR[0] TSTAT_LO and the timer output pin TOUTL. The output signal of TIMHI is driven on TIMER_TCR[16] TSTAT_HI only.

11.17.4.5 Timer Counting

The timer counter runs at the timer clock rate specified by the clock source bit (CLKSRC) in the timer control register (TIMER_TCR). Counting is enabled by setting the enabling mode (ENAMODE) bits in TIMER_TCR to 1h or 2h. When enabled, the timer counter starts incrementing until the counter reaches a value equal to the value in the timer period register. Once the timer counter matches the timer period:

- If the timer is set to enable one time (ENAMODE = 1h), the timer counter is reset to 0, then stops.
- If the timer is set to enable continuously (ENAMODE = 2h), the timer counter is reset to 0, then continues counting.
- If the timer is set to enable continuously with period reload (ENAMODE = 3h), the timer counter is reset to zero, reloads the period registers (TIMER_PRDLO and TIMER_PRDHI) with the value in the period reload registers (TIMER_RELLO and TIMER_RELHI), then continues counting.

Once the timer stops, if an external clock is used as the timer clock, the disable period must last at least one external clock period or the timer will not start counting again. When using the external clock, the count value is synchronized to the internal clock.

Note that when both the timer counter and timer period are cleared to 0, the timer can be enabled but the timer counter does not increment because the timer period is 0.

11.17.4.6 Timer Pulse Generation

The two basic timer output modes are pulse mode and clock mode. These modes can be select via the C/P_ bit. In the pulse mode, the PWID can set the pulse width to one or two or three or four timer clocks. This provides minimum pulse widths in the case in which TSTATx drives the timer output pin. This pulse can be inverted by setting INVOUT=1. The TSTATx directly drives the timer output. TSTATx and TOUTPx will be generated only when a compare of PRDx and TIMx register succeeds. They will not be generated when timeout happens due to external event when CAPMODE = 1 or due to TIMx read when READRSTMODE = 1.

11.17.4.7 Timer Reset Sources

The timer has two reset sources: hardware reset (TIMER_i_RST) and the timer reset bits (TIMER_TGCR[0] TIMLORS and TIMER_TGCR[1] TIMHIRS) in the timer global control register (TIMER_TGCR).

- When a hardware reset is asserted (TIMER_i_RST is set to 0), all the registers are set to their default values.
- When TIMER_i_RST = 1 and TIMER_TGCR[0] TIMLORS is cleared to 0, TIMER_TCR[0] TSTAT_LO is reset to 0 and TOUTL is in the high-impedance state.
- When TIMER_i_RST = 1 and TIMER_TGCR[1] TIMHIRS is cleared to 0, TIMER_TCR[16] TSTAT_HI is reset to 0.

11.17.4.8 Timer Input/ Output and TSTAT in GPIO

NOTE: In GPIO mode output for TINPL / TINPH and input for TOUTL / TOUTH is not supported.

Upon device reset, the timer input (TINPL / TINPH) and output (TOUTL / TOUTH) are not GPIO. But the timer input and output can operate as GPIO if being configured as GPIO function.

- When both timer input and output pins are configured in GPIO mode and both are the sources of interrupt and event, the input pin has higher priority than the output pin.
- Writing 1 to GPIO_ENILO(HI) or GPIO_ENOLO(HI) are ignored when TIMLO(HI)RS is 1.
- Writing 1 to TIMLO(HI)RS is ignored when GPIO_ENILO(HI) or GPIO_ENOLO(HI) is 1.
- TSTAT in GPIO mode holds the previous value.
- In GP Interrupt mode, one vbus clock period pulse of TINT is asserted when TINP goes to high (when GPINTx_INVI is 0), that is TINP doesn't drive TINT directly.
- GPIOLO(HI)_DIRI(DIRO) must be set to 0 when GPIO interrupt is used (GPINTLO(HI)_ENI(ENO) = 1).

11.17.4.9 Timer Interrupt Rate

To receive periodic interrupts, configure the timer to run in the continuous mode (ENAMODE = 2h or 3h). Each time the timer finishes counting, it can generate a timer interrupt for the CPU and a timer event for the DMA controller. The rate at which this occurs (the timer interrupt rate) depends on whether the timer has a prescaler.

If the timer does not have a prescaler, there is only one counter. When the timer counter reaches the programmed timer period, the timer generates an interrupt and a DMA event. Because the timer is in the continuous mode, one cycle after the timer counter matches the timer period, the timer counter is reset to 0 and starts counting again. The timer interrupt rate is:

$$TINT\ rate = \frac{Timer\ input\ clock\ rate}{Programmed\ timer\ period + 1}$$

timers-011

If a timer has a prescaler, there are two counters. One cycle after the prescale counter reaches the programmed prescale period, the timer counter is incremented by 1, and the prescale counter is reset to start counting again.

If the prescaler continues long enough, it increments the timer counter to the programmed timer period. At that time, the timer generates an interrupt and a DMA event. One cycle later (assuming the continuous mode), the timer counter is reset to 0 and starts counting again. The timer interrupt rate in this case is:

$$TINT\ rate = \frac{Timer\ input\ clock\ rate}{(Programmed\ prescale\ period + 1) \cdot (Programmed\ timer\ period + 1)}$$

timers-012

11.17.4.10 Timer Emulation Modes

The timer has an emulation management and clock speed register (TIMER_EMUMGT_CLKSPD). As shown in Table 11-3435, the TIMER_EMUMGT_CLKSPD[0] FREE and TIMER_EMUMGT_CLKSPD[1] SOFT bits determine how the timer responds to an emulation suspend event. An emulation suspend event corresponds to any type of emulator access to the DSP, such as a hardware or software breakpoint, a probe point, or a printf instruction.

Table 11-3435. Timer Emulation Modes Selection

FREE	SOFT	Emulation Mode
0	0	Default: The timer stops immediately.
0	1	The timer stops when the timer counter value increments and reaches the value in the timer period register.

Table 11-3435. Timer Emulation Modes Selection (continued)

FREE	SOFT	Emulation Mode
1	X	The timer runs free regardless of TIMER_EMUMGT_CLKSPD[1] SOFT bit status.

When using an internal clock as the timer clock source, the timer counter increments properly when single stepping. For example, the timer increments by one for each single step if the timer clock is equal to the CPU clock; or increments by one for every six single steps if the timer clock is equal to one-sixth of the CPU clock.

11.17.4.11 Timer Operation Boundary Conditions

Section 11.17.4.11.1 through Section 11.17.4.11.7 describe the boundary conditions affecting the timer operation.

11.17.4.11.1 Writing to and Reading From the Reserved Registers

Write the reset value to the reserved registers. Reading from the reserved registers returns zeros.

11.17.4.11.2 Timer Count = 0 and Timer Period = 0 (No Prescaler)

Consider a timer that has no prescaler:

- The 64-bit GP timer.
- TIMLO in the 32-bit dual timers configuration (unchained mode).

In the special case when timer count = 0 and timer period = 0:

- After a hardware reset and before the timer starts counting (ENAMODE bits = 0h), the timer output signal is held low.
- Once the timer is enabled, its behavior depends on the selected enabling mode (ENAMODE bits = 1h, 2h or 3h in the timer control register) and the selected timer output mode (CP bits = 0 or 1 in the timer control register). The options are summarized in Table 11-3436.
- The timer interrupt is not generated.

Table 11-3436. Timer Operation When Timer Count = 0 and Timer Period = 0

Timer Enabling Mode	Timer Output Mode	Timer Operation When Timer Count = 0 and Timer Period = 0 No Prescaler)
One-time mode (ENAMODE bits = 1h)	Pulse mode (CP bits = 0)	The timer output pulses once at the first timer clock cycle, and the timer stops counting at the next timer clock cycle. The pulse width is defined by the PWID bits of the TIMER_TCR.
	Clock mode (CP bits = 1)	The timer output toggles once at the first timer clock cycle. The timer stops counting at the next timer clock cycle.
Continuous mode (ENAMODE bits = 2h)	Pulse mode (CP bits = 0)	The timer output pulses once at the first timer clock cycle, and the timer continues to count up. Whenever the timer counter reaches its maximum value, it rolls around to 0, generating another pulse. The pulse width is defined by the PWID bits of the TIMER_TCR.
	Clock mode (CP bits = 1)	The timer output toggles once at the first timer clock cycle and then toggles with a frequency of half the timer clock frequency as the timer continues to count.
Continuous mode with reload (ENAMODE bits = 3h)	Pulse mode (CP bits = 0)	The timer output pulses once at the first timer clock cycle, and the timer continues to count up. Whenever the timer counter reaches its maximum value, it rolls around to 0, generating another pulse. The pulse width is defined by the PWID bits of the TIMER_TCR. The period registers are loaded with the reload registers.
	Clock mode (CP bits = 1)	The timer output toggles once at the first timer clock cycle and then toggles with a frequency of half the timer clock frequency as the timer continues to count. The period registers are loaded with the reload registers.

11.17.4.11.3 Timer Count = 0, Timer Period = 0, Prescale Count = 0, and Prescale Period = 0

Consider a timer that has a prescaler:

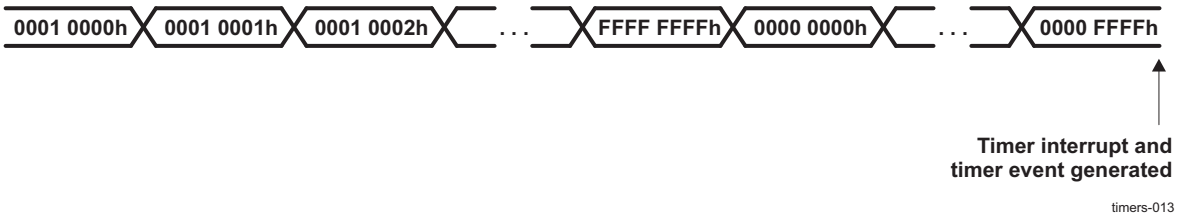
- The combination timer in the 32-bit dual timers chained mode.
- TIMHI in the 32-bit dual timer configuration (unchained mode).

In the special case when timer count = 0, timer period = 0, prescale count = 0, and prescale period = 0, the timer operates in the same manner as a non-prescaled timer with timer count = 0 and timer period = 0 (see Section 11.17.4.11.2).

11.17.4.11.4 Timer Counter Overflow

Timer counter overflow can happen when the timer counter register is set to a value greater than the value in the timer period register. The counter reaches its maximum value (FFFF FFFFh or FFFF FFFF FFFF FFFFh), rolls over to 0, and continues counting until it reaches the timer period. An example is shown in Figure 11-1402.

Figure 11-1402. 32-Bit Timer Counter Overflow Example



11.17.4.11.5 Writing to Registers of an Active Timer

Writes from the configuration bus to the timer registers are not allowed when the timer is active, except for stopping or resetting the timers. In the 64-bit and dual 32-bit timer modes, registers that are protected by hardware include TIMER_CNTLO, TIMER_CNTHI, TIMER_PRDLO, TIMER_PRDHI, TIMER_TGCR (except the TIMER_TGCR[0] TIMLORS and TIMER_TGCR[1] TIMHIRS bits), and TIMER_TCR (except the ENAMODE bits).

11.17.4.11.6 Small Timer Period Value in Pulse Mode

Small timer periods in pulse mode (CP = 0) can cause TSTAT to remain high when ENAMODE is not 0. This condition can occur when PRD . PWID + 1.

11.17.4.11.7 Reading the Counter Registers

Table 11-3437 summarizes how to read the counter registers. When reading the timer counter in 64-bit GP timer mode, the CPU must read the first 32-bit word from the TIMER_CNTLO registers. When this occurs, the timer takes a snapshot of the TIMER_CNTHI register and copies it into a shadow register CNTHIS. Note that reading TIMER_CNTHI instead of TIMER_CNTLO will not cause the timer to take a snap shot of the timer counters and copy them into the shadow registers.

Table 11-3437. Reading Counter Registers

Timer Mode	CPU	
64-bit timer	Read TIMER_CNTLO	Read TIMER_CNTLO Copy TIMER_CNTHI to CNTHIS
	Read TIMER_CNTHI	Read from CNTHIS
32-bit timer	Read TIMER_CNTLO	Read TIMER_CNTLO
	Read TIMER_CNTHI	Read TIMER_CNTHI

11.17.4.12 Watchdog Timer Mode

This subsection describes the capability of the timers to be configured as watchdog timer.

The timer can be configured also as a 64-bit watchdog timer. As a watchdog timer, it can be used to prevent system lock-up when the software becomes trapped in loops with no controlled exit. After a hardware reset, the timer is configured as a 64-bit GP timer and the watchdog mode is disabled. The timer then can be reconfigured as a watchdog timer using the timer mode bits TIMER_TGCR[3-2] TIMMODE and the Watchdog timer enable bit TIMER_WDTCR[14] WDEN. In the watchdog timer mode, the timer requires a special service sequence to be executed periodically. Without this periodic servicing, the timer counter increments until it matches the timer period and causes a watchdog timeout event or module reset.

When operating in Watchdog mode, the timer counts down to zero and generates an event. It is a requirement that software writes to the timer before the count expires, after which the count begins again. If the count ever reaches zero, the timer event output is asserted. This event can be used as a CPU exception (routed to the CPU interrupt controller), which requires a software exception handler. If routed as an exception to a DSP, software can post a request to the PLL Controller or the DSP LPSC to do a device or local reset.

Once the timer is configured as a watchdog timer, it cannot be reconfigured as a GP timer until a device reset occurs. When the timer counter matches the timer period, the timer generates two signals: an output signal and an interrupt signal (described in Section 11.17.4.12.1). Typically, one or the other is used, depending on whether an external or internal trigger is desired.

11.17.4.12.1 Timer Output Signal and Timer Interrupt Signal in Watchdog Timer Mode

When the periodic service sequence is not met, the timer counter increments until it matches the period and times out. During a timeout, a pulse is asserted on the timer output pin, and an internal maskable interrupt (INTL) is triggered. The timer output pin can be externally connected to the interrupt pin of an external device. Note that the timer pulse width must be configured to generate an active low pulse long enough for the external device to recognize it as an interrupt pulse. The pulse width is configured using the PWID bits of the timer control register (TIMER_TCR).

11.17.4.12.2 Watchdog Timer Mode Restrictions

The watchdog timer mode is selected and enabled when TIMER_TGCR[3-2] TIMMODE = 2h and TIMER_WDTCR[14] WDEN = 1. This mode has the following restrictions:

- No dual 32-bit timers mode
- No gated clock
- No external clock
- No one-time enabling
- No clock mode (only pulse mode)

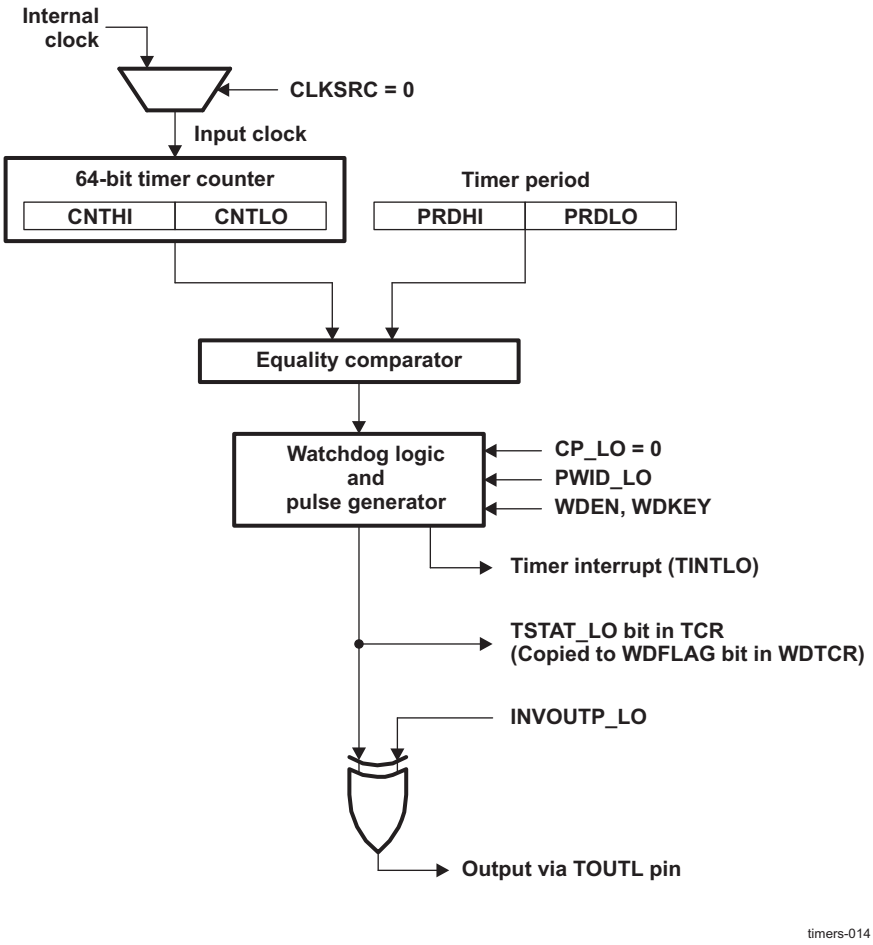
11.17.4.12.3 Watchdog Timer Mode Operation

Figure 11-1403 shows the timer when it is used in the watchdog timer mode. Note that in this mode, the timer clock must be set to the internal clock (TIMER_TCR[8] CLKSRC_LO = 0). The TIMER_TCR[3] CP_LO bit is forced to 0 because the pulse mode is required for the watchdog timer operation. The counter registers (TIMER_CNTLO and TIMER_CNTHI) form a 64-bit timer counter register and the period registers (TIMER_PRDLO and TIMER_PRDHI) form a 64-bit period register.

When the timer counter matches the timer period, the timer generates a watchdog timeout event. This event:

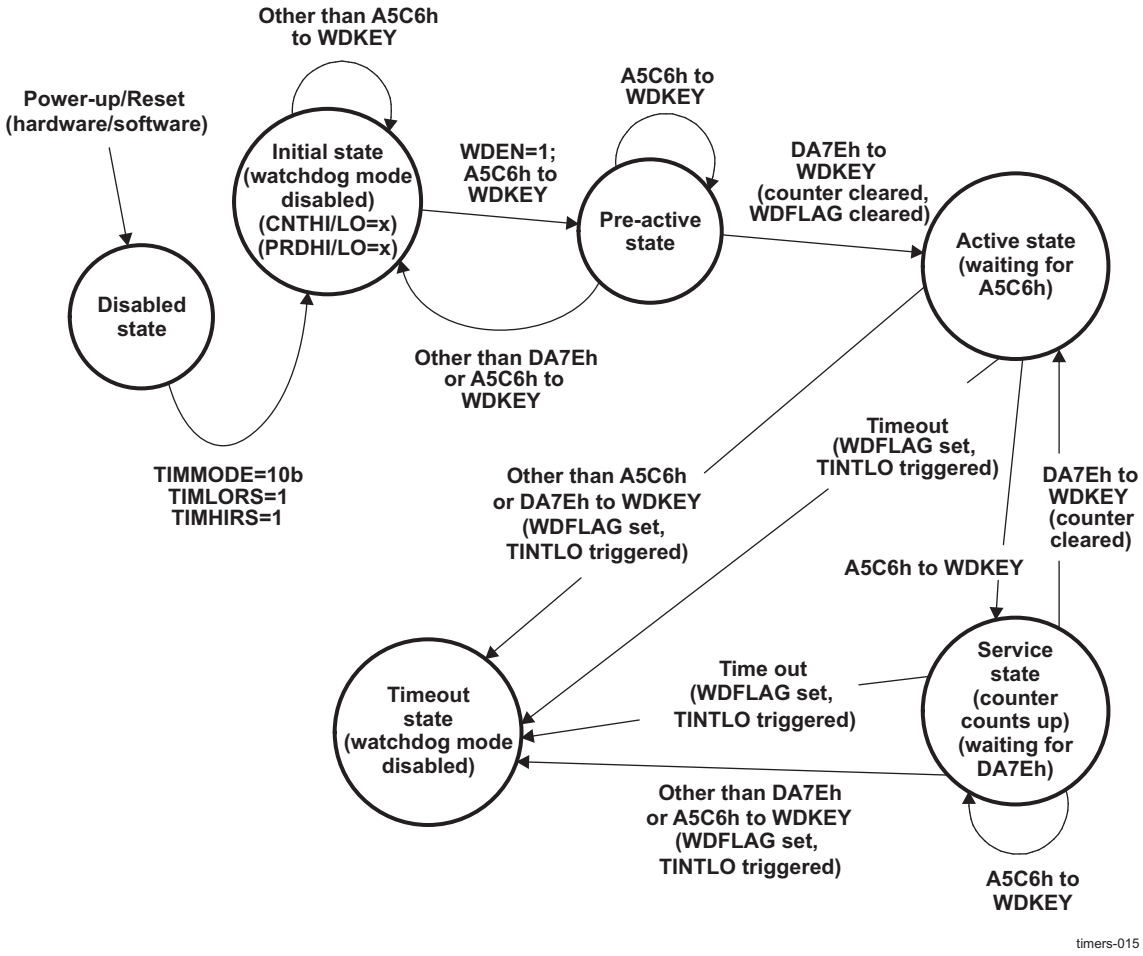
- Drives the timer output signal (TOUTL) and/or the timer interrupt signal (INTL).
- Resets the timer counter to 0.
- Sets the TIMER_TCR[0] TSTAT_LO bit, which is copied to the TIMER_WDTCR[15] WDFLAG bit.

Figure 11-1403. Timer in Watchdog Timer Mode



To activate the watchdog timer, a certain sequence of events must be followed, as shown in state diagram Figure 11-1404.

Figure 11-1404. Watchdog Timer Operation State Diagram



Once the watchdog timer is activated, it can be disabled only by a watchdog timeout event or by a hardware reset. A special key sequence is required to prevent the watchdog timer from being accidentally serviced while the software is trapped in a dead loop or by some other software failure.

To prevent a watchdog timeout event, the timer has to be serviced periodically by writing A5C6h followed by DA7Eh to the watchdog timer service key TIMER_WDTCR[31-16] WDKEY bits before the timer finishes counting up. Both A5C6h and DA7Eh are allowed to be written to the TIMER_WDTCR[31-16] WDKEY bits, but only the correct sequence of A5C6h followed by DA7Eh to the TIMER_WDTCR[31-16] WDKEY bits services the watchdog timer. Any other writes to the TIMER_WDTCR[31-16] WDKEY bits triggers the watchdog timeout event immediately. Writes to other bits in the TIMER_WDTCR are ignored when the watchdog timer is active (see Section 11.17.4.12.4).

When the watchdog timer is in the timeout state, the watchdog timer is disabled, the TIMER_WDTCR[14] WDEN bit is cleared to 0, and the timer is reset. After entering the timeout state, the watchdog timer cannot be enabled again until a hardware reset occurs.

After a hardware reset, the watchdog timer is disabled; however, reads or writes to the watchdog timer registers are allowed. Once the TIMER_WDTCR[14] WDEN bit is set and A5C6h is written to the TIMER_WDTCR[31-16] WDKEY bits, the watchdog timer enters the pre-active state. In the pre-active state:

- A write to TIMER_WDTCR is allowed only when the write comes with the correct key (A5C6h or DA7Eh) to the TIMER_WDTCR[31-16] WDKEY bits.
- A write of DA7Eh to the TIMER_WDTCR[31-16] WDKEY bits when the TIMER_WDTCR[14] WDEN bit is set to 1 resets the counters and activates the watchdog timer.

The TIMER_PRDHI, TIMER_PRDLO, TIMER_TCR, and TIMER_WDTCR registers must be configured before the watchdog timer enters the active state. The TIMER_WDTCR[14] WDEN bit must be set to 1 before writing DA7Eh to the TIMER_WDTCR[31-16] WDKEY bits in the pre-active state. Every time the watchdog timer is serviced by the correct TIMER_WDTCR[31-16] WDKEY sequence, the watchdog timer counter is automatically reset.

Before the watchdog timer enters the active state, the timer output signal is never asserted. Only the timer interrupt is asserted when the timer finishes counting up. In this case, the timer interrupt can be used to:

- Indicate that the watchdog timer is counting but is not in the active state.
- Generate a periodic interrupt, without having to service the watchdog timer.

The watchdog timer can always be disabled before entering the active state.

11.17.4.12.4 Watchdog Timer Register Write Protection

Once the watchdog timer enters the pre-active state, writes to registers TIMER_CNTHI, TIMER_CNTLO, TIMER_PRDHI, TIMER_PRDLO, TIMER_TCR, and TIMER_WDTCR (except for the TIMER_WDTCR[31-16] WDKEY bits) will have no effect. Writes to WDEN when the watchdog timer is in the timeout state have no effect.

The value A5C6h or DA7Eh must be written to the WDKEY bits depending on the current state (see Figure 11-1404). After the watchdog timer has entered the initial state, clearing the TIMER_TGCR[0] TIMLORS and TIMER_TGCR[1] TIMHIRS bits is prohibited.

11.17.5 Timers Programming Guide

11.17.5.1 Timers Low-Level Programming Models

This section describes the low-level hardware programming sequences for the configuration and use of the module.

11.17.5.1.1 Global Initialization

11.17.5.1.1.1 Global Initialization of Surrounding Modules

This section identifies the requirements for initializing the surrounding modules when the timers are to be used for the first time after a device reset. This initialization of surrounding modules is based on the integration and environment of the timers. For more information, see Section 11.17.3, *Timers Integration*, and Section 11.17.2, *Timers Environment*. Table 11-3438 summarizes the surrounding modules.

Table 11-3438. Global Initialization of Surrounding Modules

Surrounding Modules	Comments
DSP_INTC	Device INTCs must be configured to enable the interrupt request generation. For information about enabling DSP interrupts, see Section 6.2, *C66x DSP Subsystem*.
CIC	Device INTCs must be configured to enable the interrupt request generation. For information about enabling CIC interrupts, see Chapter 9, *Interrupts*.
ARM_GIC	Device INTCs must be configured to enable the interrupt request generation. For information about enabling ARM GIC interrupts, see Section 6.1, *Arm Cortex-A15 Subsystem*.
EDMACC_0 and EDMACC_1	DMA controllers (EDMACC_0 and EDMACC_1) configuration must be done to enable the module DMA channel request. See Chapter 10, *Enhanced Direct Memory Access (EDMA) Controller*.
PLL Controller	PLL Controller configuration must be done to enable the module clocks. See Section 5.4.5.3, *PLL Controller*.
Timer pin manager	Timer pin manager configuration must be done to enable the module output pins multiplexing. See Section 5.1.3.1.9, *Timer Pin Manager Control Registers*.

11.17.5.1.1.2 Timers Module Global Initialization

11.17.5.1.1.2.1 Main Sequence – Timers Module Global Initialization

After a hardware reset, the enabling mode bits TIMER_TCR[23-33] ENAMODE are cleared to 0 and the timer is disabled. The timer counter and period registers are cleared to 0. The timer can be configured to the desired mode by programming the control registers, TIMER_TCR and (in the case of the watchdog timer mode) TIMER_WDTCR.

Table 11-3439 identifies the main steps for initializing the timers module when the module is to be used for the first time.

Table 11-3439. Timers Module Global Initialization

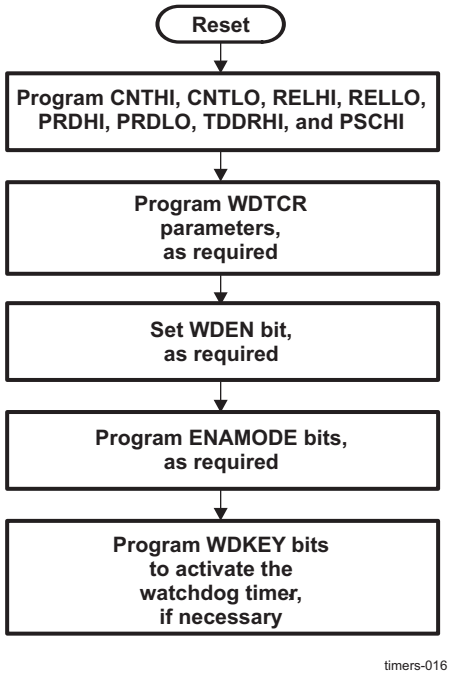
Step	Register/Bit Field/Programming Model	Value
Execute hardware reset.	TIMER_TCR[23-22] ENAMODE	0h
Write the timer counter, period reload (if used) and period values	Write the values to TIMER_CNTHI and/or TIMER_CNTLO, TIMER_RELHI, TIMER_RELLO and TIMER_PRDHI and/or TIMER_PRDLO registers	-h
Is the 4-bit prescaler to be used?	Write the values to the TIMER_TGCR[15-12] TDDRHI bits and TIMER_TGCR[11-8] PSCHI bits	-h
Set the remaining control bits to the required state		-h
Is the timer to be used as watchdog timer?	Set TIMER_WDTCR[14] WDEN bit	1h

Table 11-3439. Timers Module Global Initialization (continued)

Step	Register/Bit Field/Programming Model	Value
To start the timer, set the ENAMODE bits to use the timer as a continuous interrupt generator	ENAMODE bits	2h or 3h
or	or	
as a one-time counter.	ENAMODE bits	1h
Is the watchdog timer mode to be selected?	Program the TIMER_WDTCR[31-16] WDKEY bits	A5C6h DA7Eh

Figure 11-1405 shows a typical timer initialization:

Figure 11-1405. Timer Initialization



11.17.5.2 Additional requirements to Watchdog Timer Mode

Few additional requirements apply to watchdog timer mode:

1. When CPU is in emulation halt mode, the corresponding watchdog timer must be suspended. This is to ensure no watchdog resets during debug. Emulation suspend outputs from CPU (DSP or A15 core) are connected to TIMER_0 and TIMER_5 to enable this feature.
2. When CPU is reset, the corresponding watchdog timer also needs to be reset. For DSP this is ensured by using the same LPSC to control both the DSP and the corresponding watchdog timer. The Arm internal PSC supports a reset and clock stop handshake interface is connected to Timer clock stop/clock gating control.
3. When CPU is clock gated or powered down using the LPSC, the corresponding watchdog timer should also be clock gated. For DSP this is ensured by using the same LPSC to control both the DSP and the corresponding watchdog timer.
4. For A15 core internal clock gating mode like WFI mode, it is the responsibility of software to put the corresponding timer in the clock gated state.

11.17.6 Timers Registers

Table 11-3441 and Table 11-3442 list the memory-mapped registers for the Timers. All register offset addresses not listed in Table 11-3441 and Table 11-3442 should be considered as reserved locations and the register contents should not be modified.

NOTE: Timers' IDLE mode is not supported in this family of devices.

Table 11-3440. Timers Instances

Instance	Base Address
TIMER_0	0220 0000h
TIMER_1	0221 0000h
TIMER_2	0222 0000h
TIMER_3	0223 0000h
TIMER_4	0224 0000h
TIMER_5	0225 0000h
TIMER_6	0226 0000h

Table 11-3441. Timers Registers

Offset	Acronym	Register Name	TIMER_0 Physical Address	TIMER_1 Physical Address	TIMER_2 Physical Address	Section
0h	TIMER_PID12	Peripheral Identification Register 12	0220 0000h	0221 0000h	0222 0000h	Section 11.17.6.1
4h	TIMER_EMUMGT_CLKSPD	Emulation Management and Clock Speed Register	0220 0004h	0221 0004h	0222 0004h	Section 11.17.6.2
8h	TIMER_GPINT_EN	GPIO Interrupt Control/Enable Register	0220 0008h	0221 0008h	0222 0008h	Section 11.17.6.3
Ch	TIMER_GPDIR_DAT	GPIO Direction/Data Register	0220 000Ch	0221 000Ch	0222 000Ch	Section 11.17.6.4
10h	TIMER_CNTLO	Timer Counter Register Low	0220 0010h	0221 0010h	0222 0010h	Section 11.17.6.5
14h	TIMER_CNTHI	Timer Counter Register High	0220 0014h	0221 0014h	0222 0014h	Section 11.17.6.6
18h	TIMER_PRDLO	Timer Period Register Low	0220 0018h	0221 0018h	0222 0018h	Section 11.17.6.7
1Ch	TIMER_PRDHI	Timer Period Register High	0220 001Ch	0221 001Ch	0222 001Ch	Section 11.17.6.8
20h	TIMER_TCR	Timer Control Register	0220 0020h	0221 0020h	0222 0020h	Section 11.17.6.9
24h	TIMER_TGCR	Timer Global Control Register	0220 0024h	0221 0024h	0222 0024h	Section 11.17.6.10
28h	TIMER_WDTCR	Watchdog Timer Control Register	0220 0028h	0221 0028h	0222 0028h	Section 11.17.6.11
34h	TIMER_RELLO	Timer Reload Register Low	0220 0034h	0221 0034h	0222 0034h	Section 11.17.6.12
38h	TIMER_RELHI	Timer Reload Register High	0220 0038h	0221 0038h	0222 0038h	Section 11.17.6.13
3Ch	TIMER_CAPLO	Timer Capture (Shadow) Register Low	0220 003Ch	0221 003Ch	0222 003Ch	Section 11.17.6.14
40h	TIMER_CAPHI	Timer Capture (Shadow) Register High	0220 0040h	0221 0040h	0222 0040h	Section 11.17.6.15
44h	TIMER_INTCTL_STAT	Timer Interrupt Control and Status Register	0220 0044h	0221 0044h	0222 0044h	Section 11.17.6.16

Table 11-3442. Timers Registers

Offset	Acronym	Register Name	TIMER_3 Physical Address	TIMER_4 Physical Address	TIMER_5 Physical Address	TIMER_6 Physical Address	Section
0h	TIMER_PID12	Peripheral Identification Register 12	0223 0000h	0224 0000h	0225 0000h	0226 0000h	Section 11.17.6.1
4h	TIMER_EMUMGT_CLKSPD	Emulation Management and Clock Speed Register	0223 0004h	0224 0004h	0225 0004h	0226 0004h	Section 11.17.6.2
8h	TIMER_GPINT_EN	GPIO Interrupt Control/Enable Register	0223 0008h	0224 0008h	0225 0008h	0226 0008h	Section 11.17.6.3
Ch	TIMER_GPDIR_DAT	GPIO Direction/Data Register	0223 000Ch	0224 000Ch	0225 000Ch	0226 000Ch	Section 11.17.6.4
10h	TIMER_CNTLO	Timer Counter Register Low	0223 0010h	0224 0010h	0225 0010h	0226 0010h	Section 11.17.6.5
14h	TIMER_CNTHI	Timer Counter Register High	0223 0014h	0224 0014h	0225 0014h	0226 0014h	Section 11.17.6.6
18h	TIMER_PRDLO	Timer Period Register Low	0223 0018h	0224 0018h	0225 0018h	0226 0018h	Section 11.17.6.7
1Ch	TIMER_PRDHI	Timer Period Register High	0223 001Ch	0224 001Ch	0225 001Ch	0226 001Ch	Section 11.17.6.8
20h	TIMER_TCR	Timer Control Register	0223 0020h	0224 0020h	0225 0020h	0226 0020h	Section 11.17.6.9
24h	TIMER_TGCR	Timer Global Control Register	0223 0024h	0224 0024h	0225 0024h	0226 0024h	Section 11.17.6.10
28h	TIMER_WDTCR	Watchdog Timer Control Register	0223 0028h	0224 0028h	0225 0028h	0226 0028h	Section 11.17.6.11
34h	TIMER_RELLO	Timer Reload Register Low	0223 0034h	0224 0034h	0225 0034h	0226 0034h	Section 11.17.6.12
38h	TIMER_RELHI	Timer Reload Register High	0223 0038h	0224 0038h	0225 0038h	0226 0038h	Section 11.17.6.13
3Ch	TIMER_CAPLO	Timer Capture (Shadow) Register Low	0223 003Ch	0224 003Ch	0225 003Ch	0226 003Ch	Section 11.17.6.14
40h	TIMER_CAPHI	Timer Capture (Shadow) Register High	0223 0040h	0224 0040h	0225 0040h	0226 0040h	Section 11.17.6.15
44h	TIMER_INTCTL_STAT	Timer Interrupt Control and Status Register	0223 0044h	0224 0044h	0225 0044h	0226 0044h	Section 11.17.6.16

11.17.6.1 TIMER_PID12 Register (Offset = 0h) [reset = 4472 020Ch]

TIMER_PID12 is shown in Figure 11-1406 and described in Table 11-3444.

This register is intended to store versioning information used to identify the specific GP/WD timer peripheral implemented in a particular device. All bits within this register are read only.

Table 11-3443. TIMER_PID12 Instances

Instance	Physical Address
TIMER_0	0220 0000h
TIMER_1	0221 0000h
TIMER_2	0222 0000h
TIMER_3	0223 0000h
TIMER_4	0224 0000h
TIMER_5	0225 0000h
TIMER_6	0226 0000h

Figure 11-1406. TIMER_PID12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-4472 020Ch																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3444. TIMER_PID12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	4472 020Ch	TI internal data. Identifies revision of peripheral.

Table 11-3445. Register Call Summary for TIMER_PID12

Timers Registers
- Timers Registers: [0][1]
- TIMER_PID12 Register (Offset = 0h) [reset = 4472 020Ch]: [0]

11.17.6.2 TIMER_EMUMGT_CLKSPD Register (Offset = 4h) [reset = 0h]

TIMER_EMUMGT_CLKSPD is shown in Figure 11-1407 and described in Table 11-3447.

The TIMER_EMUMGT_CLKSPD register contains the FREE and SOFT bits that determine how the timer responds to an emulation suspend event. An emulation suspend event corresponds to any type of emulator access to the DSP, such as a hardware or software breakpoint, a probe point, or a printf instruction. For additional emulation information, see Section 11.17.4.10.

The CLKDIV field of this register also can be read to identify the ratio of the CPU clock to the timer input clock. For example, in devices where the internal timer clock frequency is equal to the CPU frequency divided by 6, the CLKDIV field will read as 6 on those devices.

Table 11-3446. TIMER_EMUMGT_CLKSPD Instances

Instance	Physical Address
TIMER_0	0220 0004h
TIMER_1	0221 0004h
TIMER_2	0222 0004h
TIMER_3	0223 0004h
TIMER_4	0224 0004h
TIMER_5	0225 0004h
TIMER_6	0226 0004h

Figure 11-1407. TIMER_EMUMGT_CLKSPD Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED				CLKDIV			
R/W-0h				R-nh			
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						SOFT	FREE
R/W-0h						R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset (1)

Table 11-3447. TIMER_EMUMGT_CLKSPD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R/W	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

(1) The reset value of this field is based on the ratio of the CPU clock to the timer internal clock.

Table 11-3447. TIMER_EMUMGT_CLKSPD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-16	CLKDIV	R	-h	Clock divide-down ratio bits. Defines the ratio of the CPU clock to the timer input clock. The CLKDIV bits are read-only bits. • 1h = Internal clock source for the timer is the CPU clock divided by 1. • 2h = Internal clock source for the timer is the CPU clock divided by 2. • 3h = Reserved • 4h= Internal clock source for the timer is the CPU clock divided by 4. • 5h = Reserved • 6h = Internal clock source for the timer is the CPU clock divided by 6. • 7h = Reserved • 8h = Internal clock source for the timer is the CPU clock divided by 8. • 9h-15h = Reserved
15-2	RESERVED	R/W	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
1	SOFT	R/W	0h	Used in conjunction with FREE bit to determine how the timer responds to an emulation suspend event. When the FREE bit is 0, the SOFT bit selects the timer response. • 0 = The timer stops immediately. • 1 = The timer stops when the timer counter register increments and reaches the value in the timer
0	FREE	R/W	0h	Used in conjunction with SOFT bit to determine how the timer responds to an emulation suspend event. When the FREE bit is 0, the SOFT bit selects the timer response. • 0 = The SOFT bit selects the timer response. • 1 = The timer runs free, regardless of the value of the SOFT bit.

Table 11-3448. Register Call Summary for TIMER_EMUMGT_CLKSPD

Timers Registers • TIMER_EMUMGT_CLKSPD Register (Offset = 4h) [reset = 0h]: [0][1] • Timers Registers: [0][1]
Timers Functional Description • Timer Emulation Modes: [0][1][2][3]

11.17.6.3 TIMER_GPINT_EN Register (Offset = 8h) [reset = 0h]

TIMER_GPINT_EN is shown in Figure 11-1408 and described in Table 11-3450.

NOTE: In GPIO mode output for TINPL / TINPH and input for TOUTL / TOUTH is not supported.

Table 11-3449. TIMER_GPINT_EN Instances

Instance	Physical Address
TIMER_0	0220 0008h
TIMER_1	0221 0008h
TIMER_2	0222 0008h
TIMER_3	0223 0008h
TIMER_4	0224 0008h
TIMER_5	0225 0008h
TIMER_6	0226 0008h

Figure 11-1408. TIMER_GPINT_EN Register

31	30	29	28	27	26	25	24
RESERVED						GPIO_ENO34	GPIO_ENI34
R/W-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED						GPIO_ENO12	GPIO_ENI12
R/W-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED		GPINT34_INVO	GPINT34_INVI	RESERVED		GPINT34_ENO	GPINT34_ENI
R/W-0h		R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED		GPINT12_INVO	GPINT12_INVI	RESERVED		GPINT12_ENO	GPINT12_ENI
R/W-0h		R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3450. TIMER_GPINT_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R/W	0h	The reserved bit location is always read as 0. A value written to this field has no effect.
25	GPIO_ENO34	R/W	0h	Enable the timer output TOUTH in GPIO mode. GPIO_ENO34=0: timer output functions as a timer output. GPIO_ENO34=1: the timer output is enabled as GPIO.
24	GPIO_ENI34	R/W	0h	Enable the timer input TINPH in GPIO mode. GPIO_ENI34=0: timer input functions as a timer input. GPIO_ENI34=1: the timer input is enabled as GPIO.
23-18	RESERVED	R/W	0h	The reserved bit location is always read as 0. A value written to this field has no effect.
17	GPIO_ENO12	R/W	0h	Enable the timer output TOUTL in GPIO mode. GPIO_ENO12=0: timer output functions as a timer output. GPIO_ENO12=1: the timer output is enabled as GPIO.
16	GPIO_ENI12	R/W	0h	Enable the timer input TINPL in GPIO mode. GPIO_ENI12=0: timer input functions as a timer input. GPIO_ENI12=1: the timer input is enabled as GPIO.
15-14	RESERVED	R/W	0h	The reserved bit location is always read as 0. A value written to this field has no effect.

Table 11-3450. TIMER_GPINT_EN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	GPINT34_INVO	R/W	0h	Invert bit for timer output TOUTH when it is used to source an interrupt or event (GPINT34_ENO=1). GPINT34_INVO=0: Rising of Timer output generate interrupt or event. GPINT34_INVO=1: Falling of Timer output generate interrupt or event. This bit has no effect when GPINT34_ENO=0.
12	GPINT34_INVI	R/W	0h	Invert bit for timer input TINPH when it is used to source an interrupt or event (GPINT34_ENI=1). GPINT34_INVI=0: Rising of Timer input generate interrupt or event. GPINT34_INVI=1: Falling of Timer input generate interrupt or event. This bit has no effect when GPINT34_ENI=0.
11-10	RESERVED	R/W	0h	The reserved bit location is always read as 0. A value written to this field has no effect.
9	GPINT34_ENO	R/W	0h	Enable the timer output TOUTH to source the interrupt or event in GPIO mode. GPINT34_ENO=0: causes the interrupt or event to be sourced normally by the timer. GPINT34_ENO=1: causes the timer output to source an interrupt or event.
8	GPINT34_ENI	R/W	0h	Enable the timer input TINPH to source the interrupt or event in GPIO mode. GPINT34_ENI=0: causes the interrupt or event to be sourced normally by the timer. GPINT34_ENI=1: causes the timer input to source an interrupt or event.
7-6	RESERVED	R/W	0h	The reserved bit location is always read as 0. A value written to this field has no effect.
5	GPINT12_INVO	R/W	0h	Invert bit for timer output TOUTL when it is used to source an interrupt or event (GPINT12_ENO=1). GPINT12_INVO=0: Timer output interrupt or event is not inverted. GPINT12_INVO=1: Timer output interrupt or event is inverted. This bit has no effect when GPINT12_ENO=0.
4	GPINT12_INVI	R/W	0h	Invert bit for timer input TINPL when it is used to source an interrupt or event (GPINT12_ENI=1). GPINT12_INVI=0: Timer input interrupt or event is not inverted. GPINT12_INVI=1: Timer input interrupt or event is inverted. This bit has no effect when GPINT12_ENI=0.
3-2	RESERVED	R/W	0h	The reserved bit location is always read as 0. A value written to this field has no effect.
1	GPINT12_ENO	R/W	0h	Enable the timer output TOUTL to source the interrupt or event in GPIO mode. GPINT12_ENO=0: causes the interrupt or event to be sourced normally by the timer. GPINT12_ENO=1: causes the timer output to source an interrupt or event.
0	GPINT12_ENI	R/W	0h	Enable the timer input TINPL to source the interrupt or event in GPIO mode. GPINT12_ENI=0: causes the interrupt or event to be sourced normally by the timer. GPINT12_ENI=1: causes the timer input to source an interrupt or event.

Table 11-3451. Register Call Summary for TIMER_GPINT_EN

Timers Registers

- TIMER_GPINT_EN Register (Offset = 8h) [reset = 0h]: [0]
- Timers Registers: [0][1]

11.17.6.4 TIMER_GPDIR_DAT Register (Offset = Ch) [reset = 0h]

TIMER_GPDIR_DAT is shown in Figure 11-1409 and described in Table 11-3453.

NOTE: In GPIO mode output for TINPL / TINPH and input for TOUTL / TOUTH is not supported.

Table 11-3452. TIMER_GPDIR_DAT Instances

Instance	Physical Address
TIMER_0	0220 000Ch
TIMER_1	0221 000Ch
TIMER_2	0222 000Ch
TIMER_3	0223 000Ch
TIMER_4	0224 000Ch
TIMER_5	0225 000Ch
TIMER_6	0226 000Ch

Figure 11-1409. TIMER_GPDIR_DAT Register

31	30	29	28	27	26	25	24
RESERVED						GPIO_DIRO34	GPIO_DIRI34
R/W-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED						GPIO_DIRO12	GPIO_DIRI12
R/W-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED						GPIO_DATO34	GPIO_DATI34
R/W-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED						GPIO_DATO12	GPIO_DATI12
R/W-0h						R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3453. TIMER_GPDIR_DAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R/W	0h	The reserved bit location is always read as 0. A value written to this field has no effect.
25	GPIO_DIRO34	R/W	0h	Controls the direction of the timer output TOUTH in GPIO mode. It must be set to 0 to be used as input in GPIO interrupt mode. GPIO_DIRO34=0: timer output functions as a GPIO input. GPIO_DIRO34=1: timer output functions as a GPIO output.
24	GPIO_DIRI34	R/W	0h	Controls the direction of the timer input TINPH in GPIO mode. It must be set to 0 to be used as input in GPIO interrupt mode. GPIO_DIRI34=0: timer input functions as a GPIO input. GPIO_DIRI34=1: timer input functions as a GPIO output.
23-18	RESERVED	R/W	0h	The reserved bit location is always read as 0. A value written to this field has no effect.
17	GPIO_DIRO12	R/W	0h	Controls the direction of the timer output TOUTL in GPIO mode. It must be set to 0 to be used as input in GPIO interrupt mode. GPIO_DIRO12=0: timer output functions as a GPIO input. GPIO_DIRO12=1: timer output functions as a GPIO output.
16	GPIO_DIRI12	R/W	0h	Controls the direction of the timer input TINPL in GPIO mode. It must be set to 0 to be used as input in GPIO interrupt mode. GPIO_DIRI12=0: timer input functions as a GPIO input. GPIO_DIRI12=1: timer input functions as a GPIO output.

Table 11-3453. TIMER_GPDIR_DAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-10	RESERVED	R/W	0h	The reserved bit location is always read as 0. A value written to this field has no effect.
9	GPIO_DATO34	R/W	0h	When GPIO_DIRO34=0: timer output TOUTH functions as a GPIO input, the logic value read in this bit is equal to the logic level present at the timer input. Writes to this bit have no effect. The GP input is sampled with the VBUS clock. When GPIO_DIRO34=1: timer output TOUTH functions as a GPIO output, the GP output is driven to the logic level which is written to and stored in this bit. The logic level of the GP output can also be read back at this bit.
8	GPIO_DATI34	R/W	0h	When GPIO_DIRI34=0: timer input TINPH functions as a GPIO input, the logic value read in this bit is equal to the logic level present at the timer input. Writes to this bit have no effect. The GP input is sampled with the VBUS clock. When GPIO_DIRI=1: timer input functions as a GPIO output, the GP output is driven to the logic level which is written to and stored in this bit. The logic level of the GP output can also be read back at this bit.
7-2	RESERVED	R/W	0h	The reserved bit location is always read as 0. A value written to this field has no effect.
1	GPIO_DATO12	R/W	0h	When GPIO_DIRO12=0: timer output TOUTL functions as a GPIO input, the logic value read in this bit is equal to the logic level present at the timer input. Writes to this bit have no effect. The GP input is sampled with the VBUS clock. When GPIO_DIRO=1: timer output TOUTL functions as a GPIO output, the GP output is driven to the logic level which is written to and stored in this bit. The logic level of the GP output can also be read back at this bit.
0	GPIO_DATI12	R/W	0h	When GPIO_DIRI12=0: timer input TINPL functions as a GPIO input, the logic value read in this bit is equal to the logic level present at the timer input. Writes to this bit have no effect. The GP input is sampled with the VBUS clock. When GPIO_DIRI12=1: timer input TINPL functions as a GPIO output, the GP output is driven to the logic level which is written to and stored in this bit. The logic level of the GP output can also be read back at this bit.

Table 11-3454. Register Call Summary for TIMER_GPDIR_DAT

Timers Registers
- Timers Registers: [0][1]
- TIMER_GPDIR_DAT Register (Offset = Ch) [reset = 0h]: [0]

11.17.6.5 TIMER_CNTLO Register (Offset = 10h) [reset = 0h]

TIMER_CNTLO is shown in Figure 11-1410 and described in Table 11-3456.

The timer counter registers (TIMER_CNTLO and TIMER_CNTHI) are 32-bit-wide registers that can be used in conjunction to form a 64-bit counter, or separately as 32-bit counters. The use of these registers depends on the configuration of the timer. In the 64-bit general-purpose timer mode and watchdog mode, the two registers work as a single 64-bit counter. The 64-bit counter increments when the timer is enabled to count.

In a dual 32-bit timer mode, the counter registers work as separate 32-bit registers. These two register pairs can be configured as chained or unchained.

A hardware reset clears both counter registers, but software resets do not affect them. When the TIMLORS bit is cleared, TIMER_CNTLO keeps its current value, and when the TIMHIRS bit is cleared, TIMER_CNTHI keeps its current value.

Table 11-3455. TIMER_CNTLO Instances

Instance	Physical Address
TIMER_0	0220 0010h
TIMER_1	0221 0010h
TIMER_2	0222 0010h
TIMER_3	0223 0010h
TIMER_4	0224 0010h
TIMER_5	0225 0010h
TIMER_6	0226 0010h

Figure 11-1410. TIMER_CNTLO Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3456. TIMER_CNTLO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CNT	R/W	0h	Value = 0000 0000h-FFFF FFFFh Counter register. This register is a 32-bit prescale counter or 32-bit timer counter, or one half of a 64-bit timer counter.

Table 11-3457. Register Call Summary for TIMER_CNTLO

Timers Registers

- TIMER_PRDLO Register (Offset = 18h) [reset = 0h]: [0]
- Timers Registers: [0][1]
- TIMER_RELLO Register (Offset = 34h) [reset = 0h]: [0]
- TIMER_CNTLO Register (Offset = 10h) [reset = 0h]: [0][1][2]
- TIMER_CAPLO Register (Offset = 3Ch) [reset = 0h]: [0][1]
- TIMER_TCR Register (Offset = 20h) [reset = 0h]: [0][1][2]
- TIMER_TGCR Register (Offset = 24h) [reset = 0h]: [0]

Table 11-3457. Register Call Summary for TIMER_CNTLO (continued)

Timers Functional Description • Event Capture Mode: [0][1][2] • Timer Counter Register Read Reset Mode: [0][1] • Watchdog Timer Mode Operation: [0] • Timer Capture Registers: [0] • Counter, Reload and Period Registers Used in GP Timer Modes: [0][1][2][3] • Watchdog Timer Register Write Protection: [0] • Writing to Registers of an Active Timer: [0] • Reading the Counter Registers: [0][1][2][3][4][5] • 64-Bit Timer Mode: [0] • Chained Mode: [0] • Unchained Mode: [0]
Timers Programming Guide • Timers Module Global Initialization: [0]

11.17.6.6 TIMER_CNTHI Register (Offset = 14h) [reset = 0h]

TIMER_CNTHI is shown in Figure 11-1411 and described in Table 11-3459.

Table 11-3458. TIMER_CNTHI Instances

Instance	Physical Address
TIMER_0	0220 0014h
TIMER_1	0221 0014h
TIMER_2	0222 0014h
TIMER_3	0223 0014h
TIMER_4	0224 0014h
TIMER_5	0225 0014h
TIMER_6	0226 0014h

Figure 11-1411. TIMER_CNTHI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3459. TIMER_CNTHI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CNT	R/W	0h	Value = 0000 0000h-FFFF FFFFh Counter register. This register is a 32-bit prescale counter or 32-bit timer counter, or one half of a 64-bit timer counter.

Table 11-3460. Register Call Summary for TIMER_CNTHI

Timers Registers
- TIMER_PRDLO Register (Offset = 18h) [reset = 0h]: [0]
- Timers Registers: [0][1]
- TIMER_RELLO Register (Offset = 34h) [reset = 0h]: [0]
- TIMER_CNTLO Register (Offset = 10h) [reset = 0h]: [0][1]
- TIMER_CAPLO Register (Offset = 3Ch) [reset = 0h]: [0][1]
- TIMER_CNTHI Register (Offset = 14h) [reset = 0h]: [0]
- TIMER_TCR Register (Offset = 20h) [reset = 0h]: [0][1][2]
- TIMER_TGCR Register (Offset = 24h) [reset = 0h]: [0][1][2][3]

Timers Functional Description
- Timer Counter Register Read Reset Mode: [0][1]
- Watchdog Timer Mode Operation: [0]
- 64-Bit Timer Mode: [0]
- Counter, Reload and Period Registers Used in GP Timer Modes: [0][1][2][3]
- Watchdog Timer Register Write Protection: [0]
- Writing to Registers of an Active Timer: [0]
- Chained Mode: [0]
- Unchained Mode: [0]
- Reading the Counter Registers: [0][1][2][3][4][5]
- Timer Capture Registers: [0][1]

Timers Programming Guide
- Timers Module Global Initialization: [0]

11.17.6.7 TIMER_PRDLO Register (Offset = 18h) [reset = 0h]

TIMER_PRDLO is shown in Figure 11-1412 and described in Table 11-3462.

The timer period registers (TIMER_PRDLO and TIMER_PRDHI) are 32-bit-wide registers which can be used in conjunction to form a single 64-bit period register or separately as 32-bit period registers.

In a 64-bit general-purpose timer mode and watchdog mode, all 64 period bits contain the number of timer input clock cycles to count. This number controls the frequency of the timer output. In a 32-bit dual timer mode, the period registers work as separate 32-bit registers.

These two registers are used in conjunction with the two counter-registers, TIMER_CNTHI and TIMER_CNTLO.

Table 11-3461. TIMER_PRDLO Instances

Instance	Physical Address
TIMER_0	0220 0018h
TIMER_1	0221 0018h
TIMER_2	0222 0018h
TIMER_3	0223 0018h
TIMER_4	0224 0018h
TIMER_5	0225 0018h
TIMER_6	0226 0018h

Figure 11-1412. TIMER_PRDLO Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3462. TIMER_PRDLO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PRD	R/W	0h	Value = 0000 0000h-FFFF FFFFh Period register. This register contains the full timer period for a 32-bit timer configuration or half the timer period for a 64-bit timer configuration.

Table 11-3463. Register Call Summary for TIMER_PRDLO

Timers Registers • TIMER_PRDLO Register (Offset = 18h) [reset = 0h]: [0][1] • Timers Registers: [0][1] • TIMER_TCR Register (Offset = 20h) [reset = 0h]: [0][1]
Timers Functional Description • Event Capture Mode: [0] • Timer Counter Register Read Reset Mode: [0] • Watchdog Timer Mode Operation: [0][1] • 64-Bit Timer Mode: [0] • Counter, Reload and Period Registers Used in GP Timer Modes: [0][1][2][3] • Watchdog Timer Register Write Protection: [0] • Writing to Registers of an Active Timer: [0] • Chained Mode: [0] • Unchained Mode: [0] • Timer Counting: [0]
Timers Programming Guide • Timers Module Global Initialization: [0]

11.17.6.8 TIMER_PRDHI Register (Offset = 1Ch) [reset = 0h]

TIMER_PRDHI is shown in Figure 11-1413 and described in Table 11-3465.

Table 11-3464. TIMER_PRDHI Instances

Instance	Physical Address
TIMER_0	0220 001Ch
TIMER_1	0221 001Ch
TIMER_2	0222 001Ch
TIMER_3	0223 001Ch
TIMER_4	0224 001Ch
TIMER_5	0225 001Ch
TIMER_6	0226 001Ch

Figure 11-1413. TIMER_PRDHI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRD																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3465. TIMER_PRDHI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PRD	R/W	0h	Value = 0000 0000h-FFFF FFFFh Period register. This register contains the full timer period for a 32-bit timer configuration or half the timer period for a 64-bit timer configuration.

Table 11-3466. Register Call Summary for TIMER_PRDHI

Timers Registers • TIMER_PRDHI Register (Offset = 1Ch) [reset = 0h]: [0] • TIMER_PRDLO Register (Offset = 18h) [reset = 0h]: [0] • Timers Registers: [0][1] • TIMER_TCR Register (Offset = 20h) [reset = 0h]: [0][1] • TIMER_TGCR Register (Offset = 24h) [reset = 0h]: [0][1]
Timers Functional Description • Timer Counter Register Read Reset Mode: [0] • Watchdog Timer Mode Operation: [0][1] • 64-Bit Timer Mode: [0] • Counter, Reload and Period Registers Used in GP Timer Modes: [0][1][2][3] • Watchdog Timer Register Write Protection: [0] • Writing to Registers of an Active Timer: [0] • Chained Mode: [0] • Unchained Mode: [0][1] • Timer Counting: [0]
Timers Programming Guide • Timers Module Global Initialization: [0]

11.17.6.9 TIMER_TCR Register (Offset = 20h) [reset = 0h]

TIMER_TCR is shown in Figure 11-1414 and described in Table 11-3468.

The lower 16 bits of the timer control register (TIMER_TCR) determine the operating mode and monitor the status of TIMLO, as well as control the function of TINPL and TOUTL. The upper 16 bits determine the operating mode and monitor the status of TIMHI. The upper 16 bits of the TIMER_TCR are used only when the timer is configured in dual 32-bit timers unchained mode (TIMMODE = 1h in TIMER_TGCR).

Table 11-3467. TIMER_TCR Instances

Instance	Physical Address
TIMER_0	0220 0020h
TIMER_1	0221 0020h
TIMER_2	0222 0020h
TIMER_3	0223 0020h
TIMER_4	0224 0020h
TIMER_5	0225 0020h
TIMER_6	0226 0020h

Figure 11-1414. TIMER_TCR Register

31	30	29	28	27	26	25	24
RESERVED		CAPEVTMODE_HI		CAPMODE_HI	READRSTMODE_HI	TIEN_HI	CLKSRC_HI
R/W-0h		R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
ENAMODE_HI		PWID_HI		CP_HI	INVINP_HI	INVOUTP_HI	TSTAT_HI
R/W-0h		R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED		CAPEVTMODE_LO		CAPMODE_LO	READRSTMODE_LO	TIEN_LO	CLKSRC_LO
R/W-0h		R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
ENAMODE_LO		PWID_LO		CP_LO	INVINP_LO	INVOUTP_LO	TSTAT_LO
R/W-0h		R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3468. TIMER_TCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved. The reserved bit location is always read as 0.
29-28	CAPEVTMODE_HI	R/W	0h	Capture event mode for TIMHI. Uses these bits to specify the type of event for Capture mode. • 0h = Event occurs on timer input rising edge. • 1h = Event occurs on time input falling edge. • 2h = Event occurs on both rising and falling edges. • 3h = Reserved
27	CAPMODE_HI	R/W	0h	Capture mode enable bit for TIMHI. Determines if external event can reset timer. Capture mode is only available in dual 32-bit unchained mode and when CLKSRC = 0 and ENAMODE = 2h or 3h. Output events (interrupt/DMA/other) are generated when capture mode event occurs. • 0 = Timer is not in capture mode. • 1 = Timer is in capture mode. External event can reset timer.

Table 11-3468. TIMER_TCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	READRSTMODE_HI	R/W	0h	Read reset mode enable bit for TIMHI. Determines the effect of a timer counter read on TIMER_CNTHI. Read reset mode is only available in dual 32-bit unchained. Output events (interrupt/DMA/other) are not generated when read reset occurs. • 0 = There is no effect when timer counter register TIMER_CNTHI is read. • 1 = Timer counter is reset when timer counter register TIMER_CNTHI is read.
25	TIEN_HI	R/W	0h	Timer input enable bit determines if the timer clock is gated by the timer input. Applicable only when CLKSRC_HI = 0. • 0 = Timer clock is not gated by the timer input. • 1 = Timer clock is gated by a high state of the timer input synchronized with the internal clock. Timer starts counting when timer input transitions from low to high. Timer stops counting when timer input transitions from high to low.
24	CLKSRC_HI	R/W	0h	Clock source bit determines the clock source for the timer. • 0 = The clock source is the internal clock. • 1 = The clock source is the signal on the timer pin.
23-22	ENAMODE_HI	R/W	0h	A value written to this Enabling mode bits determine the timer mode for TIMHI. • 0h = The timer is disabled (not counting) and maintains the current value. • 1h = The timer is enabled one time. The timer stops after the timer counter reaches the timer period. • 2h = The timer is enabled continuously. The timer counter increments until it reaches the timer period. One timer clock cycle later, the timer counter is reset to 0 and continues counting. • 3h = The timer is enabled continuously. The timer counter increments until it reaches the timer period. One timer clock cycle later, the timer counter is reset to 0, the period registers (TIMER_PRDHI and TIMER_PRDLO) reload with the reload registers (TIMER_RELHI and TIMER_RELLO) and continues counting when TIMMODE is either set to 0h (64-bit timer mode) or 1h (32-bit timers unchained mode).
21-20	PWID_HI	R/W	0h	Pulse width bits for TIMHI. PWID_HI is only used in pulse mode (CP_HI = 0). PWID_HI controls the width of the timer output signal. The polarity of the pulse is controlled by the INVOUTP_HI bit. The timer output signal is recorded in the TSTAT_HI bit and can be made visible on the timer output pin. • 0h = The pulse width is 1 timer clock cycle. • 1h = The pulse width is 2 timer clock cycles. • 2h = The pulse width is 3 timer clock cycles. • 3h = The pulse width is 4 timer clock cycles.
19	CP_HI	R/W	0h	Clock/pulse mode bit for TIMHI. In the watchdog timer mode (TIMMODE = 2h), the pulse mode is selected automatically and the CP_HI bit is a don't care. • 0 = Pulse mode. When the timer counter reaches the timer period, the timer output appears as a pulse with the width defined by the PWID_HI bits and the polarity defined by the INVOUTP_HI bits. • 1 = Clock mode. The timer output signal has a 50% duty cycle signal. When the timer counter reaches the timer period, the level of the timer output signal is toggled (from high to low or from low to high).
18	INVINP_HI	R/W	0h	Timer input inverter control bit for TIMHI. Only affects operation if CLKSRC_HI = 1. • 0 = A non-inverted timer input drives the timer. • 1 = An inverted timer input drives the timer.
17	INVOUTP_HI	R/W	0h	Timer output inverter control bit for TIMHI. • 0 = The timer output is not inverted. • 1 = The timer output is inverted.

Table 11-3468. TIMER_TCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	TSTAT_HI	R/W	0h	Timer status bit for TIMHI. This is a read-only bit that shows the value of the timer output. • 0 = Timer output is low. • 1 = Timer output is high.
15-14	RESERVED	R	0h	The reserved bit location is always read as 0. A value written to this field has no effect.
13-12	CAPEVTMODE_LO	R/W	0h	Capture event mode. Uses these bits to specify the type of event for Capture mode. • 0h = Event occurs on timer input rising edge. • 1h = Event occurs on time input falling edge. • 2h = Event occurs on both rising and falling edges. • 3h = Reserved
11	CAPMODE_LO	R/W	0h	Capture mode enable bit. Determines if external event can reset timer. Capture mode is only available in dual 32-bit unchained mode and when CLKSRC = 0 and ENAMODE = 2h or 3h. Output events (interrupt/DMA/other) are generated when capture mode event occurs. • 0 = Timer is not in capture mode. • 1 = Timer is in capture mode. External event can reset timer.
10	READRSTMODE_LO	R/W	0h	Read reset mode enable bit. Determines the effect of a timer counter read on TIMER_CNTLO. Read reset mode is only available in dual 32-bit unchained. Output events (interrupt/DMA/other) are not generated when read reset occurs. • 0 = There is no effect when timer counter register TIMER_CNTLO is read. • 1 = Timer counter is reset when timer counter register TIMER_CNTLO is read.
9	TIEN_LO	R/W	0h	Timer input enable bit determines if the timer clock is gated by the timer input. Applicable only when CLKSRC_LO = 0. • 0 = Timer clock is not gated by the timer input. • 1 = Timer clock is gated by a high state of the timer input synchronized with the internal clock. Timer starts counting when timer input transitions from low to high. Timer stops counting when timer input transitions from high to low.
8	CLKSRC_LO	R/W	0h	Clock source bit determines the clock source for the timer. • 0 = The clock source is the internal clock. • 1 = The clock source is the signal on the timer pin.
7-6	ENAMODE_LO	R/W	0h	Enabling mode bits determine the timer mode. • 0h = The timer is disabled (not counting) and maintains the current value. • 1h = The timer is enabled one time. The timer stops after the timer counter reaches the timer period. • 2h = The timer is enabled continuously. The timer counter increments until it reaches the timer period. One timer clock cycle later, the timer counter is reset to 0 and continues counting. • 3h = The timer is enabled continuously. The timer counter increments until it reaches the timer period. One timer clock cycle later, the timer counter is reset to 0, the period registers (TIMER_PRDHI and TIMER_PRDLO) reload with the reload registers (TIMER_RELHI and TIMER_RELLO) and continues counting when TIMMODE is either set to 0h (64-bit timer mode) or 1h (32-bit timers unchained mode).

Table 11-3468. TIMER_TCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	PWID_LO	R/W	0h	Pulse width bits. PWID_LO is only used in pulse mode (CP_LO = 0). PWID_LO controls the width of the timer output signal. The polarity of the pulse is controlled by the INVOUTP_LO bit. The timer output signal is recorded in the TSTAT_LO bit and can be made visible on the timer output pin. • 0h = The pulse width is 1 timer clock cycle. • 1h = The pulse width is 2 timer clock cycles. • 2h = The pulse width is 3 timer clock cycles. • 3h = The pulse width is 4 timer clock cycles.
3	CP_LO	R/W	0h	Clock/pulse mode bit for timer output. In the watchdog timer mode (TIMMODE = 2h), the pulse mode is selected automatically and the CP_LO bit is a don't care. • 0 = Pulse mode. When the timer counter reaches the timer period, the timer output appears as a pulse with the width defined by the PWID_LO bits and the polarity defined by the INVOUTP_LO bits. • 1 = Clock mode. The timer output signal has a 50% duty cycle signal. When the timer counter reaches the timer period, the level of the timer output signal is toggled (from high to low or from low to high).
2	INVINP_LO	R/W	0h	Timer input inverter control bit. Only affects operation if CLKSRC_LO = 1. • 0 = A non-inverted timer input drives the timer. • 1 = An inverted timer input drives the timer.
1	INVOUTP_LO	R/W	0h	Timer output inverter control bit. • 0 = The timer output is not inverted. • 1 = The timer output is inverted.
0	TSTAT_LO	R/W	0h	Timer status bit. This is a read-only bit that shows the value of the timer output. TSTAT_LO drives the timer pin (TOUTL) when the pin is used as a timer output pin and may be inverted by setting INVOUTP_LO = 1. • 0 = Timer output is low. • 1 = Timer output is high.

Table 11-3469. Register Call Summary for TIMER_TCR

Timers Registers • TIMER_TCR Register (Offset = 20h) [reset = 0h]: [0][1][2] • Timers Registers: [0][1] • TIMER_TGCR Register (Offset = 24h) [reset = 0h]: [0][1]
Timers Functional Description • Event Capture Mode: [0][1] • Timer Counter Register Read Reset Mode: [0] • Timer Reset Sources: [0][1] • Timer Count = 0 and Timer Period = 0 (No Prescaler): [0][1][2] • Timer Counting: [0][1] • Timer Output Mode Selection: [0][1][2][3][4][5][6][7][8][9][10][11] • Watchdog Timer Register Write Protection: [0] • Timer Output Signal and Timer Interrupt Signal in Watchdog Timer Mode: [0] • Writing to Registers of an Active Timer: [0] • Timer Clock Source Selection: [0][1][2][3][4][5][6] • Timer Enabling: [0][1][2][3] • 64-Bit Timer Mode: [0][1] • Watchdog Timer Mode Operation: [0][1][2][3] • Chained Mode: [0][1] • Unchained Mode: [0][1][2][3][4][5][6][7][8][9][10][11]
Timers Programming Guide • Timers Module Global Initialization: [0][1][2]

11.17.6.10 TIMER_TGCR Register (Offset = 24h) [reset = 0h]

TIMER_TGCR is shown in Figure 11-1415 and described in Table 11-3471. The timer global control register (TIMER_TGCR) contains a field for selecting the operating mode of the timer (TIMMODE), timer reset bits (TIMHIRS and TIMLORS), and counters for TIMHI in the dual 32-bit timers unchained mode (TDDRHI and PSCHI).

Table 11-3470. TIMER_TGCR Instances

Instance	Physical Address
TIMER_0	0220 0024h
TIMER_1	0221 0024h
TIMER_2	0222 0024h
TIMER_3	0223 0024h
TIMER_4	0224 0024h
TIMER_5	0225 0024h
TIMER_6	0226 0024h

Figure 11-1415. TIMER_TGCR Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
TDDRHI				PSCHI			
R/W-0h				R/W-0h			
7	6	5	4	3	2	1	0
RESERVED			BW_COMPATIBLE	TIMMODE		TIMHIRS	TIMLORS
R/W-0h			R/W-0h	R/W-0h		R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3471. TIMER_TGCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Reserved. The reserved bit location is always read as 0.
15-12	TDDRHI	R/W	0h	Timer divide-down ratio bits. This field is the prescale counter for TIMHI in the dual 32-bit timers unchained mode (TIMMODE = 1h). When the timer is enabled, TDDRHI increments every timer clock cycle. The timer counter (TIMER_CNTHI) increments on the cycle after the TDDRHI matches the value of PSCHI. TDDRHI resets to 0 and continues. If enabled one time, when TIMER_CNTHI matches the TIMER_PRDHI, the timer stops; if the timer is enabled continuously, TIMER_CNTHI resets to 0 on the cycle after matching the TIMER_PRDHI and the timer continues counting. The default value is 0000h.
11-8	PSCHI	R/W	0h	Prescale period bits. This field specifies the prescale period for TIMHI in the dual 32-bit timers unchained mode (TIMMODE = 1h). The default value is 0000h.
7-5	RESERVED	R/W	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

Table 11-3471. TIMER_TGCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	PLUSEN	R/W	0h	Enable Timer Plus features. • 0 = Enable backward compatibility. Disable Timer Plus features (Timer Plus features are unavailable). • 1 = Disable backward compatibility. Enable Timer Plus features (Timer Plus features are available).
3-2	TIMMODE	R/W	0h	Timer mode bits determine the timer operating mode. • 0h = The timer is in the 64-bit general-purpose timer mode. • 1h = The timer is in the dual 32-bit timers unchained mode. • 2h = The timer is in the 64-bit watchdog timer mode. • 3h = The timer is in the dual 32-bit timers chained mode.
1	TIMHIRS	R/W	0h	TIMHI reset bit. Note that in order for the timer to function properly in 64-bit general-purpose timer mode both the TIMHIRS and TIMLORS bits must be set to 1. If the timer is in the watchdog timer active state, changing this bit does not affect the timer. • 0 = TIMHI is in reset. The TSTAT_HI bit of TIMER_TCR is reset to 0. However, the counter register TIMER_CNTHI keeps its current value. • 1 = TIMHI is not in reset. TIMHI can be used as a 32-bit timer.
0	TIMLORS	R/W	0h	TIMLO reset bit. Note that in order for the timer to function properly in 64-bit general-purpose timer mode both the TIMHIRS and TIMLORS bits must be set to 1. If the timer is in the watchdog timer active state, changing this bit does not affect the timer. • 0 = TIMLO is in reset. The TSTAT_LO bit of TIMER_TCR is reset to 0, and the timer output signal is in the high-impedance state. However, the counter register TIMER_CNTLO keeps its current value. • 1 = TIMLO is not in reset. TIMLO can be used as a 32-bit timer.

Table 11-3472. Register Call Summary for TIMER_TGCR

Timers Registers • TIMER_TCR Register (Offset = 20h) [reset = 0h]: [0] • Timers Registers: [0][1] • TIMER_TGCR Register (Offset = 24h) [reset = 0h]: [0][1]
Timers Functional Description • Timer Mode Selection: [0] • Event Capture Mode: [0] • Timer Counter Register Read Reset Mode: [0] • Timer Plus/Additional Modes: [0] • Watchdog Timer Mode Restrictions: [0] • Watchdog Timer Mode: [0] • Writing to Registers of an Active Timer: [0][1][2] • Timer Enabling: [0][1][2][3][4] • Watchdog Timer Register Write Protection: [0][1] • 64-Bit Timer Mode: [0] • Dual 32-bit Timer Mode: [0] • Timer Reset Sources: [0][1][2][3][4] • Unchained Mode: [0][1][2]
Timers Overview • Timers Features: [0]
Timers Programming Guide • Timers Module Global Initialization: [0][1]

11.17.6.11 TIMER_WDTCR Register (Offset = 28h) [reset = 0h]

TIMER_WDTCR is shown in Figure 11-1416 and described in Table 11-3474. This register determines the state of the watchdog timer and monitors the watchdog timer status.

Table 11-3473. TIMER_WDTCR Instances

Instance	Physical Address
TIMER_0	0220 0028h
TIMER_1	0221 0028h
TIMER_2	0222 0028h
TIMER_3	0223 0028h
TIMER_4	0224 0028h
TIMER_5	0225 0028h
TIMER_6	0226 0028h

Figure 11-1416. TIMER_WDTCR Register

31	30	29	28	27	26	25	24
WDKEY							
R/W-0h							
23	22	21	20	19	18	17	16
WDKEY							
R/W-0h							
15	14	13	12	11	10	9	8
WDFLAG	WDEN	WDIKEY		RESERVED			
R/W-0h	R/W-0h	R/W-0h		R/W-0h			
7	6	5	4	3	2	1	0
RESERVED							
R/W-0h							

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3474. TIMER_WDTCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	WDKEY	R/W	0h	Value = 0000h-FFFFh Watchdog timer service key bits. Before a watchdog timeout event occurs, only a write sequence of a A5C6h followed by a DA7Eh services the watchdog timer. Any other write triggers a watchdog timeout event immediately. The default value of WDKEY is 0000h. WDKEY is not applicable in the general-purpose timer mode.
15	WDFLAG	R/W	0h	Watchdog timer flag bit. WDFLAG is cleared when the watchdog timer is moved to the active state, when a hardware reset occurs, and when 1 is written to WDFLAG. • 0 = No watchdog timer timeout event occurred. • 1 = Watchdog timer timeout event occurred.
14	WDEN	R/W	0h	Watchdog timer enable bit. WDEN must be set to move the watchdog timer to the pre-active state (see Figure 11-1404). • 0 = Watchdog timer is disabled. Watchdog timer output pin is disconnected from the watchdog timer timeout event, and the timer counter starts to count. The timer is in the general-purpose timer mode. • 1 = Watchdog timer is enabled. The watchdog timer output pin is connected to the watchdog timeout event. The watchdog timer can be disabled by a watchdog timeout event or by a hardware reset.

Table 11-3474. TIMER_WDTCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13-12	WDIKEY	R/W	0h	Value = 0h-3h Watchdog idle enable key bits. A write sequence of 1h followed by 2h is required before the watchdog timer can enter the idle mode. Writing 0h to WDIKEY prevents the watchdog timer from entering the idle mode. WDIKEY is not applicable in the general-purpose timer mode.
11-0	RESERVED	R/W	0h	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

Table 11-3475. Register Call Summary for TIMER_WDTCR

Timers Registers • TIMER_WDTCR Register (Offset = 28h) [reset = 0h]: [0] • Timers Registers: [0][1]
Timers Functional Description • Watchdog Timer Mode: [0] • Watchdog Timer Register Write Protection: [0][1] • Watchdog Timer Mode Operation: [0][1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16] • Watchdog Timer Mode Restrictions: [0]
Timers Programming Guide • Timers Module Global Initialization: [0][1][2]

11.17.6.12 TIMER_RELLO Register (Offset = 34h) [reset = 0h]

TIMER_RELLO is shown in Figure 11-1417 and described in Table 11-3477. The timer reload registers (TIMER_RELLO and TIMER_RELHI) are 32-bit-wide registers which can be used in conjunction to form a single 64-bit reload register or separately as 32-bit reload registers. These two registers are used in conjunction with the two counter-registers, TIMER_CNTHI and TIMER_CNTLO.

Table 11-3476. TIMER_RELLO Instances

Instance	Physical Address
TIMER_0	0220 0034h
TIMER_1	0221 0034h
TIMER_2	0222 0034h
TIMER_3	0223 0034h
TIMER_4	0224 0034h
TIMER_5	0225 0034h
TIMER_6	0226 0034h

Figure 11-1417. TIMER_RELLO Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REL																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3477. TIMER_RELLO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REL	R/W	0h	Value = 0000 0000h-FFFF FFFFh Reload register. This register contains the full timer reload for a 32-bit timer configuration or half the timer reload for a 64-bit timer configuration.

Table 11-3478. Register Call Summary for TIMER_RELLO

Timers Registers • TIMER_RELLO Register (Offset = 34h) [reset = 0h]: [0][1] • TIMER_TCR Register (Offset = 20h) [reset = 0h]: [0][1] • Timers Registers: [0][1]
Timers Functional Description • Unchained Mode: [0] • 64-Bit Timer Mode: [0] • Counter, Reload and Period Registers Used in GP Timer Modes: [0][1][2] • Timer Counting: [0]
Timers Programming Guide • Timers Module Global Initialization: [0]

11.17.6.13 TIMER_RELHI Register (Offset = 38h) [reset = 0h]

TIMER_RELHI is shown in Figure 11-1418 and described in Table 11-3480.

Table 11-3479. TIMER_RELHI Instances

Instance	Physical Address
TIMER_0	0220 0038h
TIMER_1	0221 0038h
TIMER_2	0222 0038h
TIMER_3	0223 0038h
TIMER_4	0224 0038h
TIMER_5	0225 0038h
TIMER_6	0226 0038h

Figure 11-1418. TIMER_RELHI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REL																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3480. TIMER_RELHI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REL	R/W	0h	Value = 0000 0000h-FFFF FFFFh Reload register. This register contains the full timer reload for a 32-bit timer configuration or half the timer reload for a 64-bit timer configuration.

Table 11-3481. Register Call Summary for TIMER_RELHI

Timers Registers • TIMER_RELLO Register (Offset = 34h) [reset = 0h]: [0] • TIMER_TCR Register (Offset = 20h) [reset = 0h]: [0][1] • TIMER_RELHI Register (Offset = 38h) [reset = 0h]: [0] • Timers Registers: [0][1]
Timers Functional Description • Unchained Mode: [0] • 64-Bit Timer Mode: [0] • Counter, Reload and Period Registers Used in GP Timer Modes: [0][1][2] • Timer Counting: [0]
Timers Programming Guide • Timers Module Global Initialization: [0]

11.17.6.14 TIMER_CAPLO Register (Offset = 3Ch) [reset = 0h]

TIMER_CAPLO is shown in Figure 11-1419 and described in Table 11-3483. The timer capture registers (TIMER_CAPLO and TIMER_CAPHI) are 32-bit-wide registers which can be used to store the timer counter bits. When the timer has a timeout (due to external event or read of counter register) the value of TIMER_CNTHI and TIMER_CNTLO registers are copied onto the TIMER_CAPHI and TIMER_CAPLO registers. These two registers are used in conjunction with the two counter-registers, TIMER_CNTHI and TIMER_CNTLO.

Table 11-3482. TIMER_CAPLO Instances

Instance	Physical Address
TIMER_0	0220 003Ch
TIMER_1	0221 003Ch
TIMER_2	0222 003Ch
TIMER_3	0223 003Ch
TIMER_4	0224 003Ch
TIMER_5	0225 003Ch
TIMER_6	0226 003Ch

Figure 11-1419. TIMER_CAPLO Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3483. TIMER_CAPLO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP	R/W	0h	Value = 0000 0000h-FFFF FFFFh Capture register. Captured timer counter bits.

Table 11-3484. Register Call Summary for TIMER_CAPLO

Timers Registers • TIMER_CAPLO Register (Offset = 3Ch) [reset = 0h]: [0][1][2] • Timers Registers: [0][1]
Timers Functional Description • Event Capture Mode: [0] • Timer Counter Register Read Reset Mode: [0] • Timer Capture Registers: [0]

11.17.6.15 TIMER_CAPHI Register (Offset = 40h) [reset = 0h]

TIMER_CAPHI is shown in Figure 11-1420 and described in Table 11-3486.

Table 11-3485. TIMER_CAPHI Instances

Instance	Physical Address
TIMER_0	0220 0040h
TIMER_1	0221 0040h
TIMER_2	0222 0040h
TIMER_3	0223 0040h
TIMER_4	0224 0040h
TIMER_5	0225 0040h
TIMER_6	0226 0040h

Figure 11-1420. TIMER_CAPHI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP																															
R/W-0h																															

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3486. TIMER_CAPHI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP	R/W	0h	Value = 0000 0000h-FFFF FFFFh Capture register. Captured timer counter bits.

Table 11-3487. Register Call Summary for TIMER_CAPHI

Timers Registers • TIMER_CAPLO Register (Offset = 3Ch) [reset = 0h]: [0][1] • Timers Registers: [0][1] • TIMER_CAPHI Register (Offset = 40h) [reset = 0h]: [0]
Timers Functional Description • Timer Counter Register Read Reset Mode: [0] • Timer Capture Registers: [0]

11.17.6.16 TIMER_INTCTL_STAT Register (Offset = 44h) [reset = 0h]

TIMER_INTCTL_STAT is shown in Figure 11-1421 and described in Table 11-3489.

Table 11-3488. TIMER_INTCTL_STAT Instances

Instance	Physical Address
TIMER_0	0220 0044h
TIMER_1	0221 0044h
TIMER_2	0222 0044h
TIMER_3	0223 0044h
TIMER_4	0224 0044h
TIMER_5	0225 0044h
TIMER_6	0226 0044h

Figure 11-1421. TIMER_INTCTL_STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED				EVT_INT_STAT_HI	EVT_INT_EN_HI	CMP_INT_STAT_HI	CMP_INT_EN_HI
R/W-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED				EVT_INT_STAT_LO	EVT_INT_EN_LO	CMP_INT_STAT_LO	CMP_INT_EN_LO
R/W-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; *-n* = value after reset

Table 11-3489. TIMER_INTCTL_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R/W	0h	Reserved. The reserved bit location is always read as 0.
19	EVTINTSTAT_HI	R/W	0h	Interrupt status which reflects the condition that an external event caused a timeout when timer is in capture mode. Write a 1 to clear this bit. • 0 = Interrupt has not occurred. • 1 = Interrupt has occurred.
18	EVTINTEN_HI	R/W	0h	Enables the interrupt generation when timer is in capture mode. • 0 = Disable interrupt when in event capture mode. • 1 = Enable interrupt when in event capture mode.
17	PRDINTSTAT_HI	R/W	0h	Interrupt status which reflects the condition that timer counter matched the period register when timer is enabled. Write a 1 to clear this bit. • 0 = Interrupt has not occurred. • 1 = Interrupt has occurred.
16	PRDINTEN_HI	R/W	0h	Enable interrupt generation when timer is enabled in 64-bit/32-bit chained/unchained/watchdog modes. • 0 = Disable interrupt. • 1 = Enable interrupt.
15-4	RESERVED	R/W	0h	Reserved. The reserved bit location is always read as 0.

Table 11-3489. TIMER_INTCTL_STAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	EVTINTSTAT_LO	R/W	0h	Interrupt status which reflects the condition that an external event caused a timeout when timer is in capture mode. Write a 1 to clear this bit. • 0 = Interrupt has not occurred. • 1 = Interrupt has occurred.
2	EVTINTEN_LO	R/W	0h	Enables the interrupt generation when timer is in capture mode. • 0 = Disable interrupt when in event capture mode. • 1 = Enable interrupt when in event capture mode.
1	PRDINTSTAT_LO	R/W	0h	Interrupt status which reflects the condition that timer counter matched the period register when timer is enabled. Write a 1 to clear this bit. • 0 = Interrupt has not occurred. • 1 = Interrupt has occurred.
0	PRDINTEN_LO	R/W	0h	Enable interrupt generation when timer is enabled in 64-bit/32-bit chained/unchained/watchdog modes. • 0 = Disable interrupt. • 1 = Enable interrupt.

Table 11-3490. Register Call Summary for TIMER_INTCTL_STAT

Timers Registers
- Timers Registers: [0][1]
- TIMER_INTCTL_STAT Register (Offset = 44h) [reset = 0h]: [0]

11.18 Universal Asynchronous Receiver/Transmitter (UART)

This chapter describes the function, operation, and configuration of the Universal Asynchronous Receiver/Transmitter (UART) module in the device.

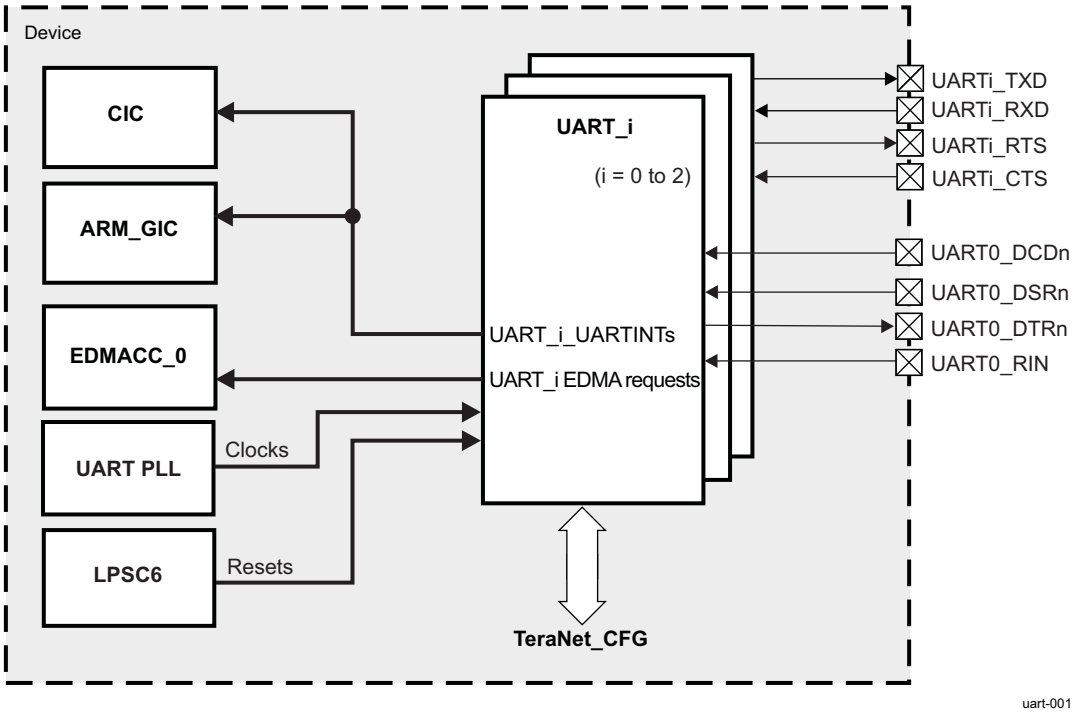
11.18.1 UART Overview

The Universal Asynchronous Receiver/Transmitter peripheral is 16550 standard compatible asynchronous communications element. The UART can be placed in an alternate FIFO mode. This relieves the CPU of excessive software overhead by buffering received and transmitted characters. The receiver and transmitter FIFOs store up to 16 bytes including three additional bits of error status per byte for the receiver FIFO.

There are 3 UART (UART_0, UART_1 and UART_2) modules in the device. Only UART_0 supports full modem control functions. Each UART can be used for configuration and data exchange with a number of external peripheral devices or interprocessor communication between devices.

Figure 11-1422 shows the UART module overview.

Figure 11-1422. UART Overview



11.18.1.1 UART Features

The UART_i (where i = 0 to 2) include the following features:

- 16550 standard compatible
- 16-byte FIFO buffer for receiver and 16-byte FIFO for transmitter
- Baud generation based on programmable divisors operating from a fixed functional clock of 192 MHz
- Oversampling is programmed by software as 16 or 13. Thus, the baud rate computation is one of two options:
  - Baud rate = (functional clock / 16) / N
  - Baud rate = (functional clock / 13) / N
- Break character detection and generation
- Configurable data format:
  - Data bit: 5, 6, 7, or 8 bits

  - Parity bit: Even, odd, none
  - Stop-bit: 1, 1.5, 2 bit(s)
- Flow control: Hardware (RTS/CTS)
- The 192 MHz functional clock option allows baud rates up to 12Mbps

The UART performs serial-to-parallel conversions on data received from a peripheral device or modem and parallel-to-serial conversion on data received from the CPU. The CPU can read the UART status at any time. The UART includes control capability and a processor interrupt system that can be tailored to minimize software management of the communications link.

11.18.2 UART Environment

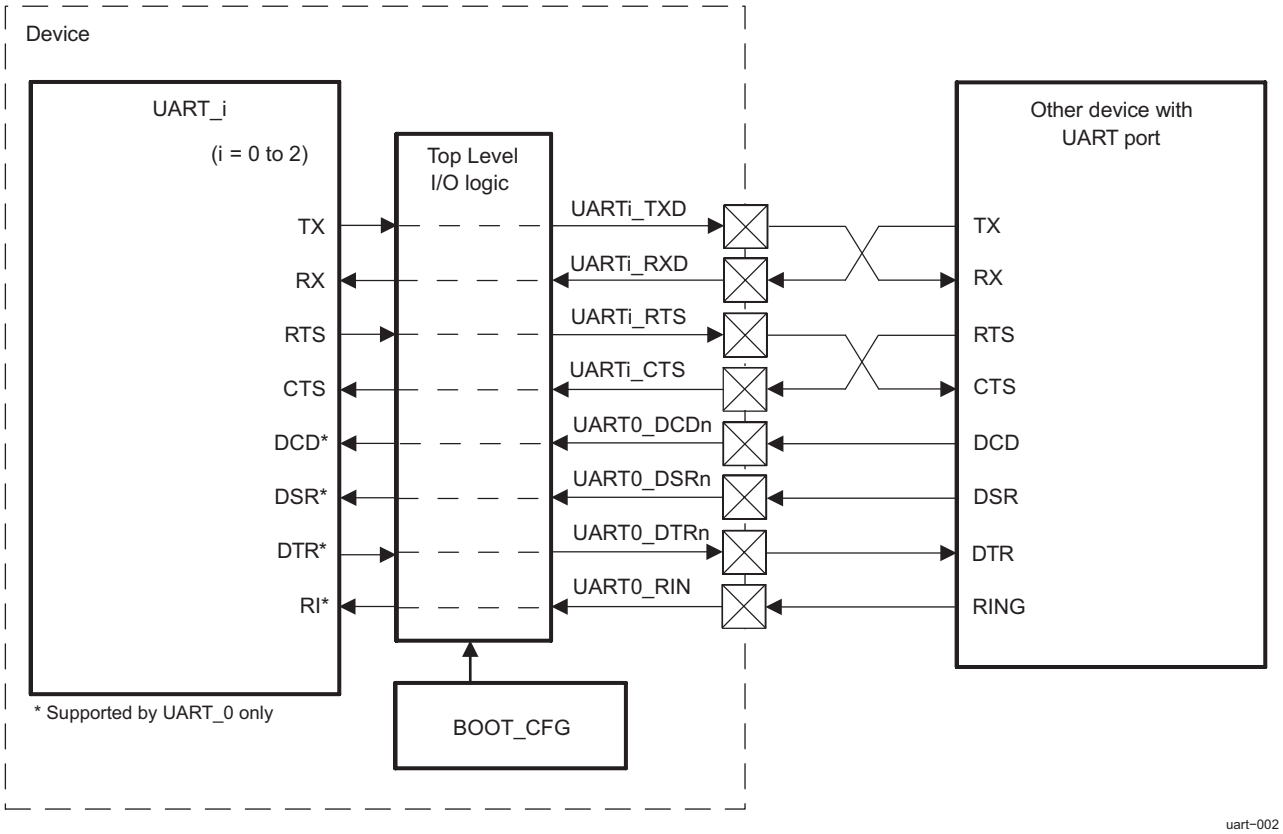
This section describes the UART connection with an external device.

11.18.2.1 UART Interface

11.18.2.1.1 System Using UART Communication With Hardware Handshake

Each UART instance can be easily connected to the UART port of an external IC (see Figure 11-1423).

Figure 11-1423. UART Mode Bus System Overview



11.18.2.1.1.1 UART Interface Description

Table 11-3491 lists the UART interface input/output (I/O) signals.

Table 11-3491. UART Interface Signals

Device Signal	Module Signal	I/O[1]	Description	Module Level Reset Value
UART_0 Interface Signals				
UART0_RXD	RX	I	UART_0 Serial data input	HiZ
UART0_TXD	TX	O	UART_0 Serial data output This pin is set to high on reset.	1
UART0_CTSn	CTS	I	UART_0 Clear to send. Active low	HiZ
UART0_RTSn	RTS	O	UART_0 Request to send. Active low	1
UART_0 Modem Signals				
UART0_DCDn	DCD	I	UART_0 Data Carrier Detect. Active low	HiZ
UART0_DSRn	DSR	I	UART_0 Data Set Ready. Active low	HiZ

(1) I = Input; O = Output; I/O = Bidirectional

Table 11-3491. UART Interface Signals (continued)

Device Signal	Module Signal	I/O[1]	Description	Module Level Reset Value
UART0_DTRn	DTR	O	UART_0 Data Terminal Ready. Active low	1
UART0_RIN	RI	I	UART_0 Ring Indicator. Active low	HiZ
UART_1 Interface Signals				
UART1_RXD	RX	I	UART_1 Serial data input	HiZ
UART1_TXD	TX	O	UART_1 Serial data output This pin is set to high on reset.	1
UART1_CTSn	CTS	I	UART_1 Clear to send. Active low	HiZ
UART1_RTSn	RTS	O	UART_1 Request to send. Active low	1
UART_2 Interface Signals				
UART2_RXD	RX	I	UART_2 Serial data input	HiZ
UART2_TXD	TX	O	UART_2 Serial data output This pin is set to high on reset.	1
UART2_CTSn	CTS	I	UART_2 Clear to send. Active low	HiZ
UART2_RTSn	RTS	O	UART_2 Request to send. Active low	1

NOTE: The path from a module pin to device pad(s) is defined at the device I/O logic level. The I/O logic maps the module signals to the different pads of the device and is programmable in the BOOT_CFG module registers. For more information on control module settings, see Section 5.1.3.1.1, *Pad Configuration Registers*, in Section 5.1, *BOOT_CFG*.

11.18.2.1.1.2 UART Protocol and Data Format

The UART device operates in two modes:

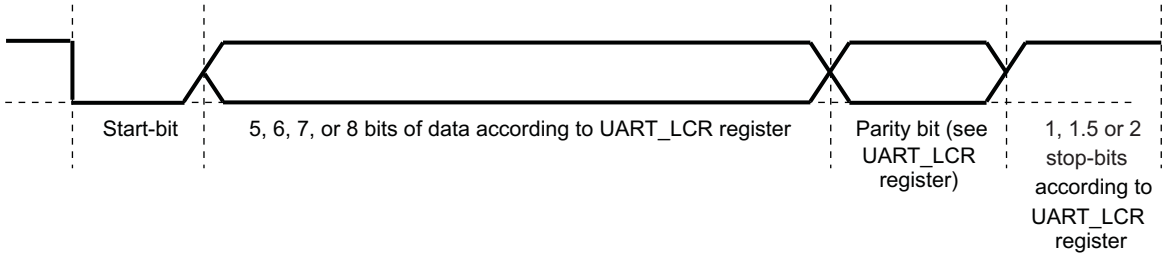
- UART 16×
- UART 13×

The UART uses a wired interface for serial communication with a remote device.

The UART is functionally compatible with the 16550 UART

Figure 11-1424 shows the UART frame data format.

Figure 11-1424. UART Frame Data Format



11.18.3 UART Integration

This section describes module integration in the device, including information about clocks, resets, and hardware requests.

Figure 11-1425 shows the device internal connections with related modules for UART functions.

Figure 11-1425. UART Integration

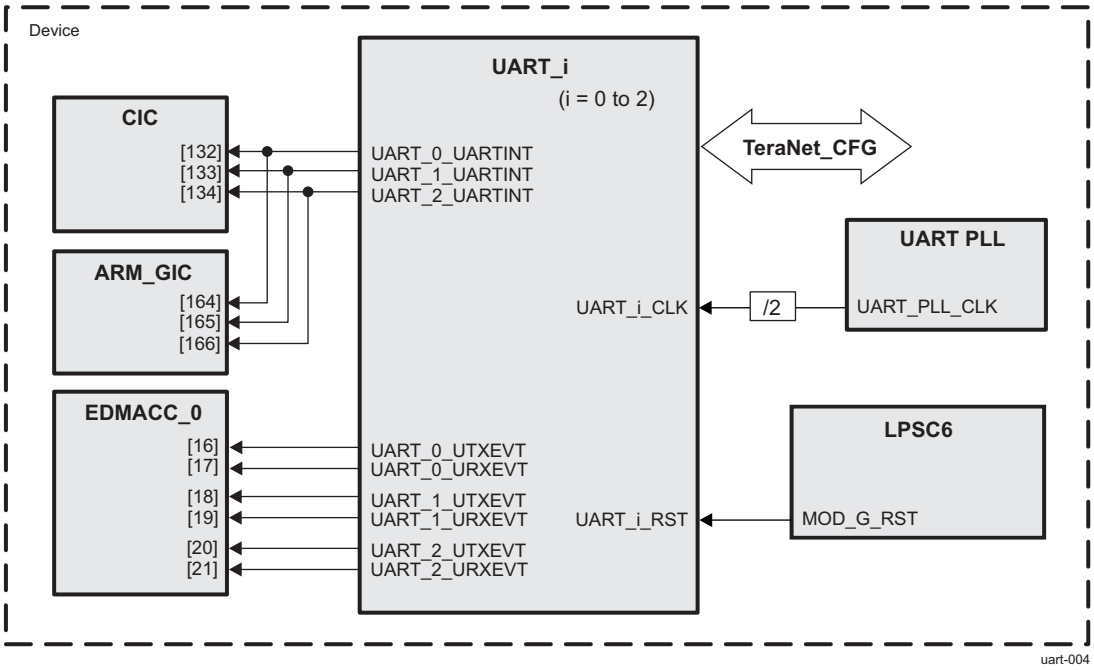


Table 11-3492 through Table 11-3494 summarize the integration of the module in the device.

Table 11-3492. UART Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
UART_0	PD5	LPSC6	TeraNet_CFG
UART_1	PD5	LPSC6	TeraNet_CFG
UART_2	PD5	LPSC6	TeraNet_CFG

Table 11-3493. UART Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
UART_0	UART_0_CLK	UART_PLL_CLK/2	UART PLL	UART_0 Interface and Functional clock
UART_1	UART_1_CLK	UART_PLL_CLK/2	UART PLL	UART_1 Interface and Functional clock
UART_2	UART_2_CLK	UART_PLL_CLK/2	UART PLL	UART_2 Interface and Functional clock
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
UART_0	UART_0_RST	MOD_G_RST	LPSC6	UART_0 Reset signal
UART_1	UART_1_RST	MOD_G_RST	LPSC6	UART_1 Reset signal
UART_2	UART_2_RST	MOD_G_RST	LPSC6	UART_2 Reset signal

Table 11-3494. UART Hardware Requests

Interrupt Requests				
Module Instance	**Event Name**	**Mapped To Input Event [Number]**		**Description**
		ARM_GIC	**CIC**	
UART_0	UART_0_UARTINT	[164]	[132]	UART_0 Interrupt event line
UART_1	UART_1_UARTINT	[165]	[133]	UART_1 Interrupt event line
UART_2	UART_2_UARTINT	[166]	[134]	UART_2 Interrupt event line
DMA Requests				
Module Instance	**Event Name**	**Mapped To Input Event [Number]**		**Description**
		EDMACC_0		
UART_0	UART_0_UTXEVT	[16]		UART_0 transmit request line
	UART_0_URXEVT	[17]		UART_0 receive request line
UART_1	UART_1_UTXEVT	[18]		UART_1 transmit request line
	UART_1_URXEVT	[19]		UART_1 receive request line
UART_2	UART_2_UTXEVT	[20]		UART_2 transmit request line
	UART_2_URXEVT	[21]		UART_2 receive request line

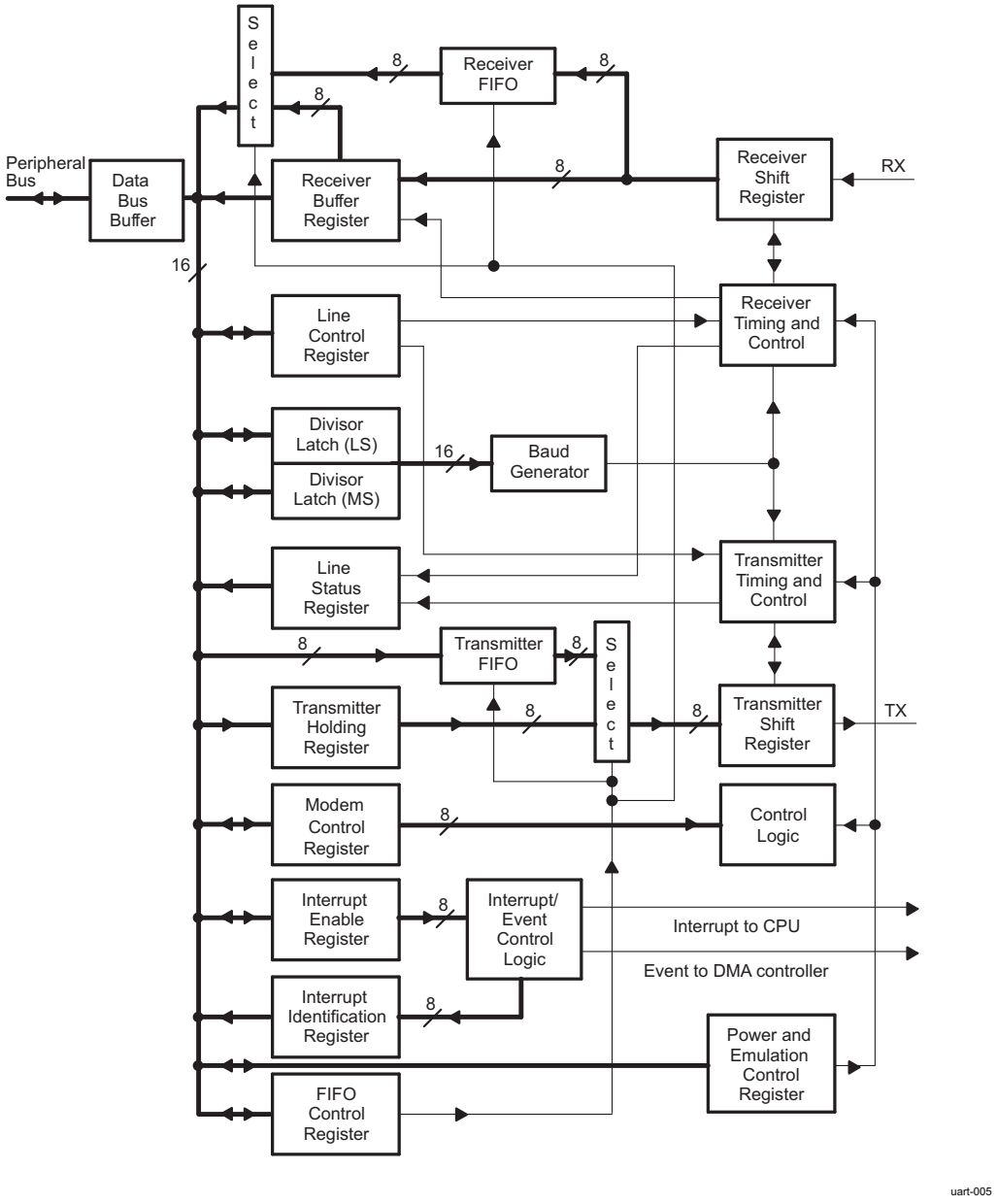
NOTE: UART interrupts are described in Section 11.18.4.3, *Interrupt Requests*.

11.18.4 UART Functional Description

11.18.4.1 Block Diagram

Figure 11-1426 is the UART block diagram.

Figure 11-1426. UART Functional Block Diagram

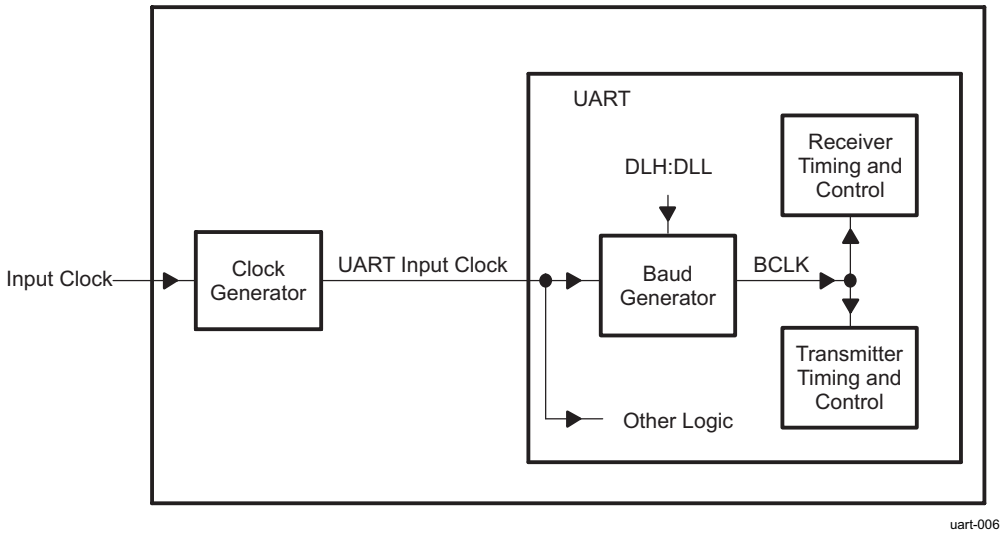


11.18.4.2 Clock Configuration

The UART bit clock is derived from an input clock to the UART.

Figure 11-1427 is a conceptual clock generation diagram for the UART. The processor clock generator receives a signal from an external clock source and produces a UART input clock with a programmed frequency. The UART contains a programmable baud generator that takes an input clock and divides it by a divisor in the range between 1 and (2^{16} - 1) to produce a baud clock (BCLK). The frequency of BCLK is sixteen times (16×) the baud rate (each received or transmitted bit lasts 16 BCLK cycles) or thirteen times (13×) the baud rate (each received or transmitted bit lasts 13 BCLK cycles). When the UART is receiving, the bit is sampled in the eighth BCLK cycle for 16× over sampling mode and on the sixth BCLK cycle for 13× oversampling mode.

Figure 11-1427. UART Clock Generation Diagram



The 16× or 13× reference clock is selected by configuring the UART_MDR[0] OSM_SEL bit. The formula to calculate the divisor is:

$$\text{Divisor} = \frac{\text{UART Input Clock Frequency}}{\text{Desired Baud Rate} \times 16} \quad \text{[MDR.OSM\_SEL = 0]}$$

$$\text{Divisor} = \frac{\text{UART Input Clock Frequency}}{\text{Desired Baud Rate} \times 13} \quad \text{[MDR.OSM\_SEL = 1]}$$

uart-007

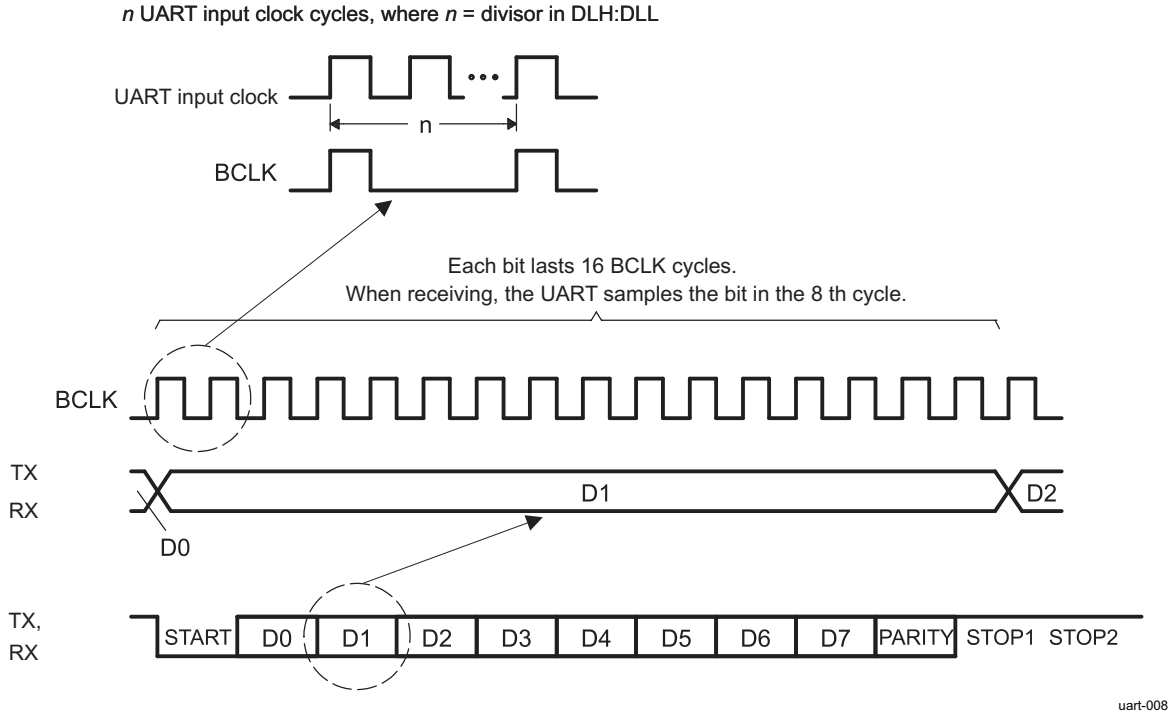
Two 8-bit register fields:

- Divisor MSB Latch [7-0] DLH
- Divisor LSB Latch [7-0] DLL,

called divisor latches, hold this 16-bit divisor. DLH holds the most significant bits of the divisor, and DLL holds the least significant bits of the divisor. For information about these register fields, see Section 11.18.6.11 and Section 11.18.6.12. These divisor latches must be loaded during initialization of the UART to ensure desired operation of the baud generator. Writing to the divisor latches results in two wait states being inserted during the write access while the baud generator is loaded with the new value.

Figure 11-1428 summarizes the relationship between the transferred data bit, BCLK, and the UART input clock. Note that the timing relationship in Figure 11-1428 shows that each bit lasts for 16 BCLK cycles. This is in case of 16× oversampling mode. For 13× oversampling mode each bit lasts for 13 BCLK cycles.

Figure 11-1428. Relationships Between Data Bit, BCLK, and UART Input Clock



11.18.4.3 Interrupt Requests

11.18.4.3.1 UART Interrupt Management

11.18.4.3.1.1 UART Interrupts

UART includes five possible interrupts prioritized to four levels.

When an interrupt is generated, the UART_IIR[0] IPEND bit is set to 0 to indicate that an interrupt is pending, and indicates the type of interrupt through the UART_IIR[3-1] INTID bit field. Table 11-3495 summarizes the interrupt control functions.

Table 11-3495. Interrupt Identification and Interrupt Clearing Information

Priority Level	UART_IIR Bits				Interrupt Type	Interrupt Source	Event That Clears Interrupt
	3	2	1	0			
None	0	0	0	1	None	None	None
1	0	1	1	0	Receiver line status	Overrun error, parity error, framing error, or break is detected.	For an overrun error, reading the UART_LSR clears the interrupt. For a parity error, framing error, or break, the interrupt is cleared only after all the erroneous data have been read.
2	0	1	0	0	Receiver data-ready	Non-FIFO mode: Receiver data is ready.	Non-FIFO mode: The UART_RBR is read.
						FIFO mode: Trigger level reached. If four character times (see Table 11-3496) pass with no access of the FIFO, the interrupt is asserted again.	FIFO mode: The FIFO drops below the trigger level. (1)
2	1	1	0	0	Receiver time-out	FIFO mode only: No characters have been removed from or input to the receiver FIFO during the last four character times (see Table 11-3496), and there is at least one character in the receiver FIFO during this time.	One of the following events: A character is read from the receiver FIFO (1), OR A new character arrives in the receiver FIFO, OR The UART_PWREMU_MGMT[13] URRST bit is loaded with 0.
3	0	0	1	0	Transmitter holding register empty	Non-FIFO mode: UART_THR is empty.	A character is written to the UART_THR.
						FIFO mode: Transmitter FIFO is empty.	
4	0	0	0	0	Modem status	Clear to send, data set ready, ring indicator, or data carrier detect	Reading the modem status register.

(1) In the FIFO mode, the receiver data-ready interrupt or receiver time-out interrupt is cleared by the CPU or by the DMA controller, whichever reads from the receiver FIFO first.

11.18.4.3.1.2 FIFO Management

The FIFO is accessed by reading and writing the UART_RBR and UART_THR registers. Parameters are controlled using the FIFO control register (UART_FCR).

The UART_FCR register controls the FIFO trigger level, which enables DMA and interrupt generation.

11.18.4.3.1.2.1 FIFO Modes

The following two modes can be used for servicing the receiver and transmitter FIFOs:

- FIFO interrupt mode. The FIFO is enabled and the associated interrupts are enabled. Interrupts are sent to the CPU to indicate when specific events occur.
- FIFO poll mode. The FIFO is enabled but the associated interrupts are disabled. The CPU polls status bits to detect specific events.

Because the receiver FIFO and the transmitter FIFO are controlled separately, either one or both can be placed into the interrupt mode or the poll mode.

11.18.4.3.1.2.1.1 FIFO Interrupt Mode

When the receiver FIFO is enabled in the UART_FCR and the receiver interrupts are enabled in the UART_IER, the interrupt mode is selected for the receiver FIFO. The following are important points about the receiver interrupts:

- The receiver data-ready interrupt is issued to the CPU when the FIFO has reached the trigger level that is programmed in UART_FCR. It is cleared when the CPU or the DMA controller reads enough characters from the FIFO such that the FIFO drops below its programmed trigger level.
- The receiver line status interrupt is generated in response to an overrun error, a parity error, a framing error, or a break. This interrupt has higher priority than the receiver data-ready interrupt. For details, see Section 11.18.4.4.2, *Interrupt Support*.
- The UART_LSR[0] DR bit (data-ready) indicates the presence or absence of characters in the receiver FIFO. The UART_LSR[0] DR bit is set when a character is transferred from the receiver shift register (UART_RSR) to the empty receiver FIFO. The UART_LSR[0] DR bit remains set until the FIFO is empty again.
- A receiver time-out interrupt occurs if all of the following conditions exist:

  - At least one character is in the FIFO.
  - The most recent character was received more than four continuous character times ago. A character time is the time allotted for 1 START bit, n data bits, 1 PARITY bit, and 1 STOP bit, where n depends on the word length selected with the UART_LCR[1-0] WLS bits. See Table 11-3496.
  - The most recent read of the FIFO has occurred more than four continuous character times before.
- Character times are calculated by using the baud rate.
- When a receiver time-out interrupt has occurred, it is cleared and the time-out timer is cleared when the CPU or the EDMA controller reads one character from the receiver FIFO. The interrupt is also cleared if a new character is received in the FIFO or if the UART_PWREMU_MGMT[13] URRST is cleared.
- If a receiver time-out interrupt has not occurred, the time-out timer is cleared after a new character is received or after the CPU or EDMA reads the receiver FIFO.

When the transmitter FIFO is enabled in UART_FCR and the transmitter holding register empty interrupt is enabled in UART_IER, the interrupt mode is selected for the transmitter FIFO. The transmitter holding register empty interrupt occurs when the transmitter FIFO is empty. It is cleared when the UART_THR is loaded (1 to 16 characters may be written to the transmitter FIFO while servicing this interrupt).

Table 11-3496. Character Time for Word Lengths

Word Length (n)	Character Time	Four Character Times
5	Time for 8 bits	Time for 32 bits
6	Time for 9 bits	Time for 36 bits
7	Time for 10 bits	Time for 40 bits
8	Time for 11 bits	Time for 44 bits

11.18.4.3.1.2.1.2 FIFO Poll Mode

When the receiver FIFO is enabled in the UART_FCR and the receiver interrupts are disabled in the UART_IER, the poll mode is selected for the receiver FIFO. Similarly, when the transmitter FIFO is enabled and the transmitter interrupts are disabled, the transmitted FIFO is in the poll mode. In the poll mode, the CPU detects events by checking bits in the UART_LSR:

- The UART_LSR[7] RXFIFOE bit indicates whether there are any errors in the receiver FIFO.
- The UART_LSR[6] TEMT bit indicates that both the transmitter holding register (UART_THR) and the transmitter shift register (UART_TSR) are empty.
- The UART_LSR[5] THRE bit indicates when UART_THR is empty.
- The UART_LSR[4] BI (break indicator), UART_LSR[3] FE (framing error), UART_LSR[2] PE (parity error), and UART_LSR[1] OE (overrun error) bits specify which error or errors have occurred.
- The UART_LSR[0] DR (data-ready) bit is set as long as there is at least one byte in the receiver FIFO.

Also, in the FIFO poll mode:

- The UART_IIR is not affected by any events because the interrupts are disabled.
- The UART does not indicate when the receiver FIFO trigger level is reached or when a receiver time-out occurs.

11.18.4.4 Protocol Formatting

The UART transmits in the following format:

1 START bit + data bits (5, 6, 7, 8) + 1 PARITY bit (optional) + STOP bit (1, 1.5, 2)

It transmits 1 START bit; 5, 6, 7, or 8 data bits, depending on the data width selection; 1 PARITY bit, if parity is selected; and 1, 1.5, or 2 STOP bits, depending on the STOP bit selection.

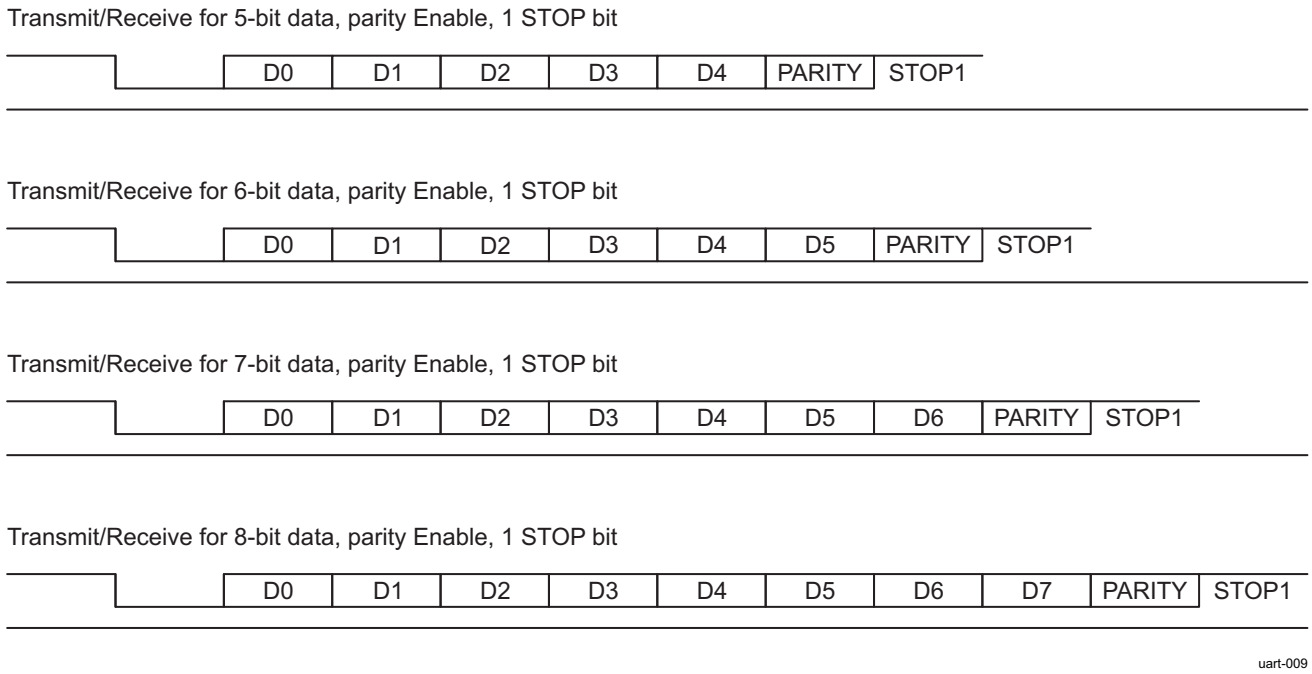
The UART receives in the following format:

1 START bit + data bits (5, 6, 7, 8) + 1 PARITY bit (optional) + 1 STOP bit

It receives 1 START bit; 5, 6, 7, or 8 data bits, depending on the data width selection; 1 PARITY bit, if parity is selected; and 1 STOP bit.

The protocol formats are shown in Figure 11-1429.

Figure 11-1429. UART Protocol Formats



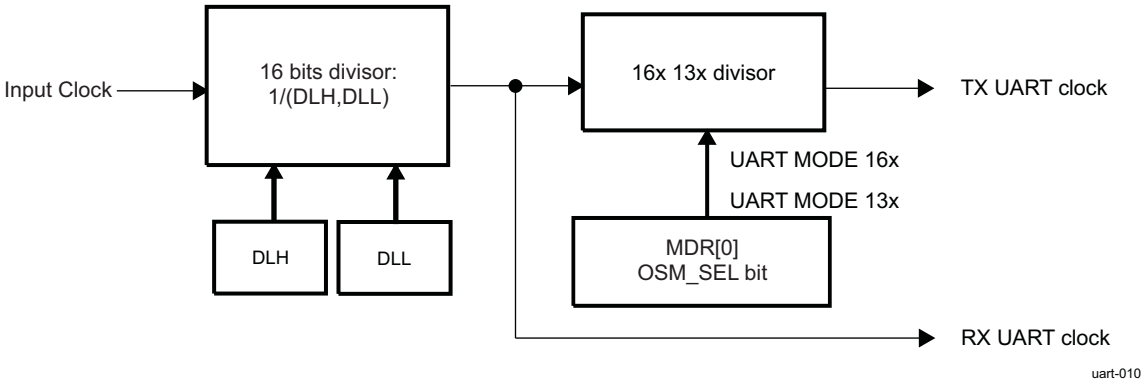
11.18.4.4.1 UART Mode

11.18.4.4.1.1 UART Clock Generation: Baud Rate Generation

The UART function contains a programmable baud generator and a set of fixed dividers that divide the functional clock input down to the expected baud rate.

Figure 11-1430 shows the baud rate generator and associated controls.

Figure 11-1430. Baud Rate Generation



11.18.4.4.1.2 Choosing the Appropriate Divisor Value

Table 11-3497 describes the UART baud rate settings (192-MHz Clock).

Table 11-3497. UART Baud Rate Settings (192-MHz Clock)

Baud Rate	Baud Multiple	DLH, DLL (Decimal) (Divisor value)	DLH, DLL (Hex)	Actual Baud Rate	Error (%)
1.2 kbps	16x	10000	27h, 10h	1.2 kbps	0
2.4 kbps	16x	5000	13h, 88h	2.4 kbps	0
4.8 kbps	16x	2500	09h, C4h	4.8 kbps	0
9.6 kbps	16x	1250	04h, E2h	9.6 kbps	0
14.4 kbps	16x	834	03h, 42h	14.388 kbps	-0.08
19.2 kbps	16x	625	02h, 71h	19.2 kbps	0
28.8 kbps	16x	417	01h, A1h	28.777 kbps	-0.08
38.4 kbps	16x	312	01h, 38h	38.462 kbps	+0.16
57.6 kbps	16x	208	00h, D0h	57.692 kbps	+0.16
115.2 kbps	16x	104	00h, 68h	115.385 kbps	+0.16
230.4 kbps	16x	52	00h, 34h	230.769 kbps	+0.16
460.8 kbps	16x	26	00h, 1Ah	461.538 kbps	+0.16
921.6 kbps	16x	13	00h, 0Dh	923.077 kbps	+0.16
1.8432 Mbps	13x	8	00h, 08h	1.846154 Mbps	+0.16
3.0 Mbps	16x	4	00h, 04h	3.0 Mbps	0
3.6864 Mbps	13x	4	00h, 04h	3.692308 Mbps	+0.16
7.3728 Mbps	13x	2	00h, 02h	7.384615 Mbps	+0.16
12.0 Mbps	16x	1	00h, 01h	12.0 Mbps	0

11.18.4.4.1.3 UART Data Formatting

The UART can use hardware flow control to manage transmission and reception. Hardware flow control significantly reduces software overhead and increases system efficiency by automatically controlling serial data flow using the RTS output and CTS input signals.

11.18.4.4.1.3.1 Frame Formatting

Character length is specified using the UART_LCR[1-0] WLS bit field (see Table 11-3499).

The number of stop-bits is specified using the UART_LCR[2] STB bit (see Table 11-3499).

The parity bit is programmed using the UART_LCR[5-3] PEN, EPS, and SP bits (see Table 11-3498).

Table 11-3498. Relationship Between ST, EPS, and PEN Bits in UART_LCR

ST Bit	EPS Bit	PEN Bit	Parity Option
x	x	0	Parity disabled: No PARITY bit is transmitted or checked.
0	0	1	Odd parity selected: Odd number of logic 1s.
0	1	1	Even parity selected: Even number of logic 1s.
1	0	1	Stick parity selected with PARITY bit transmitted and checked as set.
1	1	1	Stick parity selected with PARITY bit transmitted and checked as cleared.

Table 11-3499. Number of STOP Bits Generated

ST Bit	WLS Bit	Word Length Selected with WLS Bits	Number of STOP Bits Generated	Baud Clock (BCLK) Cycles
0	x	Any word length	1	16
1	0h	5 bits	1.5	24
1	1h	6 bits	2	32
1	2h	7 bits	2	32
1	3h	8 bits	2	32

11.18.4.4.1.4 Operation

11.18.4.4.1.4.1 Transmission

The UART transmitter section includes a transmitter hold register (UART_THR) and a transmitter shift register (UART_TSR). When the UART is in the FIFO mode, UART_THR is a 16-byte FIFO. Transmitter section control is a function of the UART line control register (UART_LCR). Based on the settings chosen in UART_LCR, the UART transmitter sends the following to the receiving device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1, 1.5, or 2 STOP bits

UART_THR receives data from the internal data bus, and when UART_TSR is ready, the UART moves the data from UART_THR to UART_TSR. The UART serializes the data in UART_TSR and transmits the data on the TX pin. In the non-FIFO mode, if UART_THR is empty and the UART_THR empty interrupt is enabled in the UART_IER, an interrupt is generated. This interrupt is cleared when a character is loaded into UART_THR. In the FIFO mode, the interrupt is generated when the transmitter FIFO is empty, and it is cleared when at least one byte is loaded into the FIFO.

11.18.4.4.1.4.2 Reception

The UART receiver section includes a receiver shift register (UART_RSR) and a receiver buffer register (UART_RBR). When the UART is in the FIFO mode, UART_RBR is a 16-byte FIFO. Receiver section control is a function of the UART line control register (UART_LCR). Based on the settings chosen in UART_LCR, the UART receiver accepts the following from the transmitting device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1 STOP bit (any other STOP bits transferred with the above data are not detected)

UART_RSR receives the data bits from the RX pin. Then UART_RSR concatenates the data bits and moves the resulting value into UART_RBR (or the receiver FIFO). The UART also stores three bits of error status information next to each received character, to record a parity error, framing error, or break.

In the non-FIFO mode, when a character is placed in UART_RBR and the receiver data-ready interrupt is enabled in the UART_IER, an interrupt is generated. This interrupt is cleared when the character is read from UART_RBR. In the FIFO mode, the interrupt is generated when the FIFO is filled to the trigger level selected in the UART_FCR, and it is cleared when the FIFO contents drop below the trigger level.

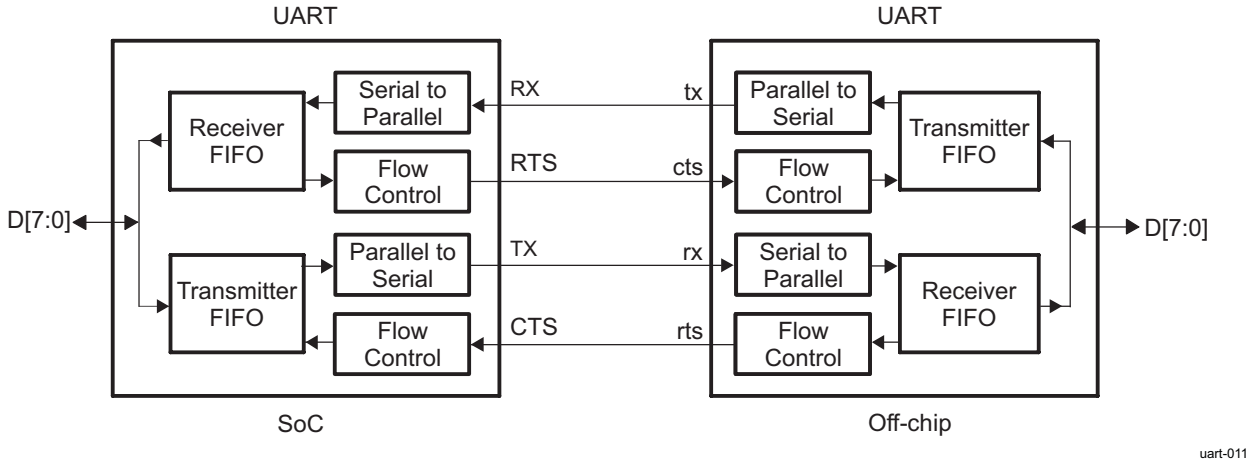
11.18.4.4.1.5 Hardware Flow Control

Hardware flow control is composed of RTS and CTS signals. They can be enabled and disabled by programming the UART_MCR[1] RTS and UART_MCR[5] AFE bit fields.

11.18.4.4.1.5.1 Autoflow Control

The UART can employ autoflow control by connecting the CTS and RTS signals. The CTS input must be active before the transmitter FIFO can transmit data. The RTS becomes active when the receiver needs more data and notifies the sending device. When RTS is connected to CTS, data transmission does not occur unless the receiver FIFO has space for the data. Therefore, when two UARTs are connected as shown in Figure 11-1431 with autoflow enabled, overrun errors are eliminated.

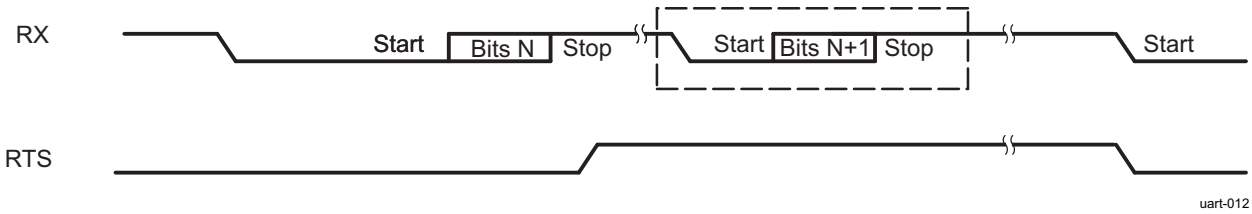
Figure 11-1431. Autoflow Control Example



11.18.4.4.1.5.1.1 RTS Behavior

The RTS data flow control originates in the receiver block (see Figure 11-1426). When the receiver FIFO level reaches a trigger level of 1, 4, 8, or 14 (see Figure 11-1432), RTS is deasserted. The sending UART may send an additional byte after the trigger level is reached (assuming the sending UART has another byte to send), because it may not recognize the deassertion of RTS until after it has begun sending the additional byte. For trigger level 1, 4, and 8, RTS is automatically reasserted after the receiver FIFO is emptied. For trigger level 14, RTS is automatically reasserted when the receiver FIFO drops below the trigger level.

Figure 11-1432. Autoflow Functional Timing Waveforms for RTS



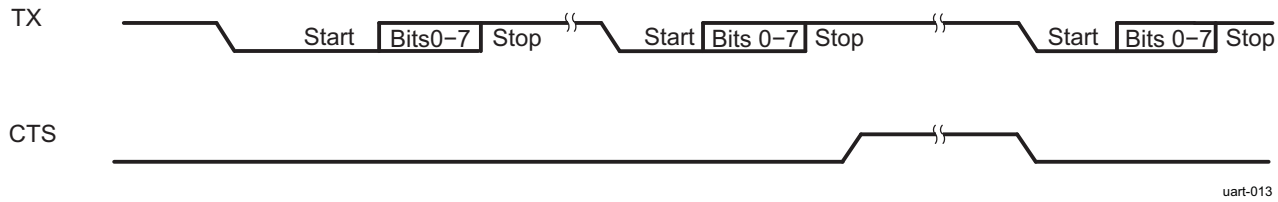
A The N = Receiver FIFO trigger level.

B The two blocks in dashed lines cover the case in which an additional byte is sent.

11.18.4.4.1.5.1.2 CTS Behavior

The transmitter checks CTS before sending the next data byte. If CTS is active, the transmitter sends the next byte. To stop the transmitter from sending the following byte, CTS must be released before the middle of the last STOP bit that is currently being sent (see Figure 11-1433). When flow control is enabled, CTS level changes do not trigger interrupts because the device automatically controls its own transmitter. Without autoflow control, the transmitter sends any data present in the transmitter FIFO and a receiver overrun error may result.

Figure 11-1433. Autoflow Functional Timing Waveforms for CTS



A When CTS is active (low), the transmitter keeps sending serial data out.

B When CTS goes high before the middle of the last STOP bit of the current byte, the transmitter finishes sending the current byte but it does not send the next byte.

C When CTS goes from high to low, the transmitter begins sending data again.

11.18.4.4.1.5.2 Loopback Control

The UART can be placed in the diagnostic mode using the LOOP bit in the modem control register (UART_MCR), which internally connects the UART output back to the UART input. In this mode, the transmit and receive data paths, the transmitter and receiver interrupts, and the modem control interrupts can be verified without connecting to another UART.

11.18.4.4.1.5.3 Modem Signal Control

Modem signals RTS, DTR, OUT1, and OUT2 can be controlled and the modem signals CTS, DCD, DSR and RI can be observed by using the Modem Control Register (UART_MCR) and the Modem Status Register (UART_MSR), respectively. Modem control function of these inputs and outputs except for RTS and CTS must be implemented in software. RTS and CTS are used for autoflow control (see Section 11.18.4.4.1.5.1, *Autoflow Control* for details) and controlled by UART hardware in that mode.

11.18.4.4.1.6 Reset Considerations

11.18.4.4.1.6.1 Software Reset Considerations

Three bits in the UART_PWREMU_MGMT control resetting the parts of the UART:

- The UART_PWREMU_MGMT[15] URST bit resets both transmitter and receiver.
- The UART_PWREMU_MGMT[14] UTRST bit controls resetting the transmitter only.
- The UART_PWREMU_MGMT[13] URRST bit controls resetting the receiver only.

For backward compatibility, the UART_PWREMU_MGMT[14] UTRST and UART_PWREMU_MGMT[13] URRST are automatically set when the UART_PWREMU_MGMT[15] URST is set. Table 11-3500 shows truth table and functions of the three software reset bits. These software bits affect only transmitter and receiver, and they don't affect modem control signals. The control of modem signals are responsibility of software.

Table 11-3500. Software Reset Bit Truth Table

Write Data[15-13]	URST	UTRST	URRST	Transmitter	Receiver
000	0	0	0	Disabled	Disabled
001	0	0	1	Disabled	Enabled
010	0	1	0	Enabled	Disabled

Table 11-3500. Software Reset Bit Truth Table (continued)

Write Data[15-13]	URST	UTRST	URRST	Transmitter	Receiver
1xx	0	1	1	Enabled	Enabled

NOTE: RTS in auto-flow control mode must be taken care of especially. User must set UART_MCR[1] RTS bit after setting UART_PWREMU_MGMT[15] URST or UART_PWREMU_MGMT[13] URRST bit. Otherwise, UART_MCR[1] RTS output goes to low indicating the UART is read to receive data while the receiver of the UART is actually disabled.

11.18.4.4.2 Interrupt Support

11.18.4.4.2.1 Interrupt Events and Requests

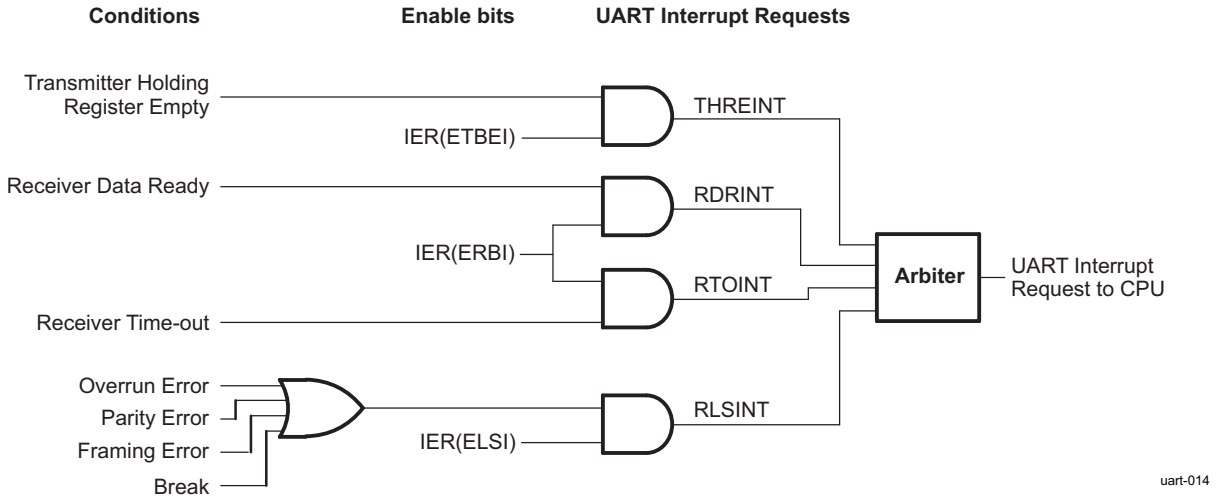
The UART generates the interrupt requests described in Table 11-3501. All requests are multiplexed through an arbiter to a single UART interrupt request to the CPU, as shown in Figure 11-1434. Each of the interrupt requests has an enable bit in the UART_IER and is recorded in the UART_IIR.

If an interrupt occurs and the corresponding enable bit is set to 1, the interrupt request is recorded in UART_IIR and is forwarded to the CPU. If an interrupt occurs and the corresponding enable bit is cleared to 0, the interrupt request is blocked. The interrupt request is neither recorded in UART_IIR nor forwarded to the CPU.

Table 11-3501. UART Interrupt Requests Descriptions

UART Interrupt Request	Interrupt Source	Comment
THREINT	THR-empty condition: The UART_THR or the transmitter FIFO is empty. All of the data has been copied from UART_THR to the transmitter shift register (UART_TSR).	• If THREINT is enabled by setting the UART_IER[1] ETBEI bit, it is recorded in UART_IIR. • As an alternative to using THREINT, the CPU can poll the UART_LSR[5] THRE bit.
RDAINT	Receive data available in non-FIFO mode or trigger level reached in the FIFO mode.	• If RDAINT is enabled by setting the UART_IER[0] ERBI bit, it is recorded in UART_IIR. • As an alternative to using RDAINT, the CPU can poll the UART_LSR[0] DR bit. In the FIFO mode, this is not a functionally equivalent alternative because the UART_LSR[0] DR bit does not respond to the FIFO trigger level. The UART_LSR[0] DR bit indicates only the presence or absence of unread characters.
RTOINT	Receiver time-out condition (in the FIFO mode only): No characters have been removed from or input to the receiver FIFO during the last four character times (see Table 11-3496), and there is at least one character in the receiver FIFO during this time.	• The receiver time-out interrupt prevents the UART from waiting indefinitely when the receiver FIFO level is below the trigger level and, therefore, does not generate a receiver data-ready interrupt. • If RTOINT is enabled by setting the UART_IER[0] ERB bit, it is recorded in UART_IIR. • There is no status bit to reflect the occurrence of a time-out condition.
RLSINT	Receiver line status condition: An overrun error, parity error, framing error, or break has occurred.	• If RLSINT is enabled by setting the UART_IER[2] ELSI bit, it is recorded in UART_IIR. • As an alternative to using RLSINT, the CPU can poll the following bits UART_LSR[1] OE, UART_LSR[2] PE, UART_LSR[3] FE, and UART_LSR[4] BI.

Figure 11-1434. UART Interrupt Request Enable Paths



11.18.4.4.3 DMA Event Support

In the FIFO mode, the UART generates the following two DMA events:

- **Receive event (URXEVT):** The trigger level for the receiver FIFO (1, 4, 8, or 14 characters) is set with the UART_FCR[7-6] RXFIFTL bit. Every time the trigger level is reached or a receiver time-out occurs, the UART sends a receive event to the EDMA controller. In response, the EDMA controller reads the data from the receiver FIFO by way of the UART_RBR.
- **Transmit event (UTXEVT):** When the transmitter FIFO is empty (when the last byte in the transmitter FIFO has been copied to the transmitter shift register), the UART sends an UTXEVT signal to the EDMA controller. In response, the EDMA controller refills the transmitter FIFO by way of the UART_THR. The UTXEVT signal is also sent to the DMA controller when the UART is taken out of reset using the UART_PWREMU_MGMT[14] UTRST bit.

Activity in DMA channels can be synchronized to these events. In the non-FIFO mode, the UART generates no DMA events. Any DMA channel synchronized to either of these events must be enabled at the time the UART event is generated. Otherwise, the DMA channel will miss the event and, unless the UART generates a new event, no data transfer will occur.

11.18.4.4.4 Module Power Management

The UART peripheral can be placed in reduced-power modes to conserve power during periods of low activity. The power management of the UART peripheral is controlled by the Power and Sleep Controller (PSC). The PSC acts as a master controller for power management for all of the peripherals on the device. When PSC requests the UART to be in idle, if a data is transferred, the UART transfers the data completely, asserts an interrupt or a DMA event if it is set up as it does so, and enter IDLE state. If no data is transferred, the UART enters IDLE state immediately. In the IDLE state, the UART holds state machines in the transmitter and the receiver to prevent them starting the next transfer. The transmitter and the receiver go into IDLE state independently except in loop back mode. For detailed information on power management procedures using the PSC, see Section 5.2, *Power Management* in *Device Configuration*.

11.18.4.4.5 Emulation Considerations

The UART_PWREMU_MGMT[0] FREE bit determines how the UART responds to an emulation suspend event such as an emulator halt or breakpoint. If UART_PWREMU_MGMT[0] FREE = 0 and a transmission is in progress, the UART halts after completing the one-word transmission; if UART_PWREMU_MGMT[0] FREE = 0 and a transmission is not in progress, the UART halts immediately. If UART_PWREMU_MGMT[0] FREE = 1, the UART does not halt and continues operating normally.

Note also that emulator accesses are essentially transparent to UART operation. Emulator read operations do not affect any register contents, status bits, or operating states. Emulator writes, however, may affect register contents and may affect UART operation, depending on what register is accessed and what value is written.

The UART registers can be read from or written to during emulation suspend events, even if the UART activity has stopped.

11.18.4.5 Exception Processing

11.18.4.5.1 Divisor Latch Not Programmed

Since the processor reset signal has no effect on the divisor latch, the divisor latch will have an unknown value after power up. If the divisor latch is not programmed after power up, the baud clock (BCLK) will not operate and will instead be set to a constant logic 1 state.

The divisor latch values should always be reinitialized following a processor reset.

11.18.4.5.2 Changing Operating Mode During Busy Serial Communication

Since the serial link characteristics are based on how the control registers are programmed, the UART will expect the control registers to be static while it is busy engaging in a serial communication. Therefore, changing the control registers while the module is still busy communicating with another serial device will most likely cause an error condition and should be avoided.

11.18.5 UART Basic Programming Model

This section describes the procedure for operating the UART with FIFO and DMA or interrupts. This procedure ensures the quick start of the UART. It does not cover every UART feature.

11.18.5.1 Global Initialization

11.18.5.1.1 Surrounding Modules Global Initialization

This section identifies the requirements for initializing the surrounding modules when the UART module is to be used for the first time after a device reset. This initialization of surrounding modules is based on the integration and environment of the UART. Refer to the UART Module Integration and Environment Sections for further information.

Table 11-3502. Global Initialization of Surrounding Modules for UART

Surrounding Modules	Comments
CIC	Device INTCs must be configured to enable the interrupt request generation. For information about enabling CIC interrupts, see Chapter 9, *Interrupts*.
ARM_GIC	Device INTCs must be configured to enable the interrupt request generation. For information about enabling ARM_GIC interrupts, see Section 6.1, *Arm Cortex-A15 Subsystem*.
EDMACC_0	DMA controllers configuration must be done to enable the module DMA channel request. See Chapter 10, *DMA Controllers*.
LPSC6	Module reset must be enabled. For more information about the module configuration, see Section 5.2, *Power Management*.
UART PLL	UART PLL configuration must be done to enable the clocks to the UART modules. See Section 5.4.5.3, *PLL Controller*.

11.18.5.1.1.1 UART Module Global Initialization

The following steps are required to initialize the UART:

Table 11-3503. Global Initialization of UART

Step	Register/Bit Field/Programming Model	Value
Set the desired baud rate by writing the appropriate clock divisor values.	UART_DLL[7-0] DLL and UART_DLH[7-0] DLH See Table 11-3497, *UART Baud Rate Settings (192-MHz Clock)*	-h and -h
If the FIFOs will be used, select the desired trigger level and enable the FIFOs by writing the appropriate values to the UART_FCR. **NOTE:** UART_FCR[0] FIFOEN bit must be set first, before the other bits in UART_FCR are configured.	UART_FCR[7-6] RXFIFTL	-h
Choose the desired protocol settings by writing the appropriate values to the UART_LCR.	See Table 11-3498 and Table 11-3499	-h
If autoflow control is desired set Autoflow control enable bit.	UART_MCR[5] AFE	1h
Also write the appropriate values to the RTS control bit.	UART_MCR[1] RTS	-h
Choose the desired response to emulation suspend events.	Configure UART_PWREMU_MGMT[0] FREE bit	-h
Enable the UART.	Set the UART_PWREMU_MGMT[15] URST and UART_PWREMU_MGMT[14] UTRST bits	-h

11.18.6 UART Registers

Table 11-3505 lists the memory-mapped registers for the UART. All register offset addresses not listed in Table 11-3505 should be considered as reserved locations and the register contents should not be modified.

Table 11-3504. UART Instances

Instance	Base Address
UART_0	0253 0C00h
UART_1	0253 1000h
UART_2	0253 1400h

Table 11-3505. UART Registers

Offset	Acronym	Register Name	UART_0 Physical Address	UART_1 Physical Address	UART_2 Physical Address	Section
0h	UART_RBR	Receiver Buffer Register	0253 0C00h	0253 1000h	0253 1400h	Section 11.18.6.1
0h	UART_THR	Transmitter Holding Register	0253 0C00h	0253 1000h	0253 1400h	Section 11.18.6.2
4h	UART_IER	Interrupt Enable Register	0253 0C04h	0253 1004h	0253 1404h	Section 11.18.6.3
8h	UART_IIR	Interrupt Identification Register	0253 0C08h	0253 1008h	0253 1408h	Section 11.18.6.4
8h	UART_FCR	FIFO Control Register	0253 0C08h	0253 1008h	0253 1408h	Section 11.18.6.5
Ch	UART_LCR	Line Control Register	0253 0C0Ch	0253 100Ch	0253 140Ch	Section 11.18.6.6
10h	UART_MCR	Modem Control Register	0253 0C10h	0253 1010h	0253 1410h	Section 11.18.6.7
14h	UART_LSR	Line Status Register	0253 0C14h	0253 1014h	0253 1414h	Section 11.18.6.8
18h	UART_MSR	Modem Status Register	0253 0C18h	0253 1018h	0253 1418h	Section 11.18.6.9
1Ch	UART_SCR	Scratch Pad Register	0253 0C1Ch	0253 101Ch	0253 141Ch	Section 11.18.6.10
20h	UART_DLL	Divisor LSB Latch	0253 0C20h	0253 1020h	0253 1420h	Section 11.18.6.11
24h	UART_DLH	Divisor MSB Latch	0253 0C24h	0253 1024h	0253 1424h	Section 11.18.6.12
28h	UART_PID	Peripheral Identification Register	0253 0C28h	0253 1028h	0253 1428h	Section 11.18.6.13
30h	UART_PWREMU_MGMT	Power and Emulation Management Register	0253 0C30h	0253 1030h	0253 1430h	Section 11.18.6.14
34h	UART_MDR	Mode Definition Register	0253 0C34h	0253 1034h	0253 1434h	Section 11.18.6.15

11.18.6.1 UART_RBR Register (Offset = 0h) [reset = 0h]

UART_RBR is shown in Figure 11-1435 and described in Table 11-3507.

The UART receiver section consists of a receiver shift register (UART_RSR) and a receiver buffer register (UART_RBR). When the UART is in the FIFO mode, UART_RBR is a 16-byte FIFO. Timing is supplied by the 16× receiver clock or 13× receiver clock by programming UART_MDR[0] OSM_SEL bit. Receiver section control is a function of the UART_LCR.

The UART_RSR receives serial data from the RX pin. Then the UART_RSR concatenates the data and moves it into the UART_RBR (or the receiver FIFO). In the non-FIFO mode, when a character is placed in UART_RBR and the receiver data-ready interrupt is enabled (UART_IER[0] DR = 1), an interrupt is generated. This interrupt is cleared when the character is read from the UART_RBR. In the FIFO mode, the interrupt is generated when the FIFO is filled to the trigger level selected in the UART_FCR, and it is cleared when the FIFO contents drop below the trigger level.

Access considerations:

- The UART_RBR, UART_THR, and UART_DLL share one address. To read the UART_RBR, write 0 to the UART_LCR[7] DLAB bit, and read from the shared address. When UART_LCR[7] DLAB = 0, writing to the shared address modifies the UART_THR. When UART_LCR[7] DLAB = 1, all accesses at the shared address read or modify the UART_DLL.
- The UART_DLL also has a dedicated address. If the dedicated address is used, UART_LCR[7] DLAB can = 0, so that the UART_RBR and UART_THR are always selected at the shared address.

Table 11-3506. UART_RBR Instances

Instance	Physical Address
UART_0	0253 0C00h
UART_1	0253 1000h
UART_2	0253 1400h

Figure 11-1435. UART_RBR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DATA							
R-0h																								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3507. UART_RBR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	DATA	R	0h	Received data

Table 11-3508. Register Call Summary for UART_RBR

UART Registers • UART_LCR Register (Offset = Ch) [reset = 0h]: [0][1][2][3][4] • UART_LSR Register (Offset = 14h) [reset = 60h]: [0][1][2][3][4][5][6][7][8][9][10][11][12] • UART Registers: [0] • UART_RBR Register (Offset = 0h) [reset = 0h]: [0][1][2][3][4][5][6][7][8] • UART_DLL Register (Offset = 20h) [reset = 0h]: [0][1][2] • UART_THR Register (Offset = 0h) [reset = 0h]: [0][1][2]
UART Functional Description • DMA Event Support: [0] • Operation: [0][1][2][3][4] • FIFO Management: [0]

11.18.6.2 UART_THR Register (Offset = 0h) [reset = 0h]

UART_THR is shown in Figure 11-1436 and described in Table 11-3510.

The UART transmitter section consists of a transmitter hold register (UART_THR) and a transmitter shift register (UART_TSR). Transmitter control is a function of the UART_LCR.

The UART_THR receives data from the internal data bus. When the UART_TSR is idle the UART then moves the data from the UART_THR to the UART_TSR. The UART serializes the data in the UART_TSR and transmits the data on the TX pin.

In the non-FIFO mode, if the UART_THR and UART_TSR is empty and the THRE interrupt is enabled (UART_IER[1] ETBEI = 1), then an interrupt is generated. This interrupt is cleared when a character is loaded into the UART_THR.

When the UART is in the FIFO mode, the UART_THR is a 16-byte FIFO. In the FIFO mode, the interrupt is generated when the transmitter FIFO is empty, and it is cleared when at least one byte is loaded into the FIFO.

Access considerations:

- The UART_RBR, UART_THR, and UART_DLL share one address. To load the UART_THR, write 0 to the UART_LCR[7] DLAB bit, and write to the shared address. When UART_LCR[7] DLAB = 0, reading from the shared address gives the content of the UART_RBR. When UART_LCR[7] DLAB = 1, all accesses at the address read or modify the UART_DLL.
- The UART_DLL also has a dedicated address. If the dedicated address is used, UART_LCR[7] DLAB can = 0, so that the UART_RBR and the UART_THR are always selected at the shared address.

Table 11-3509. UART_THR Instances

Instance	Physical Address
UART_0	0253 0C00h
UART_1	0253 1000h
UART_2	0253 1400h

Figure 11-1436. UART_THR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DATA							
R-0h																								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3510. UART_THR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	DATA	R	0h	Data to transmit

Table 11-3511. Register Call Summary for UART_THR

UART Registers
- UART_LCR Register (Offset = Ch) [reset = 0h]: [0][1][2][3][4]
- UART_LSR Register (Offset = 14h) [reset = 60h]: [0][1][2][3][4][5]
- UART Registers: [0]
- UART_RBR Register (Offset = 0h) [reset = 0h]: [0][1][2]
- UART_DLL Register (Offset = 20h) [reset = 0h]: [0][1][2]
- UART_THR Register (Offset = 0h) [reset = 0h]: [0][1][2][3][4][5][6][7][8][9]

Table 11-3511. Register Call Summary for UART_THR (continued)

UART Functional Description • DMA Event Support: [0] • Operation: [0][1][2][3][4][5][6] • FIFO Management: [0][1][2][3] • Interrupt Events and Requests: [0][1] • UART Interrupts: [0][1]

11.18.6.3 UART_IER Register (Offset = 4h) [reset = 0h]

UART_IER is shown in Figure 11-1437 and described in Table 11-3513.

The interrupt enable register (UART_IER) is used to individually enable or disable each type of interrupt request that can be generated by the UART. Each interrupt request that is enabled in the UART_IER is forwarded to the CPU.

Access considerations:

- The UART_IER and UART_DLH share one address. To read or modify the UART_IER, write 0 to the UART_LCR[7] DLAB bit. When UART_LCR[7] DLAB = 1, all accesses at the shared address read or modify the UART_DLH.
- The UART_DLH also has a dedicated address. If the dedicated address is used, UART_LCR[7] DLAB can = 0, so that the UART_IER is always selected at the shared address.

Table 11-3512. UART_IER Instances

Instance	Physical Address
UART_0	0253 0C04h
UART_1	0253 1004h
UART_2	0253 1404h

Figure 11-1437. UART_IER Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				EDSSI	ELSI	ETBEI	ERBI
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3513. UART_IER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	EDSSI	R/W	0h	Modem status interrupt enable. 0 = Modem status interrupt is disabled. 1 = Modem status interrupt is enabled.
2	ELSI	R/W	0h	Receiver line status interrupt enable. 0 = Receiver line status interrupt is disabled. 1 = Receiver line status interrupt is enabled.
1	ETBEI	R/W	0h	Transmitter holding register empty interrupt enable. 0 = Transmitter holding register empty interrupt is disabled. 1 = Transmitter holding register empty interrupt is enabled.

Table 11-3513. UART_IER Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	ERBI	R/W	0h	Receiver data available interrupt and character timeout indication interrupt enable. 0 = Receiver data available interrupt and character timeout indication interrupt is disabled. 1 = Receiver data available interrupt and character timeout indication interrupt is enabled.

Table 11-3514. Register Call Summary for UART_IER

UART Registers
- UART_LSR Register (Offset = 14h) [reset = 60h]: [0][1][2][3][4]
- UART_LCR Register (Offset = Ch) [reset = 0h]: [0][1][2][3][4][5]
- UART Registers: [0]
- UART_IIR Register (Offset = 8h) [reset = 1h]: [0][1]
- UART_RBR Register (Offset = 0h) [reset = 0h]: [0]
- UART_DLL Register (Offset = 20h) [reset = 0h]: [0][1][2]
- UART_IER Register (Offset = 4h) [reset = 0h]: [0][1][2][3][4][5]
- UART_THR Register (Offset = 0h) [reset = 0h]: [0]

UART Functional Description
- Operation: [0][1]
- FIFO Management: [0][1][2]
- Interrupt Events and Requests: [0][1][2][3][4]

11.18.6.4 UART_IIR Register (Offset = 8h) [reset = 1h]

UART_IIR is shown in Figure 11-1438 and described in Table 11-3516.

The interrupt identification register (UART_IIR) is a read-only register at the same address as the UART_FCR, which is a write-only register. When an interrupt is generated and enabled in the UART_IER, the UART_IIR indicates that an interrupt is pending in the UART_IIR[0] IPEND bit and encodes the type of interrupt in the UART_IIR[3-1] INTID bits.

The UART has an on-chip interrupt generation and prioritization capability that permits flexible communication with the CPU. The UART provides four priority levels of interrupts:

- Priority 1 - Receiver line status (highest priority)
- Priority 2 - Receiver data ready or receiver timeout
- Priority 3 - Transmitter holding register empty
- Priority 4 - Modem status (lowest priority)

The UART_IIR[7-6] FIFOEN bit field can be checked to determine whether the UART is in the FIFO mode or the non-FIFO mode.

Access considerations:

- The UART_IIR and UART_FCR share one address. Regardless of the value of the UART_LCR[7] DLAB bit, reading from the address gives the content of the UART_IIR, and writing to the address modifies the UART_FCR.

Table 11-3515. UART_IIR Instances

Instance	Physical Address
UART_0	0253 0C08h
UART_1	0253 1008h
UART_2	0253 1408h

Figure 11-1438. UART_IIR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
FIFOEN		RESERVED		INTID			IPEND
R-0h		R-0h		R-0h			R-1h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3516. UART_IIR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	FIFOEN	R	0h	FIFOs enabled. 0h = Non-FIFO mode. 1h = Reserved. 2h = Reserved. 3h = FIFOs are enabled. UART_IIR[7-6] FIFOEN bit is set to 1.
5-4	RESERVED	R	0h	Reserved

Table 11-3516. UART_IIR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-1	INTID	R	0h	Interrupt type. See Table 11-3495 0h = Modem status (priority 4, lowest). 1h = Transmitter holding register empty (priority 3). 2h = Receiver data available (priority 2). 3h = Receiver line status (priority 1, highest). 4h = Reserved. 5h = Reserved. 6h = Character timeout indication (priority 2). 7h = Reserved.
0	IPEND	R	1h	Interrupt pending. When any UART interrupt is generated and is enabled in UART_IER, IPEND is forced to 0. IPEND remains 0 until all pending interrupts are cleared or until a hardware reset occurs. If no interrupts are enabled, IPEND is never forced to 0. 0 = Interrupts pending. 1 = No interrupts pending.

Table 11-3517. Register Call Summary for UART_IIR

UART Registers • UART Registers: [0] • UART_IIR Register (Offset = 8h) [reset = 1h]: [0][1][2][3][4][5][6][7][8] • UART_FCR Register (Offset = 8h) [reset = 0h]: [0][1][2]
UART Functional Description • FIFO Management: [0] • Interrupt Events and Requests: [0][1][2][3][4][5][6]

11.18.6.5 UART_FCR Register (Offset = 8h) [reset = 0h]

UART_FCR is shown in Figure 11-1439 and described in Table 11-3519.

The FIFO control register (UART_FCR) is a write-only register at the same address as the interrupt identification register (UART_IIR), which is a read-only register. Use the UART_FCR to enable and clear the FIFOs and to select the receiver FIFO trigger level. The FIFOEN bit must be set to 1 before other UART_FCR bits are written to or the UART_FCR bits are not programmed.

Access considerations:

- The UART_IIR and UART_FCR share one address. Regardless of the value of the DLAB bit, reading from the address gives the content of the UART_IIR, and writing to the address modifies the UART_FCR.

CAUTION

For proper communication between the UART and the EDMA controller, the DMAMODE1 bit must be set to 1. Always write a 1 to the DMAMODE1 bit, and after a hardware reset, change the DMAMODE1 bit from 0 to 1.

Table 11-3518. UART_FCR Instances

Instance	Physical Address
UART_0	0253 0C08h
UART_1	0253 1008h
UART_2	0253 1408h

Figure 11-1439. UART_FCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RXFIFTL		RESERVED		DMAMODE1	TXCLR	RXCLR	FIFOEN
R-0h		R-0h		W-0h	W1toCl-0h	W1toCl-0h	W-0h

LEGEND: R = Read Only; W = Write Only; W1toCl = Write 1 to Clear Bit; *-n* = value after reset

Table 11-3519. UART_FCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	RXFIFTL	R	0h	Receiver FIFO trigger level. RXFIFTL sets the trigger level for the receiver FIFO. When the trigger level is reached, a receiver data-ready interrupt is generated (if the interrupt request is enabled). When the FIFO drops below the trigger level, the interrupt is cleared. 0h = 1 byte. 1h = 4 bytes. 2h = 8 bytes. 3h = 14 bytes.

Table 11-3519. UART_FCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	RESERVED	R	0h	Reserved
3	DMAMODE1	W	0h	DMA MODE1 enable if FIFOs are enabled. 0 = DMA MODE1 is disabled. 1 = DMA MODE1 is enabled.
2	TXCLR	W1toCl	0h	Transmitter FIFO clear. Write a 1 to TXCLR to clear the bit. 0 = No effect. 1 = Clears transmitter FIFO and resets the transmitter FIFO counter. The shift register is not cleared.
1	RXCLR	W1toCl	0h	Receiver FIFO clear. Write a 1 to RXCLR to clear the bit. 0 = No effect. 1 = Clears receiver FIFO and resets the receiver FIFO counter. The shift register is not cleared.
0	FIFOEN	W	0h	Transmitter and receiver FIFOs mode enable. FIFOEN must be set before other FCR bits are written to or the FCR bits are not programmed. Clearing this bit clears the FIFO counters. 0 = Non-FIFO mode. The transmitter and receiver FIFOs are disabled, and the FIFO pointers are cleared. 1 = FIFO mode. The transmitter and receiver FIFOs are enabled.

Table 11-3520. Register Call Summary for UART_FCR

UART Registers • UART Registers: [0] • UART_IIR Register (Offset = 8h) [reset = 1h]: [0][1][2] • UART_RBR Register (Offset = 0h) [reset = 0h]: [0] • UART_FCR Register (Offset = 8h) [reset = 0h]: [0][1][2][3][4][5][6]
UART Functional Description • DMA Event Support: [0] • Operation: [0] • FIFO Management: [0][1][2][3][4][5]
UART Basic Programming Model • UART Module Global Initialization: [0][1][2][3]

11.18.6.6 UART_LCR Register (Offset = Ch) [reset = 0h]

UART_LCR is shown in Figure 11-1440 and described in Table 11-3522.

The system programmer controls the format of the asynchronous data communication exchange by using the line control register (UART_LCR). In addition, the programmer can retrieve, inspect, and modify the content of the UART_LCR. This eliminates the need for separate storage of the line characteristics in system memory.

Table 11-3521. UART_LCR Instances

Instance	Physical Address
UART_0	0253 0C0Ch
UART_1	0253 100Ch
UART_2	0253 140Ch

Figure 11-1440. UART_LCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
DLAB	BC	SP	EPS	PEN	STB	WLS	
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3522. UART_LCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved.
7	DLAB	R/W	0h	Divisor latch access bit. The divisor latch registers (UART_DLL and UART_DLH) can be accessed at dedicated addresses or at addresses shared by UART_RBR, UART_THR, and UART_IER. Using the shared addresses requires toggling DLAB to change which registers are selected. If the dedicated addresses are used, DLAB can = 0. 0 = Allows access to the receiver buffer register (UART_RBR), the transmitter holding register (UART_THR), and the interrupt enable register (UART_IER) selected. At the address shared by UART_RBR, UART_THR, and UART_IER, the CPU can read from UART_RBR and write to UART_THR. At the address shared by UART_IER and UART_DLH, the CPU can read from and write to UART_IER. 1 = Allows access to the divisor latches of the baud generator during a read or write operation (UART_DLL and UART_DLH). At the address shared by UART_RBR, UART_THR, and UART_DLL, the CPU can read from and write to UART_DLL. At the address shared by UART_IER and UART_DLH, the CPU can read from and write to UART_DLH.
6	BC	R/W	0h	Break control. 0 = Break condition is disabled. 1 = Break condition is transmitted to the receiving UART. A break condition is a condition in which the TX signal is forced to the spacing (cleared) state.

Table 11-3522. UART_LCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	SP	R/W	0h	Stick parity. The SP bit works in conjunction with the EPS and PEN bits. The relationship between the SP, EPS, and PEN bits is summarized in Table 11-3498. 0 = Stick parity is disabled. 1 = Stick parity is enabled. When odd parity is selected (EPS = 0), the PARITY bit is transmitted and checked as set. When even parity is selected (EPS = 1), the PARITY bit is transmitted and checked as cleared.
4	EPS	R/W	0h	Even parity select. Selects the parity when parity is enabled (PEN = 1). The EPS bit works in conjunction with the SP and PEN bits. The relationship between the SP, EPS, and PEN bits is summarized in Table 11-3498. 0 = Odd parity is selected (an odd number of logic 1s is transmitted or checked in the data and PARITY bits). 1 = Even parity is selected (an even number of logic 1s is transmitted or checked in the data and PARITY bits).
3	PEN	R/W	0h	Parity enable. The PEN bit works in conjunction with the SP and EPS bits. The relationship between the SP, EPS, and PEN bits is summarized in Table 11-3498. 0 = No PARITY bit is transmitted or checked. 1 = Parity bit is generated in transmitted data and is checked in received data between the last data word bit and the first STOP bit.
2	STB	R/W	0h	Number of STOP bits generated. STB specifies 1, 1.5, or 2 STOP bits in each transmitted character. When STB = 1, the WLS bit determines the number of STOP bits. The receiver clocks only the first STOP bit, regardless of the number of STOP bits selected. The number of STOP bits generated is summarized in Table 11-3499. 0 = One STOP bit is generated. 1 = WLS bit determines the number of STOP bits: • When WLS = 0, 1.5 STOP bits are generated. • When WLS = 1h, 2h, or 3h, 2 STOP bits are generated.
1-0	WLS	R/W	0h	Word length select. Number of bits in each transmitted or received serial character. When STB = 1, the WLS bit determines the number of STOP bits. 0h = 5 bits 1h = 6 bits 2h = 7 bits 3h = 8 bits

Table 11-3523. Register Call Summary for UART_LCR

UART Registers • UART_LSR Register (Offset = 14h) [reset = 60h]: [0] • UART_LCR Register (Offset = Ch) [reset = 0h]: [0][1][2] • UART Registers: [0] • UART_IIR Register (Offset = 8h) [reset = 1h]: [0] • UART_RBR Register (Offset = 0h) [reset = 0h]: [0][1][2][3][4] • UART_DLL Register (Offset = 20h) [reset = 0h]: [0][1] • UART_IER Register (Offset = 4h) [reset = 0h]: [0][1][2] • UART_THR Register (Offset = 0h) [reset = 0h]: [0][1][2][3][4]
UART Functional Description • Operation: [0][1][2][3] • FIFO Management: [0] • UART Data Formatting: [0][1][2]
UART Basic Programming Model • UART Module Global Initialization: [0]

11.18.6.7 UART_MCR Register (Offset = 10h) [reset = 0h]

UART_MCR is shown in Figure 11-1441 and described in Table 11-3525.

The modem control register (UART_MCR) provides the ability to enable/disable the autoflow functions, and enable/disable the loopback function for diagnostic purposes.

Table 11-3524. UART_MCR Instances

Instance	Physical Address
UART_0	0253 0C10h
UART_1	0253 1010h
UART_2	0253 1410h

Figure 11-1441. UART_MCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		AFE	LOOP	OUT2	OUT1	RTS	DTR
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3525. UART_MCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	AFE	R/W	0h	Autoflow control enable. Autoflow control allows the RTS and CTS signals to provide handshaking between UARTs during data transfer. When AFE = 1, the RTS bit determines the autoflow control enabled. 0 = Autoflow control is disabled. 1 = Autoflow control is enabled: • When RTS = 0, only CTS is enabled. • When RTS = 1, RTS and CTS are enabled.
4	LOOP	R/W	0h	Loopback mode enable. LOOP is used for the diagnostic testing using the loopback feature. 0 = Loopback mode is disabled. 1 = Loopback mode is enabled. When LOOP is set, the following occur: • The TX signal is set high. • The RX pin is disconnected • The output of the transmitter shift register (UART_TSR) is looped back in to the receiver shift register (UART_RSR) input.
3	OUT2	R/W	0h	OUT2 Control Bit.
2	OUT1	R/W	0h	OUT1 Control Bit.
1	RTS	R/W	0h	RTS control. When AFE = 1, the RTS bit determines the autoflow control enabled. 0 = RTS is disabled, only CTS is enabled. 1 = RTS and CTS are enabled.

Table 11-3525. UART_MCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	DTR	R/W	0h	Data terminal ready control bit.

Table 11-3526. Register Call Summary for UART_MCR

UART Registers • UART Registers: [0] • UART_MSR Register (Offset = 18h) [reset = 000000-0h]: [0][1][2][3][4][5][6][7] • UART_MCR Register (Offset = 10h) [reset = 0h]: [0][1]
UART Functional Description • Hardware Flow Control: [0][1][2][3] • Reset Considerations: [0][1]
UART Basic Programming Model • UART Module Global Initialization: [0][1]

11.18.6.8 UART_LSR Register (Offset = 14h) [reset = 60h]

UART_LSR is shown in Figure 11-1442 and described in Table 11-3528.

The line status register (UART_LSR) provides information to the CPU concerning the status of data transfers. It is intended for read operations only — do not write to this register. Bits 1 through 4 record the error conditions that produce a receiver line status interrupt.

Table 11-3527. UART_LSR Instances

Instance	Physical Address
UART_0	0253 0C14h
UART_1	0253 1014h
UART_2	0253 1414h

Figure 11-1442. UART_LSR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RXFIFOE	TEMT	THRE	BI	FE	PE	OE	DR
R-0h	R-1h	R-1h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3528. UART_LSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	RXFIFOE	R	0h	Receiver FIFO error. **In non-FIFO mode:** 0 = There has been no error, or RXFIFOE was cleared because the CPU read the erroneous character from the receiver buffer register (UART_RBR). 1 = There is a parity error, framing error, or break indicator in the receiver buffer register (UART_RBR). **In FIFO mode:** 0 = There has been no error, or RXFIFOE was cleared because the CPU read the erroneous character from the receiver FIFO and there are no more errors in the receiver FIFO. 1 = At least one parity error, framing error, or break indicator in the receiver FIFO.

Table 11-3528. UART_LSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	TEMT	R	1h	Transmitter empty (TEMT) indicator. **In non-FIFO mode:** 0 = Either the transmitter holding register (UART_THR) or the transmitter shift register (UART_TSR) contains a data character. 1 = Both the transmitter holding register (UART_THR) and the transmitter shift register (UART_TSR) are empty. **In FIFO mode:** 0 = Either the transmitter FIFO or the transmitter shift register (UART_TSR) contains a data character. 1 = Both the transmitter FIFO and the transmitter shift register (UART_TSR) are empty.
5	THRE	R	1h	Transmitter holding register empty (THRE) indicator. If the THRE bit is set and the corresponding interrupt enable bit is set (ETBEI = 1 in UART_IER), an interrupt request is generated. **In non-FIFO mode:** 0 = Transmitter holding register (UART_THR) is not empty. UART_THR has been loaded by the CPU. 1 = Transmitter holding register (UART_THR) is empty (ready to accept a new character). The content of UART_THR has been transferred to the transmitter shift register (UART_TSR). **In FIFO mode:** 0 = Transmitter FIFO is not empty. At least one character has been written to the transmitter FIFO. The transmitter FIFO may be written to if it is not full. 1 = Transmitter FIFO is empty. The last character in the FIFO has been transferred to the transmitter shift register (UART_TSR).
4	BI	R	0h	Break indicator. The BI bit is set whenever the receive data input (RX) was held low for longer than a fullword transmission time. A full-word transmission time is defined as the total time to transmit the START, data, PARITY, and STOP bits. If the BI bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in UART_IER), an interrupt request is generated. **In non-FIFO mode:** 0 = No break has been detected, or the BI bit was cleared because the CPU read the erroneous character from the receiver buffer register (UART_RBR). 1 = A break has been detected with the character in the receiver buffer register (UART_RBR). **In FIFO mode:** 0 = No break has been detected, or the BI bit was cleared because the CPU read the erroneous character from the receiver FIFO and the next character to be read from the FIFO has no break indicator. 1 = A break has been detected with the character at the top of the receiver FIFO.

Table 11-3528. UART_LSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	FE	R	0h	Framing error (FE) indicator. A framing error occurs when the received character does not have a valid STOP bit. In response to a framing error, the UART sets the FE bit and waits until the signal on the RX pin goes high. When the RX signal goes high, the receiver is ready to detect a new START bit and receive new data. If the FE bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in UART_RBR), an interrupt request is generated. **In non-FIFO mode:** 0 = No framing error has been detected, or the FE bit was cleared because the CPU read the erroneous data from the receiver buffer register (UART_RBR). 1 = A framing error has been detected with the character in the receiver buffer register (UART_RBR). **In FIFO mode:** 0 = No framing error has been detected, or the FE bit was cleared because the CPU read the erroneous data from the receiver FIFO and the next character to be read from the FIFO has no framing error. 1 = A framing error has been detected with the character at the top of the receiver FIFO.
2	PE	R	0h	Parity error (PE) indicator. A parity error occurs when the parity of the received character does not match the parity selected with the EPS bit in the line control register (UART_LCR). If the PE bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in UART_IER), an interrupt request is generated. **In non-FIFO mode:** 0 = No parity error has been detected, or the PE bit was cleared because the CPU read the erroneous data from the receiver buffer register (UART_RBR). 1 = A parity error has been detected with the character in the receiver buffer register (UART_RBR). **In FIFO mode:** 0 = No parity error has been detected, or the PE bit was cleared because the CPU read the erroneous data from the receiver FIFO and the next character to be read from the FIFO has no parity error. 1 = A parity error has been detected with the character at the top of the receiver FIFO.
1	OE	R	0h	Overrun error (OE) indicator. An overrun error in the non-FIFO mode is different from an overrun error in the FIFO mode. If the OE bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in UART_IER), an interrupt request is generated. **In non-FIFO mode:** 0 = No overrun error has been detected, or the OE bit was cleared because the CPU read the content of the line status register (UART_LSR). 1 = Overrun error has been detected. Before the character in the receiver buffer register (UART_RBR) could be read, it was overwritten by the next character arriving in UART_RBR. **In FIFO mode:** 0 = No overrun error has been detected, or the OE bit was cleared because the CPU read the content of the line status register (UART_LSR). 1 = Overrun error has been detected. If data continues to fill the FIFO beyond the trigger level, an overrun error occurs only after the FIFO is full and the next character has been completely received in the shift register. An overrun error is indicated to the CPU as soon as it happens. The new character overwrites the character in the shift register, but it is not transferred to the FIFO.

Table 11-3528. UART_LSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	DR	R	0h	Data-ready (DR) indicator for the receiver. If the DR bit is set and the corresponding interrupt enable bit is set (ERBI = 1 in UART_IER), an interrupt request is generated. **In non-FIFO mode:** 0 = Data is not ready, or the DR bit was cleared because the character was read from the receiver buffer register (UART_RBR). 1 = Data is ready. A complete incoming character has been received and transferred into the receiver buffer register (UART_RBR). **In FIFO mode:** 0 = Data is not ready, or the DR bit was cleared because all of the characters in the receiver FIFO have been read.. 1 = Data is ready. There is at least one unread character in the receiver FIFO. If the FIFO is empty, the DR bit is set as soon as a complete incoming character has been received and transferred into the FIFO. The DR bit remains set until the FIFO is empty again.

Table 11-3529. Register Call Summary for UART_LSR

UART Registers • UART Registers: [0] • UART_LSR Register (Offset = 14h) [reset = 60h]: [0][1][2][3]
UART Functional Description • FIFO Management: [0][1][2][3][4][5][6][7][8][9][10][11] • Interrupt Events and Requests: [0][1][2][3][4][5][6][7] • UART Interrupts: [0]

11.18.6.9 UART_MSR Register (Offset = 18h) [reset = 000000-0h]

UART_MSR is shown in Figure 11-1443 and described in Table 11-3531.

The modem status register (UART_MSR) provides information to the CPU concerning the status of modem control signals. It is intended for read operations only — do not write to this register.

Table 11-3530. UART_MSR Instances

Instance	Physical Address
UART_0	0253 0C18h
UART_1	0253 1018h
UART_2	0253 1418h

Figure 11-1443. UART_MSR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CD	RI	DSR	CTS	DCD	TERI	DDSR	DCTS
R--h	R--h	R--h	R--h	R-0h	R-0h	R-0h	R-0h

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3531. UART_MSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	CD	R	-h	Complement of the Carrier Detect input. When the UART is in the diagnostic test mode (loopback mode UART_MCR[4] = 1), this bit is equal to the UART_MCR bit 3 (OUT2).
6	RI	R	-h	Complement of the Ring Indicator input. When the UART is in the diagnostic test mode (loopback mode UART_MCR[4] = 1), this bit is equal to the UART_MCR bit 2 (OUT1)
5	DSR	R	-h	Complement of the Data Set Ready input. When the UART is in the diagnostic test mode (loopback mode UART_MCR[4] = 1), this bit is equal to the UART_MCR bit 0 (DTR).
4	CTS	R	-h	Complement of the Clear To Send input. When the UART is in the diagnostic test mode (loopback mode UART_MCR[4] = 1), this bit is equal to the UART_MCR bit 1 (RTS).
3	DCD	R	0h	Change in DCD indicator bit. DCD indicates that the DCD input has changed state since the last time it was read by the CPU. When DCD is set and the modem status interrupt is enabled, a modem status interrupt is generated.
2	TERI	R	0h	Trailing edge of RI (TERI) indicator bit. TERI indicates that the RI input has changed from a low to a high. When TERI is set and the modem status interrupt is enabled, a modem status interrupt is generated.
1	DDSR	R	0h	Change in DSR indicator bit. DDSR indicates that the DSR input has changed state since the last time it was read by the CPU. When DDSR is set and the modem status interrupt is enabled, a modem status interrupt is generated.

Table 11-3531. UART_MSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	DCTS	R	0h	Change in CTS indicator bit. DCTS indicates that the CTS input has changed state since the last time it was read by the CPU. When DCTS is set (autoflow control is not enabled and the modem status interrupt is enabled), a modem status interrupt is generated. When autoflow control is enabled, no interrupt is generated.

Table 11-3532. Register Call Summary for UART_MSR

UART Registers • UART Registers: [0] • UART_MSR Register (Offset = 18h) [reset = 000000-0h]: [0][1]
UART Functional Description • Hardware Flow Control: [0]

11.18.6.10 UART_SCR Register (Offset = 1Ch) [reset = 0h]

UART_SCR is shown in Figure 11-1444 and described in Table 11-3534.

The scratch pad register (UART_SCR) is intended for the programmer's use as a scratch pad. It temporarily holds the programmer's data without affecting UART operation.

Table 11-3533. UART_SCR Instances

Instance	Physical Address
UART_0	0253 0C1Ch
UART_1	0253 101Ch
UART_2	0253 141Ch

Figure 11-1444. UART_SCR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SCR							
R-0h																								R-0h							

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3534. UART_SCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	SCR	R	0h	These bits are intended for the programmer's use as a scratch pad in the sense that it temporarily holds the programmer's data without affecting any other UART operation.

Table 11-3535. Register Call Summary for UART_SCR

UART Registers
- UART Registers: [0]
- UART_SCR Register (Offset = 1Ch) [reset = 0h]: [0][1]

11.18.6.11 UART_DLL Register (Offset = 20h) [reset = 0h]

UART_DLL is shown in Figure 11-1445 and described in Table 11-3537.

Two 8-bit register fields (DLL and DLH), called divisor latches, store the 16-bit divisor for generation of the baud clock in the baud generator. The latches are in the UART_DLH and UART_DLL. The UART_DLH holds the most-significant bits of the divisor, and the UART_DLL holds the least-significant bits of the divisor. These divisor latches must be loaded during initialization of the UART to ensure desired operation of the baud generator. Writing to the divisor latches results in two wait states being inserted during the write access while the baud generator is loaded with the new value.

Access considerations:

- The UART_RBR, UART_THR, and UART_DLL share one address. When DLAB = 1 in the UART_LCR, all accesses at the shared address are accesses to the UART_DLL. When DLAB = 0, reading from the shared address gives the content of the UART_RBR, and writing to the shared address modifies the UART_THR.
- The UART_IER and UART_DLH share one address. When DLAB = 1 in the UART_LCR, accesses to the shared address read or modify the UART_DLH. When DLAB = 0, all accesses at the shared address read or modify the UART_IER.

The UART_DLL and UART_DLH also have dedicated addresses. If dedicated addresses are used, the DLAB bit can be kept cleared, so that the UART_RBR, UART_THR, and UART_IER are always selected at the shared addresses.

Table 11-3536. UART_DLL Instances

Instance	Physical Address
UART_0	0253 0C20h
UART_1	0253 1020h
UART_2	0253 1420h

Figure 11-1445. UART_DLL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
DLL							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3537. UART_DLL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	DLL	R/W	0h	The 8 least-significant bits (LSBs) of the 16-bit divisor for generation of the baud clock in the baud rate generator.

Table 11-3538. Register Call Summary for UART_DLL

UART Registers • UART Registers: [0] • UART_RBR Register (Offset = 0h) [reset = 0h]: [0][1][2] • UART_DLL Register (Offset = 20h) [reset = 0h]: [0][1][2][3][4][5] • UART_LCR Register (Offset = Ch) [reset = 0h]: [0][1][2][3] • UART_THR Register (Offset = 0h) [reset = 0h]: [0][1][2]
UART Basic Programming Model • UART Module Global Initialization: [0]

11.18.6.12 UART_DLH Register (Offset = 24h) [reset = 0h]

UART_DLH is shown in Figure 11-1446 and described in Table 11-3540.

Table 11-3539. UART_DLH Instances

Instance	Physical Address
UART_0	0253 0C24h
UART_1	0253 1024h
UART_2	0253 1424h

Figure 11-1446. UART_DLH Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
DLH							
R/W-0h							

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3540. UART_DLH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	DLH	R/W	0h	The 8 most-significant bits (MSBs) of the 16-bit divisor for generation of the baud clock in the baud rate generator.

Table 11-3541. Register Call Summary for UART_DLH

UART Registers • UART Registers: [0] • UART_DLH Register (Offset = 24h) [reset = 0h]: [0] • UART_IER Register (Offset = 4h) [reset = 0h]: [0][1][2] • UART_LCR Register (Offset = Ch) [reset = 0h]: [0][1][2][3][4] • UART_DLL Register (Offset = 20h) [reset = 0h]: [0][1][2][3][4]
UART Basic Programming Model • UART Module Global Initialization: [0]

11.18.6.13 UART_PID Register (Offset = 28h) [reset = 44141102h]

UART_PID is shown in Figure 11-1447 and described in Table 11-3543.

Table 11-3542. UART_PID Instances

Instance	Physical Address
UART_0	0253 0C28h
UART_1	0253 1028h
UART_2	0253 1428h

Figure 11-1447. UART_PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-44141102h																															

LEGEND: R = Read Only; *-n* = value after reset

Table 11-3543. UART_PID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	44141102h	TI internal data. Identifies revision of peripheral.

Table 11-3544. Register Call Summary for UART_PID

UART Registers
- UART Registers: [0]
- UART_PID Register (Offset = 28h) [reset = 4414 1102h]: [0]

11.18.6.14 UART_PWREMU_MGMT Register (Offset = 30h) [reset = 2h]

UART_PWREMU_MGMT is shown in Figure 11-1448 and described in Table 11-3546.

Table 11-3545. UART_PWREMU_MGMT Instances

Instance	Physical Address
UART_0	0253 0C30h
UART_1	0253 1030h
UART_2	0253 1430h

Figure 11-1448. UART_PWREMU_MGMT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
URST	UTRST	URRST	RESERVED				
R/W-0h	R/W-0h	R/W-0h	R-1h				
7	6	5	4	3	2	1	0
RESERVED							FREE
R-1h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3546. UART_PWREMU_MGMT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	URST	R/W	0h	This bit resets and enables both the receiver and transmitter. If this bit is set, the UTRST and URRST are set automatically. 0 = The receiver and transmitter are disabled and in reset state. 1 = The receiver and transmitter are enabled.
14	UTRST	R/W	0h	UART transmitter reset. Resets and enables the transmitter. 0 = Transmitter is disabled and in reset state. 1 = Transmitter is enabled.
13	URRST	R/W	0h	UART receiver reset. Resets and enables the receiver. 0 = Receiver is disabled and in reset state. 1 = Receiver is enabled.
12-1	RESERVED	R	1h	Reserved
0	FREE	R/W	0h	Free-running enable mode bit. This bit determines the emulation mode functionality of the UART. When halted, the UART can handle register read/write requests, but does not generate any transmission/reception, interrupts or events. 0 = If a transmission is not in progress, the UART halts immediately. If a transmission is in progress, the UART halts after completion of the one-word transmission. 1 = Free-running mode is enabled. UART continues to run normally.

Table 11-3547. Register Call Summary for UART_PWREMU_MGMT

UART Registers
- UART Registers: [0]
- UART_PWREMU_MGMT Register (Offset = 30h) [reset = 2h]: [0]

Table 11-3547. Register Call Summary for UART_PWREMU_MGMT (continued)

UART Functional Description • DMA Event Support: [0] • FIFO Management: [0] • Emulation Considerations: [0][1][2][3] • Reset Considerations: [0][1][2][3][4][5][6][7][8] • UART Interrupts: [0]
UART Basic Programming Model • UART Module Global Initialization: [0][1][2]

11.18.6.15 UART_MDR Register (Offset = 34h) [reset = 0h]

UART_MDR is shown in Figure 11-1449 and described in Table 11-3549.

Table 11-3548. UART_MDR Instances

Instance	Physical Address
UART_0	0253 0C34h
UART_1	0253 1034h
UART_2	0253 1434h

Figure 11-1449. UART_MDR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							OSM_SEL
R-0h							R/W-0h

LEGEND: R = Read Only; R/W = Read/Write; *-n* = value after reset

Table 11-3549. UART_MDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	OSM_SEL	R/W	0h	Over-Sampling Mode Select. 0 = 16× oversampling. 1 = 13× oversampling.

Table 11-3550. Register Call Summary for UART_MDR

UART Registers • UART Registers: [0] • UART_MDR Register (Offset = 34h) [reset = 0h]: [0] • UART_RBR Register (Offset = 0h) [reset = 0h]: [0]
UART Functional Description • Clock Configuration: [0]

11.19 Universal Serial Bus Subsystem (USB)

This section describes the USB 2.0 Dual-Role-Device (DRD) subsystem of the device.

NOTE: This chapter serves to describe the integration of the third party USB subsystem and should not be considered sufficient for those wishing to modify the existing Linux or RTOS USB driver(s) or create a new driver to support this controller implementation. For those who do wish to substantially modify the existing Linux or RTOS USB driver(s), or create new drivers, contact your TI sales representative for more information on how to obtain the third party documentation under NDA.

NOTE: The supported set of features and peripherals is device part number dependent. For more information, see the device Data Manual.

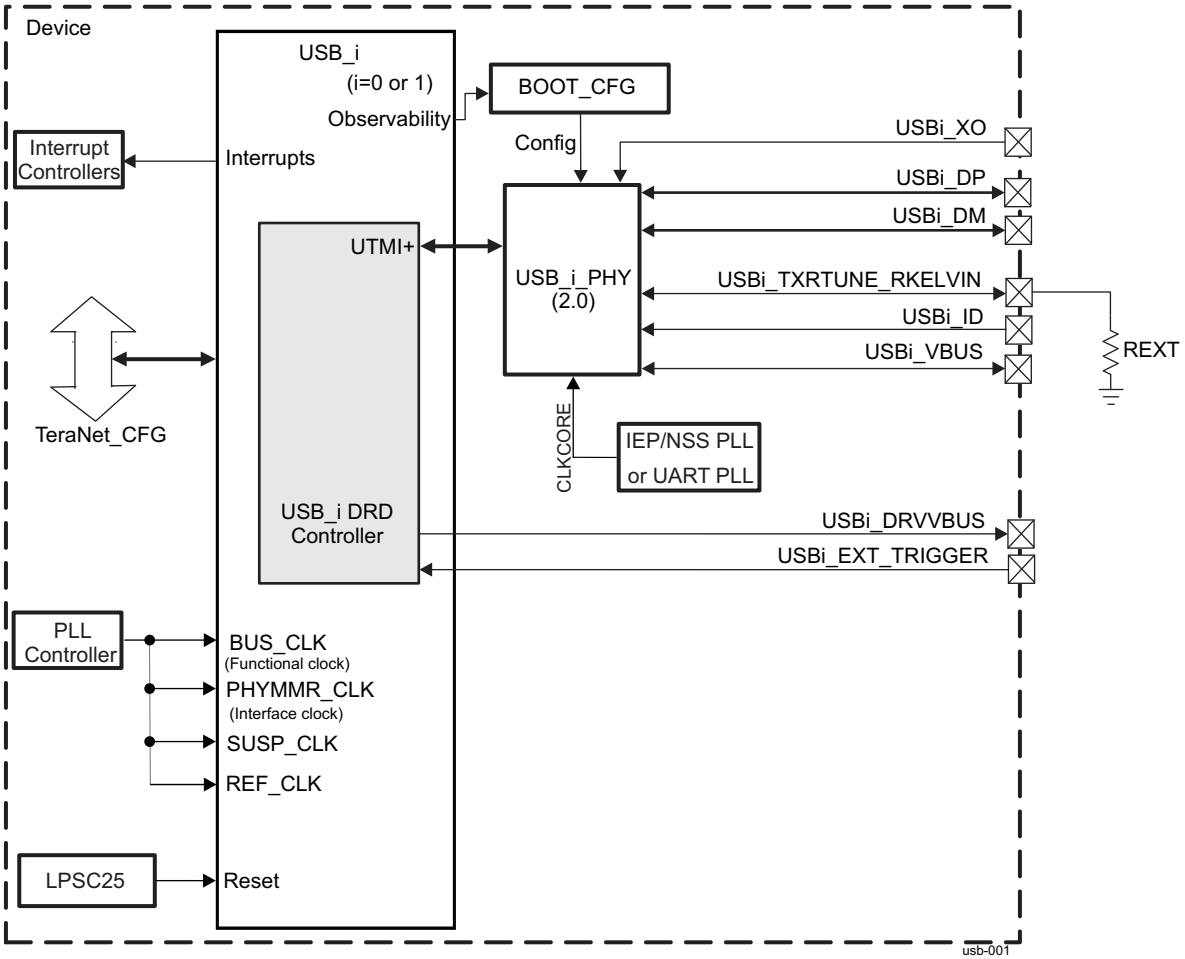
11.19.1 USB Overview

Similar to earlier versions of USB bus, USB 2.0 is a general-purpose cable bus, supporting data exchange between a host device and a wide range of simultaneously accessible peripherals.

The device supports two USB 2.0 subsystems with High Speed Dual-Role-Device (DRD) ports with integrated PHY.

Figure 11-1450 shows the USB module highlights.

Figure 11-1450. USB Overview



11.19.1.1 Terminology

The following acronyms and abbreviations are used in the document.

Term	Definition
A-dev	A USB device with an A-plug connected to its port (ID pin grounded). Default host
ACA	Accessory Charger Adaptor
ADP	Attach Detection Protocol: detects USB OTG attach/detach events.
B-dev	A USB device with a B-plug connected to its port (ID pin floating). Default peripheral.
BER	Bit Error Rate.
CDC	Communication Device Class: USB device class.
DRD	Dual Role Device: USB Host and Peripheral capable.
DS	Down Stream (A USB port facing from a Host or Hub to a Device/Peripheral).
EOP	End of Packet
EP	Endpoint: USB communication channel between the USB link partners carrying a single transfer type (BULK, ISOCH, INT, or CONTROL) and bus sharing arbitration scheme.
FS	Full-Speed USB data rate (12 Mbps).
HCD	Host Controller Driver, designates the USB SW.
HNP	Host Negotiation Protocol, OTG extension to swap USB host and peripheral roles.
HS	High-Speed USB data rate (480 Mbps).
ISOCH	Isochronous transfer type.
ITP	Isochronous Timestamp Packet: USB SS micro-frame boundary packets.
J	Logical USB2 line level: Diff1 in HS/FS, Diff0 in LS. Idle FS/LS bus state.
K	Logical USB2 line level: Diff0 in HS/FS, Diff1 in LS. Resume bus state.
LFPS	Low-Frequency Periodic Signal.
LMP	Link Management Packet.
LS	Low-Speed USB data rate (1.5 Mbps).
MSC	Mass Storage Class.
OTG	On-The-Go extension to USB protocol.
PHY	Physical Layer Device.
PIPE	PHY Interface for PCI Express.
SDR	Single Data Rate.
SE	Single-Ended: USB state based on individual lines' state (D+/D-).
SE0	Single-Ended zero line state where D+=D-=0. Used for reset, FS/LS EOP.
SER	Soft Error Rate.
SOF	Start Of Frame.
SRP	Session Request Protocol. OTG extension to wake-up a system, allowing a B-device to request an A-device to turn on the VBUS power and start session.
SS	Super Speed
TRB	Transfer Request Block.
TX	Transmit.
UAS	USB-Attached SCSI: ANSI standard for USB-attached storage.
ULPI	UTMI+ Low Pin count Interface.
US	Upstream facing from a device (or hub) to the host (or a hub).
USB	Universal Serial Bus.
USB IF	USB Implementers Forum. The governing organization that develops and maintains the USB specifications and compliance standard. www.usb.org
UTMI	USB 2.0 PHY Transceiver Macrocell Interface specification.
UTMI+	UTMI plus extensions to the UTMI spec. Among other improvements, it defines the PHY interface.
xHC	Host Controller, designates the USB HW that implements the xHCI specification.
xHCI	eXtensible Host Controller Interface. The specification that defines the register level interface for host controller.

11.19.1.2 Main Features

The USB 2.0 subsystem, supports the following USB features:

- Dual-role-device (DRD) capability:
  - Supports USB 2.0 Peripheral (or Device) mode at Highspeed (480 Mbps) and Fullspeed (12 Mbps)
  - Supports USB 2.0 Host mode at Highspeed (480 Mbps), Fullspeed (12 Mbps), and Lowspeed (1.5 Mbps)
  - USB 2.0 static peripheral operation
  - USB 2.0 static host operation
  - xHCI Debug Capability
  - External Buffer Control (EBC) mode for IN (Tx) Endpoint
- Each controller instance contains single xHCI with the following features:
  - Compatible to the xHCI specification (revision 1.1) in Host mode
  - Supports 15 Transmit (TX), 15 Receive (RX) endpoints (EPs), and one EP0 endpoint which is bidirectional
  - Internal DMA controller
  - Interrupt moderation and blocking
  - Supports for all USB transfer modes - Control, Bulk, Interrupt, and Isochronous
  - Supports high bandwidth ISO mode
  - Descriptor caching and data pre-fetching used to improve system performance
  - Dynamic FIFO memory allocation for all endpoints
- Operation flexibility:
  - Uniform programming model for HS, FS, and LS operation
  - Multiple interrupt lines:
    - 16 interrupts associated with 16 programmable Event Rings for multi-core support
    - A MISC interrupt line for all miscellaneous events
- ECC RAM
- External requirements:
  - An external charge pump for VBUS 5 V generation
  - An external reference clock input for USB PHY operation
  - An external high-precision resistor for internal PHY termination calibration

The following are USB features which are not supported:

- USB 3.0 SuperSpeed (5Gbps) or USB3.1 SuperSpeed+ (10Gbps) operation in either host or device modes
- OTG Functionality
- HSIC (High Speed inter-chip)
- ULPI Interface for external PHY
- Battery Charger Support
- Accessory Charger Adaptor Support
- xHCI Virtualization
- Hibernation (separate power domain for wake up from USB and save/ restore on wakeup) mode
- External Buffer Control (EBC) for OUT (Rx) Endpoint

11.19.2 USB Environment

11.19.2.1 USB Subsystem I/O Environment

Figure 11-1451 shows the I/O interface signals of the USB subsystem.

Figure 11-1451. USB Subsystem Environment

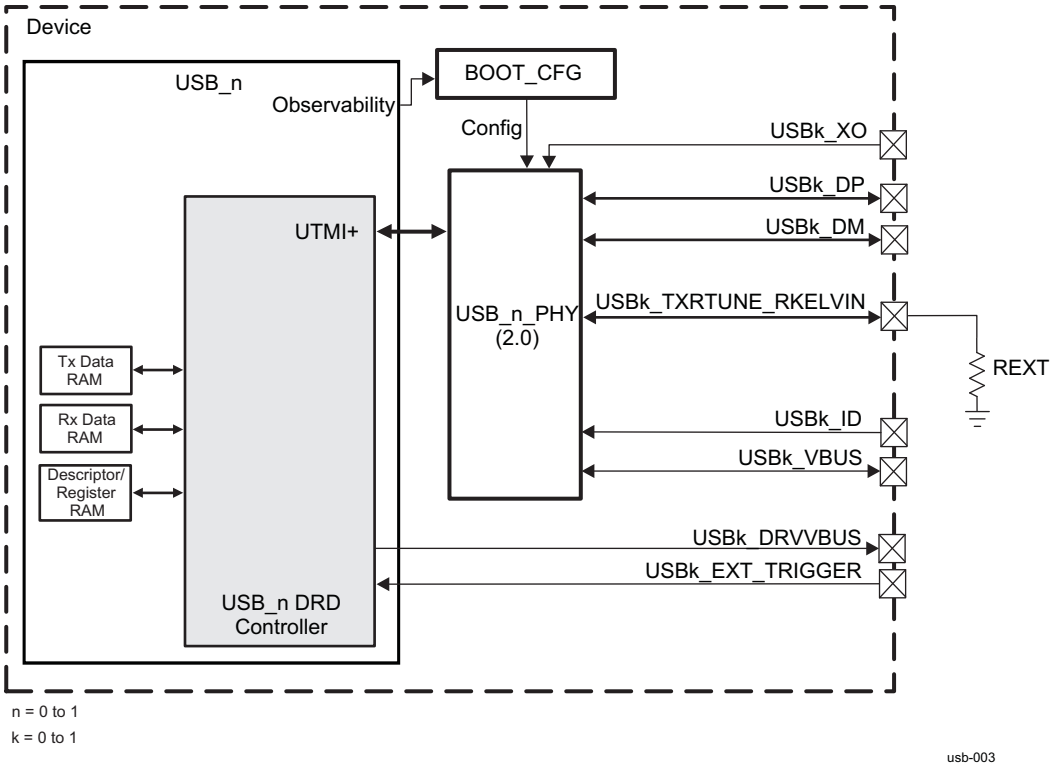


Table 11-3551 describes the external signals of the USB_0 module.

Table 11-3551. USB_0 Input/Output Description

Module Pin	Device Signal	I/O[1]	Description	Value at Reset
USB_0_PHY (2.0)				
USB0XO	USB0_XO	I	Crystal oscillator XO pin or Board Clock Reference Input.	HiZ
USB0DP	USB0_DP	I/O	USB 2.0 and 1.1 specification-compliant signal pins. These are HS/FS/LS bidirectional differential data pins.	HiZ
USB0DM	USB0_DM	I/O		HiZ
USB0ID	USB0_ID	I	USB mini-receptacle identifier. ID Pin with internal pull-up.	A
USBV0BUS	USB0_VBUS	I/O	An analog input for monitoring the voltage on VBUS.	A
USB0RESREF	USB0_TXRTUNE_RKELVIN	I/O	Termination Training Resistor. It is intended to connect an external resistor (REXT) to ground for internal USB 2.0 PHY termination calibration.	A
USB_0 DRD Controller				
USB0DRVVBUS	USB0_DRVVBUS	O	A digital output signal for VBUS Power Supply Enabling. Used to enable an external charge pump to supply +5V power to the VBUS port of the device as well as the VBUS port of the USB receptacle, when appropriate per the USB standard.	0
USB0EXTTRIGGER	USB0_EXT_TRIGGER	I	External trigger for tracing data.	0

(1) I = Input; O = Output

Table 11-3552 describes the external signal of the USB_1 module.

Table 11-3552. USB_1 Input/Output Description

Module Pin	Device Signal	I/O[1]	Description	Value at Reset
USB_1_PHY (2.0)				
USB1XO	USB1_XO	I	Crystal oscillator XO pin or Board Clock Reference Input.	HiZ
USB1DP	USB1_DP	I/O	USB 2.0 and 1.1 specification-compliant signal pins. These are HS/FS/LS bidirectional differential data pins.	HiZ
USB1DM	USB1_DM	I/O		HiZ
USB1ID	USB1_ID	I	USB mini-receptacle identifier. ID Pin with internal pull-up.	A
USBV1BUS	USB1_VBUS	I/O	An analog input for monitoring the voltage on VBUS.	A
USB1RESREF	USB1_TXRTUNE_RKELVIN	I/O	Termination Training Resistor. It is intended to connect an external resistor (REXT) to ground for internal USB 2.0 PHY termination calibration.	A
USB_1 DRD Controller				
USB1DRVVBUS	USB1_DRVVBUS	O	A digital output signal for VBUS Power Supply Enabling. Used to enable an external charge pump to supply +5V power to the VBUS port of the device as well as the VBUS port of the USB receptacle, when appropriate per the USB standard.	0
USB1EXTTRIGGER	USB1_EXT_TRIGGER	I	External trigger for tracing data.	0

(1) I = Input; O = Output

11.19.3 USB2.0 Subsystem Application

As the USB controller modules present in this device are DRD (Dual-Role Devices), they can support operation as either a USB Host or a USB Device. The operational mode determination is made based on the state of the USBn_ID pin; when this pin is grounded, the controller will operate as a USB Host, when this pin is left floating, the controller assumes the role of a USB Device. For implementations that do not require DRD functionality, the USBn_ID pin can either be left floating or can be grounded in the board design depending on the static role required. For implementations that do require DRD functionality, the USBn_ID pin should be connected directly to the corresponding ID pin on a USB Micro-AB socket. In doing so, the USBn_ID pin will be correctly terminated (open or grounded) depending on the cable attached and the controller will enter Host or Device mode accordingly.

Figure 11-1452 shows typical application of the USB_0 controller in the USB2.0 DRD (dual-role-device) mode. The power switch drives 5-V VBUS when required by monitoring the USB0_DRV_VBUS output from the USB controller.

Figure 11-1452. USB_0 Controller Application: USB2.0 DRD

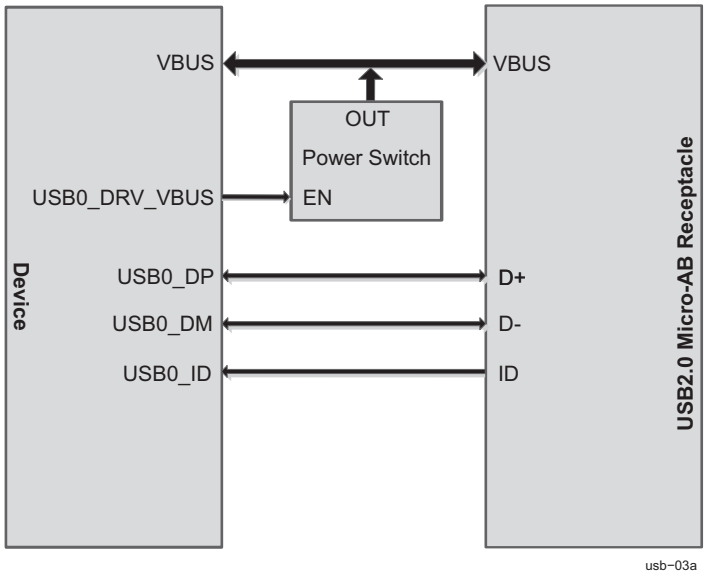


Figure 11-1453 shows typical application of the USB_0 controller in USB2.0 Host mode. The power switch drives 5-V VBUS when required by monitoring the USB0_DRV_VBUS output from the USB controller.

Figure 11-1453. USB_0 Controller Application: USB2.0 Host

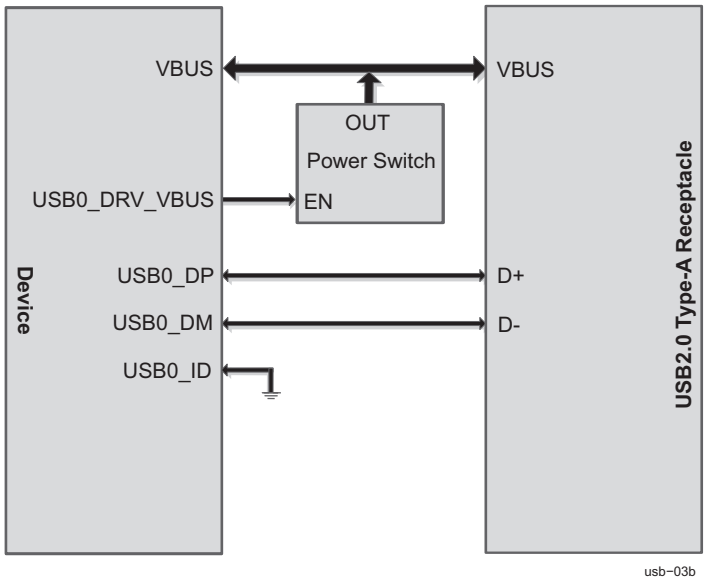


Figure 11-1454 shows typical application of the USB_0 controller in USB2.0 Device mode.

Figure 11-1454. USB_0 Controller Application: USB2.0 Device

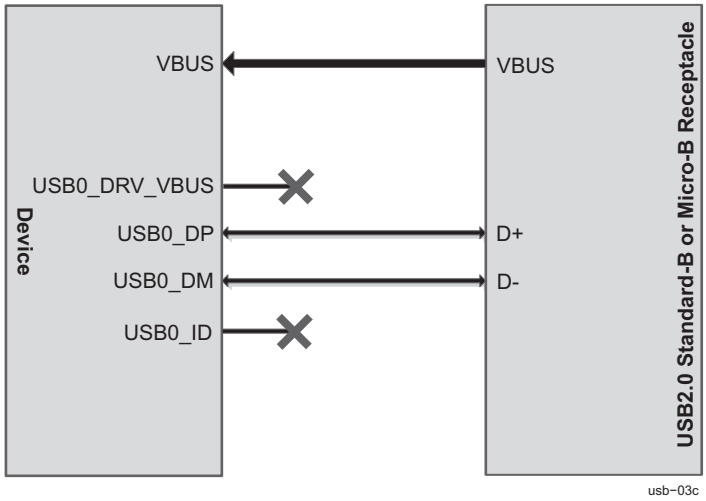


Figure 11-1455 shows typical application of the USB_1 controller in the USB2.0 DRD (dual-role-device) mode. The power switch drives 5-V VBUS when required by monitoring the USB1_DRV_VBUS output from the USB controller.

Figure 11-1455. USB_1 Controller Application: USB2.0 DRD

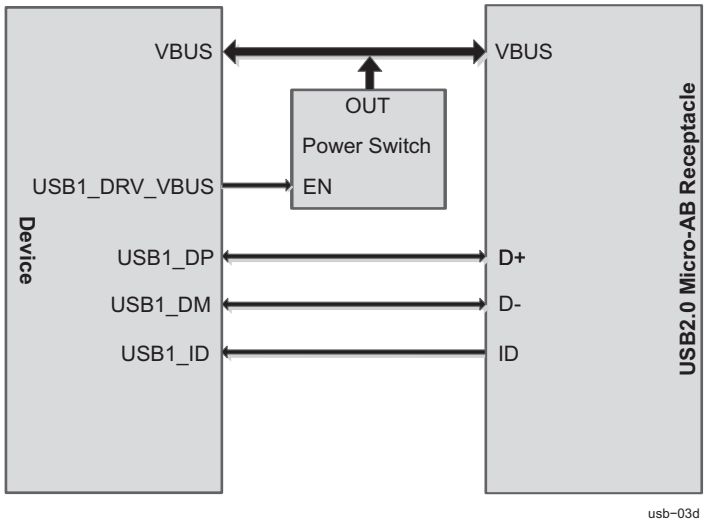


Figure 11-1456 shows typical application of the USB_1 controller in USB2.0 Host mode. The power switch drives 5-V VBUS when required by monitoring the USB1_DRV_VBUS output from the USB controller.

Figure 11-1456. USB_1 Controller Application: USB2.0 Host

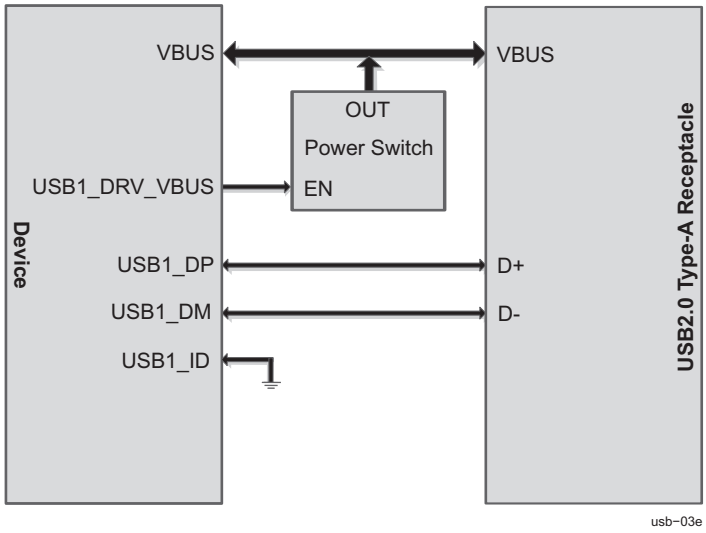
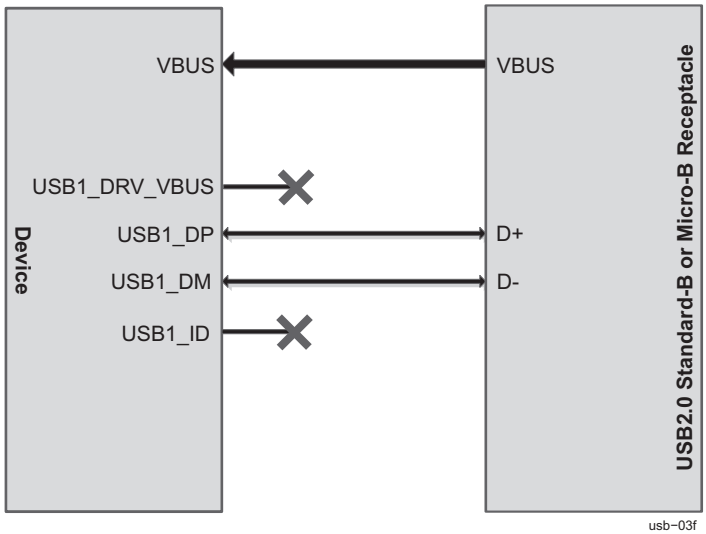


Figure 11-1457 shows typical application of the USB_1 controller in USB2.0 device mode.

Figure 11-1457. USB_1 Controller Application: USB2.0 Device



11.19.4 USB Integration

This section describes USB2.0 module integration in the device, including information about clocks, resets, and hardware requests.

NOTE: USB 2.0 OTG is not supported for this device.

Figure 11-1458 shows the integration of the USB_0 module in the device.

Figure 11-1458. USB_0 Integration

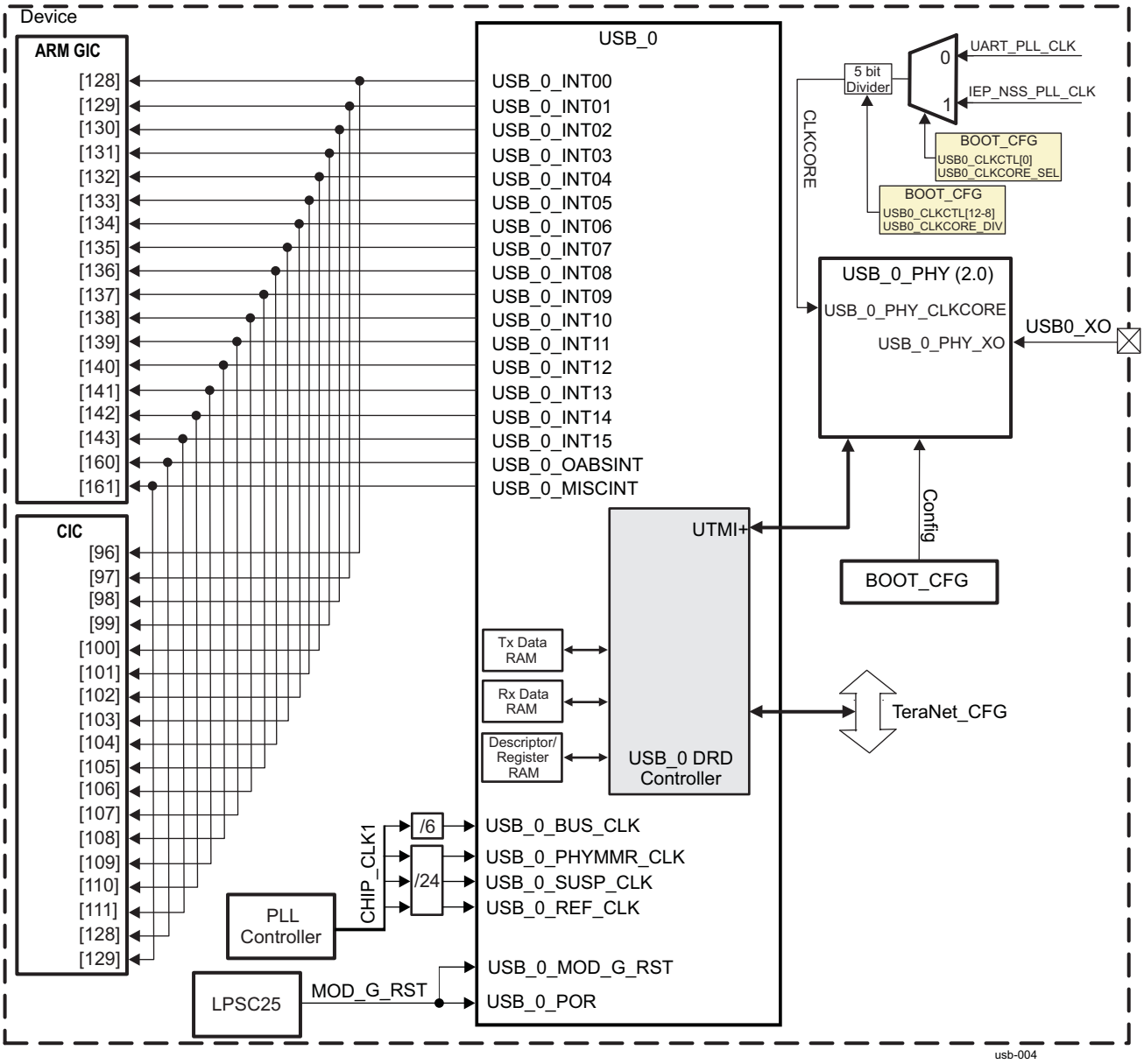


Figure 11-1459 shows the integration of the USB_1 module in the device.

Figure 11-1459. USB_1 Integration

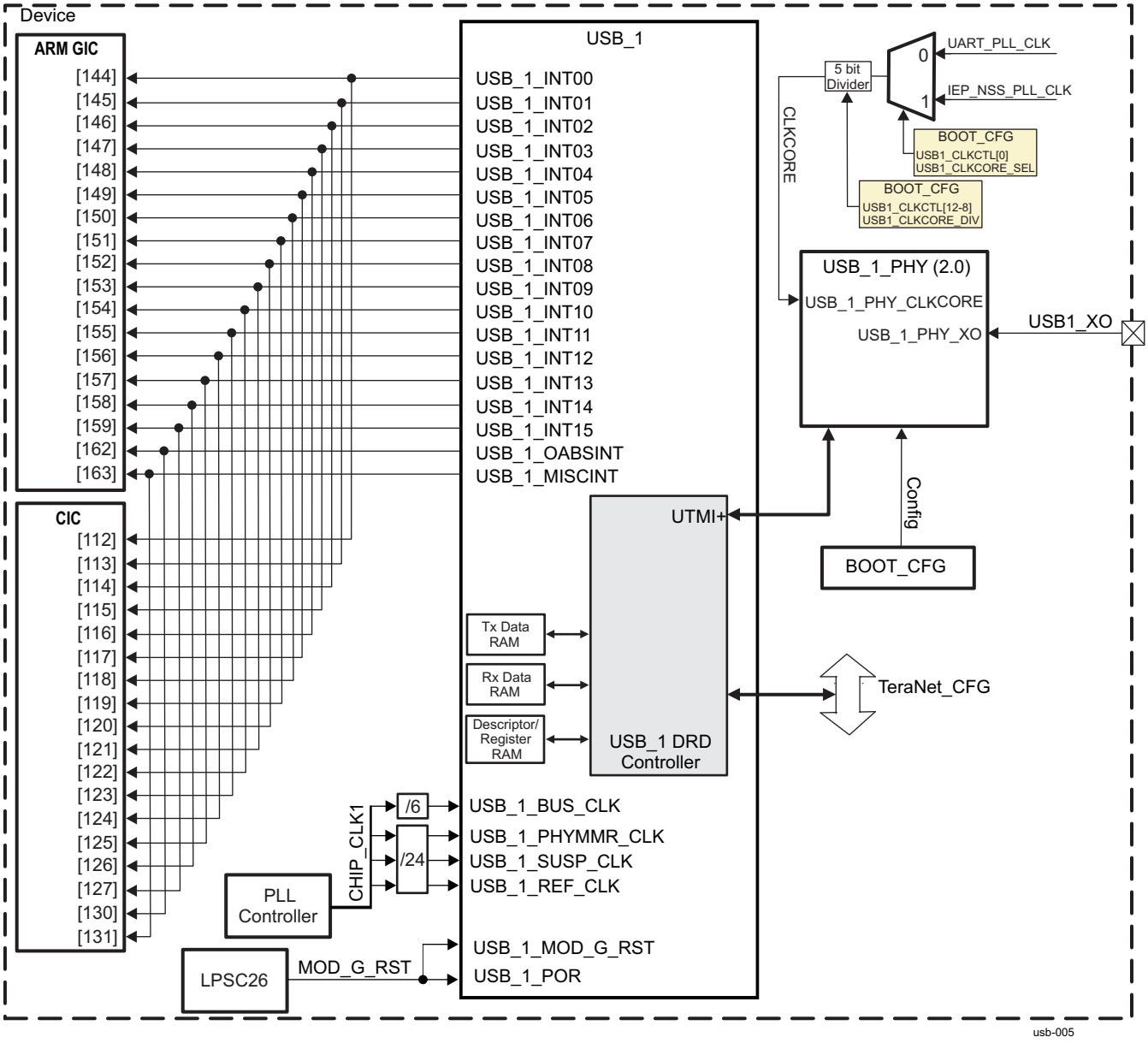


Table 11-3553 through summarize the integration of the USB modules in the device.

Table 11-3553. USB Integration Attributes

Module Instance	Attributes		
	Power Domain	Module Domain	Interconnect
USB_0	PD14	LPSC25	TeraNet_CFG
USB_1	PD14	LPSC26	TeraNet_CFG

Table 11-3554. USB Clocks and Resets

Clocks				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
USB_0	USB_0_BUS_CLK	CHIP_CLK1 / 6	PLL Controller	The main BUS_CLK clock to the wrapper module and to the DRD controller module.
	USB_0_PHYMMR_CLK	CHIP_CLK1 / 24	PLL Controller	The PHYMMR_CLK clock is used to access the PHY registers. The Slave Bus-1 interface to PHY registers is completely asynchronous.
	USB_0_SUSP_CLK	CHIP_CLK1 / 24	PLL Controller	The USB_0_SUSP_CLK drives a portion of DRD controller.
	USB_0_REF_CLK	CHIP_CLK1 / 24	PLL Controller	The REF_CLK clock is used to a portion of DRD controller when USB_0 PHY is in the suspended state.
	USB_0_PHY_CLKCORE	CLKCORE	Divided version of UART PLL or IEP / NSS PLL output clocks	On-chip reference clock source for USB_0 PHY.
	USB_0_PHY_XO	USB0XO_REF_CLK	External USB0_XO input pin	Optional USB_0 PHY reference clock. External clock connected to the USB0_XO pin.
USB_1	USB_1_BUS_CLK	CHIP_CLK1 / 6	PLL Controller	The main BUS_CLK clock to the wrapper module and to the DRD controller module.
	USB_1_PHYMMR_CLK	CHIP_CLK1 / 24	PLL Controller	The PHYMMR_CLK clock is used to access the PHY registers. The Slave Bus-1 interface to PHY registers is completely asynchronous.
	USB_1_SUSP_CLK	CHIP_CLK1 / 24	PLL Controller	The USB_1_SUSP_CLK drives a portion of DRD controller. The ADP (Attach Detection Protocol) logic use also the Suspend clock (SUSP_CLK).
	USB_1_REF_CLK	CHIP_CLK1 / 24	PLL Controller	The REF_CLK clock is used to a portion of DRD controller when USB_1 PHY is in the suspended state.
	USB_1_PHY_CLKCORE	CLKCORE	Divided version of UART PLL or IEP / NSS PLL output clocks	On-chip reference clock source for USB_1 PHY.
	USB_1_PHY_XO	USB1XO_REF_CLK	External USB1_XO input pin	Optional USB_1 PHY reference clock. External clock connected to the USB1_XO pin.
Resets				
Module Instance	**Destination Signal**	**Source Signal**	**Source**	**Description**
USB_0	USB_0_MOD_G_RST	MOD_G_RST	LPSC25	Module Asynchronous Reset. It will reset all regions, except the RAMs within the xHC controller. This signal resets the USB_0 PHY transmit and receive logic.
	USB_0_POR	MOD_G_RST	LPSC25	Module level power on reset (PHY Reset). This signal resets all state machines and registers inside the USB_0 PHY.

Table 11-3554. USB Clocks and Resets (continued)

USB_1	USB_1_MOD_G_RST	MOD_G_RST	LPSC26	Module Asynchronous Reset. It will reset all regions, except the RAMs within the xHC controller. This signal resets the USB_1 PHY transmit and receive logic.
	USB_1_POR	MOD_G_RST	LPSC26	Module level power on reset (PHY Reset). This signal resets all state machines and registers inside the USB_1 PHY.

Table 11-3555. USB Hardware Requests

Interrupt Requests				
Module Instance	**Event Name**	**Mapped To Input Event [Number]**		**Description**
		ARM GIC	**CIC**	
USB_0	USB_0_INT00	[128]	[96]	USB_0 Event ring 0 interrupt
	USB_0_INT01	[129]	[97]	USB_0 Event ring 1 interrupt
	USB_0_INT02	[130]	[98]	USB_0 Event ring 2 interrupt
	USB_0_INT03	[131]	[99]	USB_0 Event ring 3 interrupt
	USB_0_INT04	[132]	[100]	USB_0 Event ring 4 interrupt
	USB_0_INT05	[133]	[101]	USB_0 Event ring 5 interrupt
	USB_0_INT06	[134]	[102]	USB_0 Event ring 6 interrupt
	USB_0_INT07	[135]	[103]	USB_0 Event ring 7 interrupt
	USB_0_INT08	[136]	[104]	USB_0 Event ring 8 interrupt
	USB_0_INT09	[137]	[105]	USB_0 Event ring 9 interrupt
	USB_0_INT10	[138]	[106]	USB_0 Event ring 10 interrupt
	USB_0_INT11	[139]	[107]	USB_0 Event ring 11 interrupt
	USB_0_INT12	[140]	[108]	USB_0 Event ring 12 interrupt
	USB_0_INT13	[141]	[109]	USB_0 Event ring 13 interrupt
	USB_0_INT14	[142]	[110]	USB_0 Event ring 14 interrupt
	USB_0_INT15	[143]	[111]	USB_0 Event ring 15 interrupt
	USB_0_OABSINT	[160]	[128]	USB_0 OTG/ ADP/ BC/ SER interrupt
	USB_0_MISCINT	[161]	[129]	USB_0 Miscellaneous interrupt

Table 11-3555. USB Hardware Requests (continued)

USB_1	USB_1_INT00	[144]	[112]	USB_1 Event ring 0 interrupt
	USB_1_INT01	[145]	[113]	USB_1 Event ring 1 interrupt
	USB_1_INT02	[146]	[114]	USB_1 Event ring 2 interrupt
	USB_1_INT03	[147]	[115]	USB_1 Event ring 3 interrupt
	USB_1_INT04	[148]	[116]	USB_1 Event ring 4 interrupt
	USB_1_INT05	[149]	[117]	USB_1 Event ring 5 interrupt
	USB_1_INT06	[150]	[118]	USB_1 Event ring 6 interrupt
	USB_1_INT07	[151]	[119]	USB_1 Event ring 7 interrupt
	USB_1_INT08	[152]	[120]	USB_1 Event ring 8 interrupt
	USB_1_INT09	[153]	[121]	USB_1 Event ring 9 interrupt
	USB_1_INT10	[154]	[122]	USB_1 Event ring 10 interrupt
	USB_1_INT11	[155]	[123]	USB_1 Event ring 11 interrupt
	USB_1_INT12	[156]	[124]	USB_1 Event ring 12 interrupt
	USB_1_INT13	[157]	[125]	USB_1 Event ring 13 interrupt
	USB_1_INT14	[158]	[126]	USB_1 Event ring 14 interrupt
	USB_1_INT15	[159]	[127]	USB_1 Event ring 15 interrupt
	USB_1_OABSINT	[162]	[130]	USB_1 OTG/ ADP/ BC/ SER interrupt
	USB_1_MISCINT	[163]	[131]	USB_1 Miscellaneous interrupt

11.19.5 USB Registers

NOTE: Details on the registers found in the USB subsystem are not provided in this document. The existing Linux or RTOS drivers should be used for proper USB operation. For those who do wish to substantially modify the existing Linux or RTOS USB driver(s), or create new drivers, contact your TI sales representative for more information on how to obtain the third party documentation under NDA.

On-chip Debug

This chapter describes the debug and trace features of the device.

Topic **Page**

12.1 Introduction to SoC Debug Architecture

12.1.1 Overview

This chapter describes the capabilities of the various features available through the debug architecture on the device.

The SoC debug capabilities are centered around the Debug Subsystem (DEBUGSS) module. The DEBUGSS module contains the ICEPick module which handles the JTAG TAP and multiple secondary TAPs for the various processing cores of the device. It also provides: DAP port for system wide memory access from debugger, cross-triggering, system trace, peripheral suspend generation, debug port (EMUx) pin management, etc.

The DEBUGSS module works in conjunction with the debug capability integrated in the processing cores (Arm Cortex®-A15, C66x DSP, etc.) to provide a comprehensive hardware platform for a rich debug and development experience.

In addition, the device provides the following debug and trace capabilities to meet the requirement to optimize for power efficiency:

- Ability to turn off the power and clock of the DEBUGSS
- Debug and instrumentation via Instruction Trace Macrocell (ITM) in PMMC

12.1.2 On-chip Debug Features

The SoC implements the following debug features:

- Debug interface
  - Support for 1149.1 standard (JTAG + boundary scan) and 1149.6 (boundary scan extensions). No support for 1149.7 (cJTAG)
  - Supports 20 EMU pins (EMU[19:0]) for exporting system and processor trace data. Some pins may be muxed with other functional pins and thus not always available in all system configurations
  - Supports cross-triggering between devices, and debug boot mode control via EMU[1:0] pins
- ARMSS debug and trace
  - Support for invasive debug like halt mode debugging (breakpoint, watchpoints) and monitor mode debugging
  - Support for non-invasive debugging (program trace, performance monitoring)
  - Support for Cortex-A15 Performance Monitoring Unit (cycle counters)
  - Support for CoreSight Program Trace Module (CS-PTM) with timing
  - Support for an integrated CoreSight System Trace Module (CS-STM) for hardware event and software instrumentation
  - A shared timestamp counter for the Cortex-A15 core and CS-STM is integrated in ARMSS for trace data correlation
  - Support for a 16KB Trace Buffer and Router (TBR) to hold PTM/STM trace. The trace data can be copied by EDMA to external memory for draining by SoC high-speed serial interfaces
  - Support for simultaneous draining of trace stream through EMUx pins and TBR (to achieve higher aggregate trace throughput)
  - Support for debug authentication interface to disable debug accesses in secure devices
  - Support for cross-triggering between Arm core, CS-STM and CT-TBR
  - Support for debug through warm reset
- DSP debug and trace
  - Support for halt mode debug
  - Support for real-time debug
  - Support for monitor mode debug
  - Advanced Event Triggering (AET) for data/PC watchpoints, event monitoring and visibility into external events

- Support for PC/Timing/Data/Event trace
- 4KB TETB (TI Embedded Trace Buffer) for storing PC/Timing/Data/Event trace. The trace data can be copied by EDMA to external memory for draining by SoC high-speed serial interfaces or it can be drained through EMU[19:0] pins
- Support for cross-triggering source/sink to other SoC subsystems

- SoC-level debug and trace:
  - Support debug capability for following processors and hardware accelerators
    - All PRUs in PRU-ICSS modules
    - Cortex-M3 processor in the PMMC, support for ITM trace
  - Support limited debug capabilities (not via CCS) of Hardware Accelerators (HWAs), including:
    - DSS_UL
    - Crypto in the NSS
  - Support emulation mode aware peripherals (suspend features and debug access features)
  - Power domain requirement for DEBUGSS
    - The DEBUGSS resides in its own power domain
    - Domain is ON by default, can be switched OFF by application
    - Domain will automatically wake up upon connection of debugger
    - Automatic turn off when external debugger is disconnected
  - Debug activities will not be interrupted by events of clock-gate or power-down of an individual subsystem
  - Multicore debug
    - Two cross-triggering channels (Trigger0, and Trigger1), shared by:
      - Processor subsystems – ARMSS, DSP. Note that PMMC and PRU-ICSS do not support cross-triggering
      - External device (via EMU[1:0] pins)
      - ARMSS STM, Tracers, CT-TBRs and DEBUGSS STM
    - Synchronized (global) run of C66x DSP, Arm Cortex-A15, PMMC Cortex-M3, and all HWAs
    - Global stop of C66x DSP, Arm Cortex-A15, PMMC Cortex-M3, PRU-ICSS PRUs, and all HWAs
  - Debug access to system resources
    - Support system memory access via the DAP port (natively support 32-bit address, and it can support 36-bit address through configuration of MPAX inside MSMC)
    - Debug access to any invalid memory location (reserved/clock-gated/power-down) will not cause system hang
  - Debug activities does not affect the correctness of an application
    - Peripheral, which is sensitive to a debug access, distinguishes access initiated from debugger versus from application
    - Peripheral, which is sensitive to a debug event (CPU halt), behaves properly during processor suspension
    - Errors triggered by a debug access does not cause any undesired interrupt or exception to the normal execution of a processor
  - Support for debug related reset features, including:
    - Debugger-generated system reset and subsystem resets via ICEPick-D
    - Debugger-generated local resets to individual processor (DSP Subsystem can generate a local reset)
    - Blocking certain system reset generated by applications
    - Blocking subsystem local resets generated by applications
    - Debug logics will survive all resets except global cold resets or TRSTz/TLR
    - TRSTz/TLR resets only affect debug and test logics. It does not affect any functional operation

of the device
- Wait-In-Reset (WIR) debug boot mode

– System trace: support STM-based system trace as following:
  - Software messages generated by application code running on each processor cores
  - Hardware messages generated by Tracers for logging of bus transactions for selected end-points
  - Support 32KB DEBUGSS TBR (Trace Buffer and Router) to hold system trace
  - Support for trace export (from all processor cores and DEBUGSS STM) through Debug Port pins (EMU[19:0])
  - TBR can be drained to on-chip DDR via EDMA
– Debug will be able to get DSP core back to ready after a DSP hang occurs. Note this capability only applies to the DSP core.

12.1.3 Application Integration

TI provides a set of libraries known as CToolsLib that allows for easy integration of debug and trace capabilities into existing software. For more information about CToolsLib, see Section 12.7.

12.2 SoC Debug Interfaces

12.2.1 IEEE1149.1 JTAG Interface

The SoC supports a standard 5-pin JTAG (IEEE1149.1) interface for emulator-based debug. This interface controls the ICEPick module (generic TAP for emulation) to allow the debugger to access several debug resources through its secondary (output) JTAG ports.

Table 12-1 describes the 5-pin JTAG interface.

Table 12-1. JTAG Interface Signals

Pin Name	I/O Type	Internal Pull Type	Description
TRSTz	I	PD	*Test Reset* input. Iinitializes and disables the test interface
TCK	I	PU	*Test Clock* input. Controls the timing of the test interface independently from any system clocks. TCK is pulsed by the equipment controlling the test and not by the tested device.
TMS	I	PU	*Test Mode Select* input. Controls the transitions of the test interface state machine.
TDI	I	PU	*Test Data Input*. Supplies the data to the JTAG registers (Boundary Scan Register, Instruction Register, or other data registers).
TDO	O	PU	*Test Data Output*. Used to serially output the data from the JTAG registers to the equipment controlling the test.

NOTE: The state of all JTAG pins and trace (EMU) pins shall not be affected by the chip-level functional reset. They would only be affected by TRSTz or PORn.

For maximum reliability, the device includes an internal pull-down resistor on the TRSTz pin to ensure that TRSTz will always be asserted upon power-up and the device internal emulation logic will always be properly initialized when this pin is not routed out. JTAG controllers from Texas Instruments actively drive TRSTz high. However, some third-party JTAG controllers may not drive TRSTz high, but expect the use of an external pull-up resistor on TRSTz. When using this type of JTAG controller, assert TRSTz to initialize the device after power-up and externally drive TRSTz high before attempting any emulation or boundary scan operations.

12.2.2 ICEPick Module

12.2.2.1 ICEPick Overview

The debugger is connected to the SoC through its external JTAG interface. The first level of debug interface seen by the debugger is connected to the ICEPick (version D) module embedded in the DEBUGSS. ICEPick is the chip-level TAP, responsible for providing access to the IEEE1149.1 and IEEE1149.6 boundary scan capabilities of the device.

The SoC has multiple processors and debug modules, some with secondary JTAG TAPs (e.g., DSP) and others with an APB memory-mapped interface (e.g., Arm subsystem and CoreSight components). ICEPick manages the TAPs as well as the power/reset/clock controls for the logics associated with the TAPs and the APB ports.

The ICEPick module is visible only from the debugger point of view and thus cannot be programmed by application software. The debugger can configure ICEPick through its own TAP controller. The ICEPick TAP has an instruction length of 6 bits and is the primary TAP. It is always visible in the scan chain and is used to control and monitor the other (secondary) TAPs.

ICEPick provides the following debug capabilities:

- Debug connect logic for enabling or disabling most ICEPick instructions
- Dynamic TAP insertion
  - Serially linking up to 32 TAP controllers
  - Individually selecting one or more of the TAPs for scan without disrupting the instruction register (IR) state of other TAPs

- Power, reset and clock management
  - Provides the power and clock status of the domain to the debugger
  - Provides debugger control of the power domain of a processor
    - Force the domain power and clocks on
    - Prohibit the domain from being clock-gated or powered-down
  - Applies system reset
  - Provides wait-in-reset (WIR) boot mode as described in Section 12.6.7
  - Provides global and local WIR release
  - Provides global and local reset block

The ICEPick module implements a connect register, which must be configured with a predefined key to enable the full set of JTAG instructions. Once the debug connect key has been properly programmed, ICEPick signals and subsystems emulation logic should be turned on.

12.2.2.2 ICEPick Dynamic Tap Insertion

To include more or fewer secondary TAPs in the scan chain, the debugger must use the ICEPick TAP router to program the TAPs. At its root, ICEPick is a scan-path linker that lets the debugger selectively choose which subsystem TAPs are accessible through the device-level debug interface. Each secondary TAP can be dynamically included in or excluded from the scan path. From external JTAG interface point of view, secondary TAPs that are not selected appear not to exist.

There are two types of components connected through ICEPick to external debug interface:

- Legacy JTAG Components
  - DSPSS (single C66x core) implements a JTAG-compatible port and is directly interfaced with ICEPick and individually attached to an ICEPick secondary TAP
- CoreSight Components
  - The CoreSight components are interfaced with ICEPick through the CS_DAP module. The CS_DAP is attached to an ICEPick secondary TAP and translates JTAG transactions into APBv3 transactions.

Table 12-2 shows the ICEPick secondary TAPs in the system.

Table 12-2. ICEPick Debug Secondary TAPs

Tap #	Type	Name	IR Scan Length	Description
0	N/A	N/A	N/A	Reserved. This is an internal TAP and not exposed at the DEBUGSS boundary
1	JTAG	DSP/C66x	38	Access to DSP
2-7	JTAG	Reserved	N/A	Spare ports
8	JTAG	PMMC/Cortex-M3	4	Access to PMMC[1]
9-13	JTAG	Reserved	N/A	Spare ports
14	CS	CS_DAP (APB-AP)	4	APB-AP is used to access Arm A15 Core and other APB components
		CS_DAP (AHB-AP)		AHB-AP is used to access system memory map.

(1) PMMC/Cortex-M3 is not directly interfaced with ICEPick. There is a DAP instantiated to connect the JTAG pins from ICEPick to the AHB-AP port inside Cortex-M3.

Besides secondary debug TAPs, ICEPick also supports power/reset/clock controls for non-JTAG debug cores. These debug cores are accessible through the CS_DAP.

Table 12-3 summarizes the ICEPick debug core mapping.

Table 12-3. ICEPick Debug Cores

Debug Core #	Name
0	Arm Cortex-A15 core
1	Reserved
2	Reserved
3	Reserved
4	PRU0 in PRU-ICSS_0
5	PRU1 in PRU-ICSS_0
6	PRU0 in PRU-ICSS_1
7	PRU1 in PRU-ICSS_1
8-23	Reserved

12.2.3 Debug Port

The SoC supports a 20-pin emulation dedicated interface – EMU[19:0]. The EMU[19:0] pins are collectively called the "Debug Port" and are shared by the following functions:

- Trace
  - ARMSS/Cortex-A15 trace (via PTM)
  - PMMC/Cortex-M3 trace (via ITM)
  - DSP/C66x trace
  - STM trace
- Cross-triggering
- Debug boot modes

12.2.3.1 Debug Port Configuration

Table 12-4 shows the Debug Port configuration. The application software must make sure to program the Debug Resource Manager (DRM) integrated in the Debug Subsystem to appropriately export trace data to an external trace receiver. The Debug Pin Manager (DPM) of the DRM specifies this configuration by matching each of the EMU pins listed below to a specific function such as: DSP trace, STM trace, cross-triggering, etc.

Table 12-4. Debug Port Configuration

EMU Pins	Cross-Triggering	PTM/ITM Trace (via CS-TPIU)		DSP Trace		STM	Debug Boot Mode
EMU19		TRCDTa[15]	TRCDTb[17]	TRCDTa[15]	TRCDTb[17]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU18		TRCDTa[14]	TRCDTb[16]	TRCDTa[14]	TRCDTb[16]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU17		TRCDTa[13]	TRCDTb[15]	TRCDTa[13]	TRCDTb[15]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU16		TRCDTa[12]	TRCDTb[14]	TRCDTa[12]	TRCDTb[14]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU15		TRCDTa[11]	TRCDTb[13]	TRCDTa[11]	TRCDTb[13]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU14		TRCDTa[10]	TRCDTb[12]	TRCDTa[10]	TRCDTb[12]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	

Table 12-4. Debug Port Configuration (continued)

EMU Pins	Cross-Triggering	PTM/ITM Trace (via CS-TPIU)		DSP Trace		STM	Debug Boot Mode
EMU13		TRCDTa[9]	TRCDTb[11]	TRCDTa[9]	TRCDTb[11]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU12		TRCDTa[8]	TRCDTb[10]	TRCDTa[8]	TRCDTb[10]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU11		TRCDTa[7]	TRCDTb[9]	TRCDTa[7]	TRCDTb[9]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU10		TRCDTa[6]	TRCDTb[8]	TRCDTa[6]	TRCDTb[8]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU9		TRCDTa[5]	TRCDTb[7]	TRCDTa[5]	TRCDTb[7]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU8		TRCDTa[4]	TRCDTb[6]	TRCDTa[4]	TRCDTb[6]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU7		TRCDTa[3]	TRCDTb[5]	TRCDTa[3]	TRCDTb[5]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU6		TRCDTa[2]	TRCDTb[4]	TRCDTa[2]	TRCDTb[4]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU5		TRCDTa[1]	TRCDTb[3]	TRCDTa[1]	TRCDTb[3]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU4		TRCDTa[0]	TRCDTb[2]	TRCDTa[0]	TRCDTb[2]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU3		TRCCLKB	TRCCLKB	TRCCLKB	TRCCLKB	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU2		TRCCLKA	TRCCLKA	TRCCLKA	TRCCLKA	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	
EMU1	Trigger1		TRCDTb[1]		TRCDTb[1]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	Debug boot mode [1]
EMU0	Trigger0		TRCDTb[0]		TRCDTb[0]	TRCDT3, or TRCDT2, or TRCDT1, or TRCDT0, or TRCCLK or Tri-state	Debug boot mode [0]

NOTE: If the EMU[1:0] signals are shared for cross-triggering purposes at board level, they SHOULD NOT be used for trace purposes.

12.2.3.2 Concurrent Use of Debug Port

The following combinations are possible concurrently:

- Trigger 0/1
- Trigger 0/1 and STM Trace (up to 4 data pins)
- Trigger 0/1 and STM Trace (up to 4 data pins) and DSP Trace (up to 11 data pins)
- Trigger 0/1 and STM Trace (up to 4 data pins) and TPIU Trace (up to 11 data pins)
- STM Trace (up to 4 datapins) and TPIU Trace (up to 13 data pins)
- Trigger 0/1 and TPIU Trace (up to 16 data pins)

- Trigger 0/1 and DSP Trace (up to 16 data pins)
- TPIU Trace (up to 18 data pins)
- DSP Trace (up to 18 data pins)

ARMSS/PMMC (via TPIU) and DSP simultaneous trace is not supported.

12.2.4 Board Design Considerations

In order to facilitate correct debug functionality between the SoC emulation signals and a particular board JTAG interface, certain procedures must be followed. More information on board design guidelines for trace advanced emulation can be found in the *Emulation and Trace Headers Technical Reference Manual* (SPRU655i).

Useful guidelines can also be found at the "XDS Target Connection Guide" TI wiki page:

http://processors.wiki.ti.com/index.php/XDS_Target_Connection_Guide

The previous link connects to TI community resources. Linked contents are provided “AS IS” by the respective contributors. They do not constitute TI specifications and do not necessarily reflect TI's views; see TI's Terms of Use.

12.3 DSP Subsystem Debug Features

The following sections define the types of emulation actions possible for the DSP subsystem (C66x CorePac).

The DSP supports:

- Invasive debug techniques, such as:
  - Halt mode debug (Section 12.3.1.1)
  - Real-time debug (Section 12.3.1.2)
  - Monitor mode debug (Section 12.3.1.3)
- Non-invasive debug mechanisms, such as:
  - Trace (Section 12.3.2)
- Triggering options possible with AET (Advanced Event Triggering) supported by C66x (Section 12.3.3)

12.3.1 DSP Debug Modes

12.3.1.1 Halt Mode Debug

The DSP supports traditional stop-mode emulation, whereby host tooling can halt the C66x core, as well as any of the device peripherals. The C66x core has a direct connection to the JTAG interface. Emulation has full visibility of the DSP memory map, and can access the chip memory through the core.

The core has visibility requirements critical for successful software development and debug in the field.

The following are requirements in a debugger environment (host control through JTAG)

- Run-time execution control (e.g. halt, step, run)
- Accurate reflection of memory and register contents
- Visibility into system stalls
  - Bank conflict
  - Cache miss/coherency overhead
  - Memory latency
  - DMA conflict
  - Cache state (coherency with external memory)
- Shared memory view:
  - Cache state (coherency between cores)
  - Conflict between cores (performance degradation)
- Shared atomic resources (memory structure, IP, etc.)
  - Ownership through Hardware Semaphore – view/force
  - IPC flags – view/force
  - Volatile shared structs – view/update
- Shared multichannel resources (e.g. DMA channels)
  - View/change context without impacting other cores
- Cross-triggering
  - Local halt, step, run
  - Global halt, step, run (synchronized)
  - Local hardware breakpoint
  - Global software breakpoint (assert and receive)
  - Global hardware breakpoint (assert and receive)

12.3.1.2 Real-Time Debug

Real-Time Debug (DSP-only feature) provides a base set of capabilities for real-time execution control (run, step, halt, etc.) and register/memory visibility. This infrastructure component allows the user to debug application code while interrupts designated as real-time continue to be serviced. Registers and memory may be accessed while code is running or stopped. Users define the interrupts that are to be treated as real-time through a special debug interrupt enable register (DIER). The user can also mark code that must not be disturbed by real-time debug memory accesses within the application. These facilities provide the basic execution control and memory/register accessibility needed by the Code Composer Debugger.

Real-time debug encompasses three code execution states. These states generate bus activity that is presented to triggering logic. An applications developer may want to constrain triggering behavior to a limited set of these states. Before triggering behavior can be discussed, it is important to understand the three CPU code execution states related to debug:

- Normal code execution – running code prior to a debug event (normal code and interrupt service routines). Normal code execution is the behavior of the system as if the emulator is not connected to the chip where the peripheral resides. In other words, the CPU is running code without a debug event halting execution with the peripheral operating continuously.
- Secondary code execution – running code related to the service of a real-time interrupt after a debug event has halted code execution. The CPU execution is designated by the applications developer as real-time, allowing the service of interrupts designated as real-time (via the debug interrupt enable register) after code execution is halted.
- No code execution – not running code. This occurs when the emulation functions are enabled, a debug event halts code execution, and no real-time interrupt is being serviced after code execution is halted. Generally, break detection is always enabled for normal code execution. The applications developer may desire to either include or exclude secondary code execution. A programmable option is provided to either include or exclude secondary code execution in trigger generation.

12.3.1.3 Monitor Mode (Run Mode) Debug

It is possible to have a real-time debug thread of execution that provides application-level debugging features at run-time that complement the Stop-Mode/Real-Time Emulation features noted above. The debug thread would respond to commands sent by a host debugger and configure the AET logic to raise an AINT interrupt whenever a user-specified trigger condition occurred. The ISR for the AINT interrupt would be handled by the real-time debugger, which would implement whatever functionality was required to perform the required debug actions requested by the host debugger.

The debug thread software would be implemented as a library that is linked in by the customer application on the device, and would give the ability to:

- Suspend a specified task (on interruptible region)
- Resume execution of a the task
- "Single step" the task (i.e. set a pending interrupt flag, resume execution of the task and suspend execution of the task when the pending interrupt is serviced)
- View/modify memory, CPU registers

12.3.2 DSP Trace

12.3.2.1 DSP Trace Overview

DSP trace provides for the recording of program flow, memory references, and application-specific data with a timestamp. Trace targets the debug of unstable code, performance analysis, and quality assurance. The use of dedicated hardware to both collect, buffer, transfer trace data to the host makes trace non-intrusive. Data is exported via trace packets. These packets are converted to trace transmission packets via the trace export block. Transmission packets can be from 1 to 20-bits wide. This allows all or a percentage of the Debug Port pins to be allocated to trace. This allows the sharing of Debug Port pins between trace and other functions, with the number of pins allocated to trace and the corresponding trace port width varied at run-time.

The trace function can collect and export a record of the program flow and timing at the same rate generated by the CPU. Tracing data references must be restricted, however, as the export mechanism is generally limited to a number of pins insufficient to sustain tracing of all memory references. In the case of data trace, the Advanced Event Triggering facilities provide a means to restrict the trace data exported to data of interest to maintain the non-intrusive aspect of trace. This reduces the export bandwidth needs and facilitates the successful collection of the data of interest. Error indications are embedded in the debug stream in the event the export logic is unable to keep up with the data rate generated by the collection logic. This notification allows the user to scale back the amount of requested data collection.

The user can optionally select the export of all specified trace data. In this case the CPU is stalled to avoid the loss of trace data, with trace becoming intrusive to the application if trace related stalls are generated. The user is notified that trace stalls have occurred although the number of stalls and their location is not recorded.

The trace technology is modular with different collection elements for different types of data (PC, Timing, Memory References). This allows the customization of the trace function if the trace function is not encapsulated in the CPU megamodule. Collection elements for different types of data are modular allowing the customer to trade off functionality versus gates and pins required to meet the bandwidth requirements. Trace export shares debug pins with other functions.

Another option to export the trace data is via TETB buffer. DSP supports one dedicated TETB with 4KB memory. The trace data in TETB can be drained to some other memory (L2, shared, or external) while continuing to trace. Trace data can be further exported by using an application interface such as Ethernet, PCIe and etc. However, these measures introduce some intrusion to the normal operation of the system.

12.3.2.2 DSP Trace Clock

DSP trace clock is derived from the Main PLL in conjuction with the PLL Controller. This clock is used for both:

- Storing DSP trace data on-chip into dedicated trace buffer (TETB)
- Exporting DSP trace data off-chip via Debug Port pins

Note that the DSP trace clock is referenced as "GEM_TRACE_CLK" from PLL perspective, and is referenced as "TRCCLK" from Debug Port perspective (Table 12-4).

For more information about DSP trace clock, Main PLL, and PLL Controller, see section 5.2, *Power Management*.

WARNING

When using TI's provided tools (CCS with an XDDS560T or XDS560v2 Trace Pro), do not modify the PLL used for DSP Core Trace. This will cause conflicts with TI's software that uses a calibration scheme to set the trace data rate at the fastest rate possible the receiver can collect data at.

12.3.2.3 DSP Trace Export Configuration

DSP trace data is formatted into 1 to 20-bit packets. For external capture through the Debug Port, the trace port has a width of 20 bits. This port mapping is software configurable to 16 or 18 data pins (TRCDTa[15:0] and TRCDTb[17:0], respectively) plus two pins dedicated to the export clock (TRCCLKA and TRCCLKB). See Section 12.2.3 for details.

12.3.3 DSP Advanced Event Triggering (AET)

Triggers manage breakpoints, trace acquisition, data collection via an interrupt, timing measurement, and generate external triggers. They also control a state machine and counters used to create the intermediate events (loop counts and state machines). AET uses instruction and data bus comparators, auxiliary event detection, sequencers/state machines, and event counters to identify events of interest. These events can be combined to create simple or complex triggers using modules call trigger builders. AET logic is provided for monitoring program, memory bus, auxiliary input activity, remembering event sequences, counting event occurrences, or measuring the interval between events. Bus comparators can perform range and identity comparisons, detect exact transactions or the mere touching of a byte or range of bytes by memory references. External event detectors provide a means to monitor external triggers or internal states of interest (i.e., cache miss). A state machine with four states that allows transitions from any state to any other state provides for the identification of a sequence of triggers, with counters/timers (16-bit and/or 32-bit) providing for the implementation of loop counts or a predetermined number of events. The state machine and counter components are controlled by triggers and generate events (state number and count equals zero). The state machine and counter events are part of the event pool used to create triggers. Trigger builder components are used to create triggers from the events created by all event generation sources.

Triggering facilities are used to identify specific system activity to generate breakpoints, an interrupt used for the collection of system data, or the identification of program activity that is observed through Trace.

Any system event routed to the C66x core can be routed (through software selection) to the AET. This is controlled through software by configuring the DSP interrupt controller as well as the Chip-level Interrupt Controller (CIC). The DSP interrupt controller supports routing up to eight events to the AET logic for tracking by the emulation hardware. Events that can be routed include the system events, CPU interrupts and exception inputs, and interrupt and exception acknowledges. Likewise, cross-triggering is possible by having an AET trigger on one DSP core signal (through EMU0/1) to cause an action on another core (or STM) within the chip or even on another device (assuming the device pins are connected properly).

For more information on how to program the AET for various application use, see Section 12.7.2.

12.3.4 DSP and Related Debug Components

The DSP debug features are supported by a variety of debug modules as follows:

- The ICEMAKER provides DSP emulation with the classic debug features, such as: download code, access to memory or registers, run/halt/single-step
- DT-DMA is a debug dedicated DMA engine intended for fast block data transfer between a debugger and DSP over JTAG port
- TCU unit is used for trigger management
- AET unit is used to generate debug actions for managing breakpoints, watchpoints, trace, timers/counters, event outputs to external logic of DSP, based on events detected by instruction and data bus comparators or by auxiliary event detectors. DSP AET can also handle complex events with event state machine and event counters
- DSP XMC supports 36-bit addressing and emulation can access the 36-bit address space
- ADTF provides trace formatting to convert the DSP generated trace into a format that is required by the DEBUGSS. The ADTF provides the mechanism for embedded trace with the TETB, or a path to the DRM within DEBUGSS for pin trace
- TETB (TI Embedded Trace Buffer) provides buffering of ADTF formatted trace stream and is capable of generating DMA events to enable EDMA draining of the FIFO. The TETB supports trace during run-time with no intrusion to the application. There are two primary application models that are facilitated by the TETB:
  - Crash analysis: Here, the TETB contents are not accessed during run-time, but rather only at defined points. Typically this would be when an external host detects a system failure. In this case, the TETB can be set to trace continuously, and when a system fault occurs the host reads the TETB contents for offline analysis of the pre-fault execution.
  - Continuous trace: The TETB can be drained during run-time through the use of EDMA channels. This adds some system loading, as EDMA resources and memory bandwidth must be consumed for this purpose. However, continuous trace allows for trace data to be:

- Stored in DDR to extend the size of the embedded trace buffer, which provides a larger history to the host
- Streamed out through a standard peripheral interface such as Ethernet, or PCIe, to be captured by an external host in the actual embedded system for real-time analysis

NOTE: The ADTF and TETB debug components are not part of the DSP subsystem. They are integrated at SoC level.

12.3.5 DSP Debug Summary

Table 12-5 summarizes the DSPSS debug capabilities.

Table 12-5. DSP Subsystem Debug Capabilities

Feature	Usage	DSP Support
Basic Debug	Execution control	Y
	System visibility	Y
Real-Time Debug	Interrupts serviced while halted	Y
	Low/non-intrusive system visibility while running	Y
Advanced Debug	Global start	Y
	Global stop	Y
	Specify targeted memory level(s) during memory accesses	Y
Advanced System Control	Subsystem reset via debug	Y
	Peripheral notification of debug events	Y
	Cache-coherent debug accesses	Y
Security	Configurable levels of security and debug visibility	Y
	Memory accesses prevented to secure memory	Y
	Debug halts prevented during secure mode	Y
Program Trace	Program flow corruption	Y
	Code coverage	Y
	Path coverage	Y
	Thread/interrupt synchronization problems	Y
Data Trace	Memory corruption	Y
Timing Trace	Profiling	Y
Analysis Actions	Stop program execution	Y
	Control trace streams	Y
	Generate debug interrupt	Y
	Benchmarking with counters	Y
	External trigger generation	Y
	Debug state machine state transition	Y
	Combinational and Sequential event generation	Y
Analysis Events	Program event detection	Y
	Data event detection	Y
	External trigger detection (cross-trigger)	Y
	System event detection (i.e., cache miss)	Y
	Debug state machine state detection	Y
Analysis Configuration	Application access	Y
	Debugger access	Y
	Debug state restoration across power cycle	N

Table 12-6 summarizes the DSPSS debug features.

Table 12-6. DSP Subsystem Debug Features

Category	Hardware Feature	DSP Support
Basic Debug	Software breakpoint (SWBP)	Unlimited and DSP also supports embedded SWBP. SWBPs are global, affecting any/all CPUs that execute an instruction with SWBP inserted
	Hardware breakpoint (HWBP)	Up to 10 HWBPs: • 4 HWBPs are in core (HWBP0 is associated with a counter) • 2 dedicated HWBPs are in AET • 4 HWBPs which are shared with watchpoint (also in AET)
Analysis	Watchpoint	4 watchpoints registers are shared with HWBPs, and can also be used as 2 watchpoints with data (32-bits)
	Watchpoint with Data (Data Watchpoint)	2 (they are shared with the 4 watchpoints).
	Counters/Timers	The DSP core has one 64-bit free running time-stamp counter (TSC)
	External Event Trigger In	2 (the two cross-triggering signals. They can be used to halt target as a HWBP (they can be called system HWBPs))
	External Event Trigger Out	
Trace Control	Address range for trace	3
	Data qualification for trace	2 (see "Watchpoint with Data")
	System events for trace control	32 events (directly connected to AET), which include: • 32 CPU memory events (cache) • 32 CPU system events (stall) • 32 others (miscellaneous) events including 8 event inputs which all internal and external interrupts events can be muxed to via the DSP internal INTC, as well as the system INTC module (CIC).
	Counters/Timers for trace control	2x32 bits
	State Machines/Sequencers	1x4-states or 2x2-states state machine(s)
	Context/Thread-ID Comparator	0
	Independent trigger control units	1
On-chip Trace Capture	Capture depth PC	4KB TETB buffer
	Capture depth PC + Timing	4KB TETB buffer
	Application accessible	Y
Performance Monitoring Unit	Application accessible cycle counters	Y (supported by AET)

12.4 Arm Subsystem Debug Features

12.4.1 ARMSS Debug Overview

The Cortex-A15 processor supports the following native features:

- Halt mode and monitor mode debug
- Six hardware breakpoints and four watchpoints
- Asynchronous aborts
- Processor trace and software instrumentation
- Performance monitoring
- Cross-triggering: allows stopping the Cortex-A15 CPU upon debug event (for example, breakpoint) detection in another device CPU or debug/trace component

For more information about Cortex-A15 native debug features, see the *Arm Cortex-A15 Technical Reference Manual*.

12.4.2 ARMSS Debug Modes

The Cortex-A15 CPU supports invasive and non-invasive debug modes.

The invasive debug mode is intended primarily for run-control debugging. Invasive debug mode provides support for:

- Monitor mode: In monitor mode, a debug event causes a debug exception to occur. A debug exception that relates to instruction execution generates a Prefetch Abort exception. A debug exception that relates to a data access generates a Data Abort exception. A software handler can then take control to examine or alter the processor state. Monitor debug mode is essential in real-time systems where the processor cannot be halted to collect debug information.
- Halt mode: In halt mode, a debug event causes the processor to enter Debug state. In Debug state, the processor stops executing instructions from the location indicated by the program counter, but is instead controlled through the external debug interface, in particular using the Instruction Transfer Register (DBGITR)

Non-invasive debug includes all debug features that permit data and program flow to be observed, but that do not permit modification of the main processor state. Non-invasive debug mode support includes Processor Instruction Trace, Sample-based Profiling, and Performance Monitoring.

12.4.3 ARMSS Trace

The Cortex-A15 MPCore integrates a CoreSight Processor Trace Macrocell (PTM or CS-PTM) which performs real-time instruction flow tracing. It generates information that can be used by trace receiver to reconstruct the execution of all or part of a program.

The PTM identifies certain instructions in the program, and certain events, as waypoints. A waypoint is a point where instruction execution by the processor may involve a change in the program flow.

The PTM traces only the following waypoints:

- All indirect branches
- All conditional and unconditional direct branches
- All exceptions
- Any instruction that changes the instruction set state of the processor
- When halting debug mode is enabled, entering or leaving debug state
- Synchronization primitives

The PTM provides the ability to reconstruct the complete instruction flow.

For more information about PTM, see *Arm CoreSight Program Flow Trace Architecture Specification*.

The Cortex-A15 processor trace characteristics are as follows:

- Program trace only (no data trace)

- Two exclusive trace sinks:
  - TPIU: Trace exported to an external trace receiver (via Debug Port)
  - CT-TBR: Trace stored to ARMSS local trace buffer for draining through high-speed serial interfaces
- Trace can be optionally:
  - Cycle accurate (useful for profiling sections of code)
  - Locally time-stamped using a 64-bit free-running graycode counter exported to the PTM

12.4.4 ARMSS Software Instrumentation

The ARMSS software instrumentation is supported by its integrated CoreSight STM (CS-STM). The CS-STM provides 'extended stimulus port' registers designated to be accessible by software with minimum instrumentation cycles overhead.

Each 'extended stimulus port' occupies 256 consecutive bytes in the memory map and is write-only. Up to 64K instrumentation channels are available at ARMSS level.

The CS-STM supports 'guaranteed' or 'invariant timing' transactions:

- Guaranteed transactions are guaranteed to be traced. This might involve stalling the Cortex-A15 core to ensure the transaction is accepted by the CS-STM
- Invariant timing transactions are not guaranteed to be traced. These transactions will take an invariant amount of time regardless of the state of the CS-STM

12.4.5 ARMSS Performance Monitoring

The Cortex-A15 processor includes a Performance Monitoring Unit (PMU) that enables events, such as cache misses and instructions executed, to be counted over a period of time. The PMU gathers statistics about the operation of the processor and memory system.

The Cortex-A15 PMU events are listed in the *Arm Cortex A15 Technical Reference Manual*.

12.4.6 ARMSS Cross-Triggering

The Arm subsystem provides cross-triggering support by implementing CoreSight Cross-Trigger Interface (CTI) and Cross-Trigger Matrix (CTM) modules. The following CTI and CTM modules are integrated in the ARMSS:

- At Cortex-A15 MPCore level:
  - One CTI. The CTI is programmable through AXI bus accesses or by debugger from APB bus directly.
  - One CTM
- At ARMSS level (that is, outside the Cortex-A15 MPCore):
  - One additional CTM to carry cross triggers between the Arm core, integrated CS-STM, integrated CT-TBR and from DEBUGSS outside the ARMSS wrapper

The ARMSS cross-triggering modules provide support for useful cases, such as cross-triggering between PTM and CT-TBR:

- PTM can generate a trigger that can be used to stop CT-TBR data capture
- CT-TBR can send buffer-full or acquisition-complete events to the PTM

12.5 PMMC Debug Features

The PMMC integrates a single Cortex-M3 processor, which supports the following native debug features:

- Program halt and stepping
- Hardware breakpoints, breakpoint instruction
- Data watchpoint on access to data address, address range, and data value
- Register value accesses
- Debug monitor exception
- Memories accesses

For more information about Cortex-M3 native debug features, see the *Arm Cortex-M3 Technical Reference Manual*.

The PMMC implements an Instruction Trace Macrocell (ITM) unit to provide trace-based debugging, such as PC sampling, data trace, event trace and software instrumentation. ITM supports two data sources:

- Software trace: generated by software application to provide hardware assisted **printf()**-like messages
- Hardware trace: system and application events generated by DWT unit

The Cortex-M3 trace is stored into a dedicated PMMC TraceCell integrated at SoC level. The PMMC TraceCell includes a CT-TBR module with 16KB RAM which can generate a DMA request to the EDMACC_0 controller for further trace draining.

12.6 SoC Level Debug Features

As devices get more complicated and more IPs are integrated into the device (SoC), there is a requirement to provide sufficient hooks in the silicon to provide enough visibility to facilitate debug in the system. Multiple cores (processing engines) executing concurrently and asynchronously stress the system memory in variable fashion during execution in response to system stimulus (events). Multiple 'DMA' transactions move a very large amount of data through the chip concurrently, converging at memory and accelerator endpoints. The data movement is a mixture of synchronous and asynchronous activity, both driving and driven by the program execution.

12.6.1 System Trace

12.6.1.1 Overview

System trace can provide significant debug visibility into how multiple cores and masters are interacting and executing in the SoC. System trace provides visibility on bandwidth consumption by SoC masters as well as transaction trace information. Its purpose is to collect system debug and performance data from the SoC and encapsulate it into a standard format. These features are accomplished on the device through both hardware and software messages.

The on-chip Tracer modules provide the functionality which generate hardware messages (as explained in Section 12.6.2).

System trace also supports the capture of software messages as a minimally intrusive means to log information during run-time. To trigger SW messages, applications write data to be logged to certain memory mapped locations of the CTools System Trace Module (CT-STM) embedded in the Debug Subsystem. The CT-STM provides 256 dedicated channels for simultaneous logging of software messages.

The CT-STM provides support for:

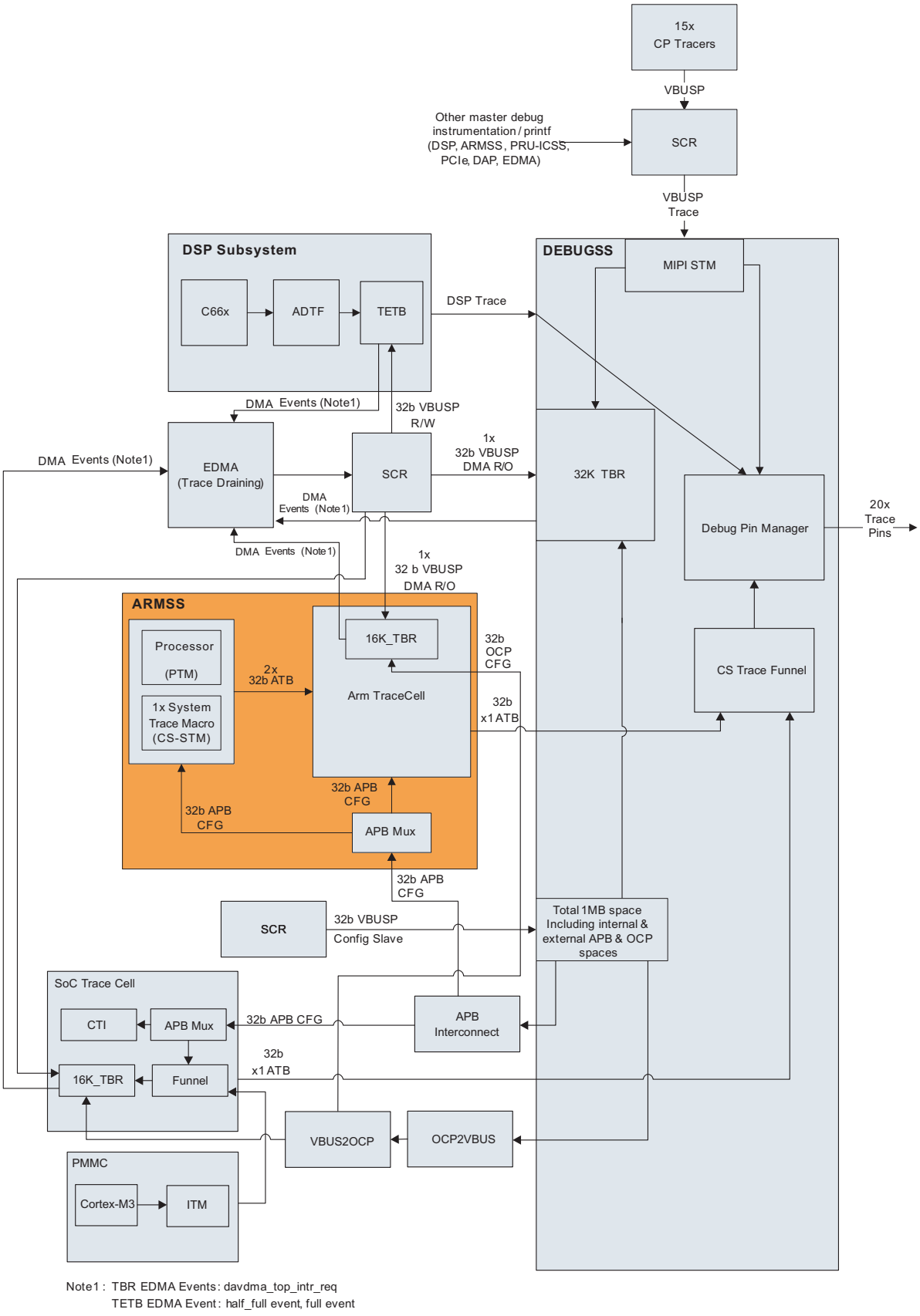
- Storing system trace data on-chip into dedicated CTools Trace Buffer Router (CT-TBR)
- Exporting system trace data off-chip to an external trace receiver via the Debug Port

When storing data on-chip, the CT-STM interfaces with the dedicated CT-TBR through the ATB port.

When exporting data off-chip, the CT-STM interfaces with the Debug Resource Manager (DRM) module through the Parallel Trace Interface (PTI). The system trace port width and clock can both be configured through the PTI. More information on configuring the PTI can be found in Section 12.7.2.

Figure 12-1 shows the overall SoC trace architecture with trace generation of Cortex-A15 core, DSP core, Cortex-M3 core, hardware instrumentation (Tracers), and software instrumentation. The figure also shows the trace draining using EMU pins in addition to on-chip buffers for trace storage (TETB for DSP; CT-TBR(s) for ARMSS, CT-STM, and PMMC).

Figure 12-1. SoC Trace Architecture



Note1 : TBR EDMA Events: davdma_top_intr_req
TETB EDMA Event: half_full event, full event

12.6.1.2 System Trace Module

The Debug Subsystem integrates the CTools System Trace Module (CT-STM) to handle hardware instrumentation/messages (generated by the Tracer modules) and software instrumentation/messages (from various processing cores).

The main features of the CT-STM are as follows:

- Implements MIPI STP protocol (rev 2.0) with the following characteristics:
  - Highly optimized for software-generated traces from ARMSS, DSP, PMMC, PRU-ICSS sources
  - Automatic timestamping of messages
  - Support for 8-, 16-, and 32-bit data types
- Collects the following information:
  - Software messages
  - Hardware instrumentation trace from Tracer modules
- Supports following types of trace sinks:
  - Pin trace (EMU pins) through the Parallel Trace Interface
  - Capture of trace into DEBUGSS internal buffer (CT-TBR) through ATB interface
- Pin trace available in 1-, 2-, or 4-pin mode with single- or dual-edge clock, depending on the trace bandwidth requirements and characteristics of the trace receiver

12.6.1.3 System Trace Sinks

12.6.1.3.1 Trace Capture into CT-TBR

The CT-TBR module residing in the Debug Subsystem provides on-chip storage of system trace data using a 32-KB RAM memory. The CT-TBR receives trace data through its ATB port. The debugger can then access the trace data through the APB port of the CS_DAP module.

The CT-TBR can act like a circular buffer; that is, it can continuously capture and write data into memory when trace capture is enabled and its trigger counter is static at 0 or has not yet decremented to 0.

Trace window can be adjusted around a specific trigger spot using the CT-TBR trigger counter:

- Trace before
- Trace after
- Trace around

Once the capture has been disabled, the trace history can be read from the CS_DAP.

12.6.1.3.2 Trace Export Through Debug Port

For exporting trace data off-chip to an external trace receiver through the Debug Port, the CT-STM has a configurable export width of 1, 2, or 4 data pins (STM_DATA) plus a dedicated export clock (STM_CLK). Data width and clock selection are configured through the PTI and should not be changed on-the-fly.

STM trace export clock is derived from the Main PLL in conjuction with the PLL Controller.

Note that the STM trace export clock is referenced as "STM_CLK" from PLL perspective, and is referenced as "TRCCLK" from Debug Port perspective (Table 12-4).

For more information about STM trace export clock, Main PLL, and PLL Controller, see section 5.2, *Power Management*.

12.6.1.4 Software Messages

STM-based SW message is a software instrumentation method, which is light-weight in term of intrusiveness, and is very easy to use. With it, an application directly writes data to be logged to some memory mapped location of CT-STM, and CT-STM will automatically capture and export the data to external receiver or to the dedicated on-chip CT-TBR buffer. To have this, all SW masters have a write access path to the memory space of CT-STM, and each SW master has a distinguished Master ID.

The list of SW masters supported on this device is as follows:

- DSP C66x
- Arm Cortex-A15
- PMMC Cortex-M3
- PRU-ICSS_0 PRU0/1
- PRU-ICSS_1 PRU0/1

CT-STM supports 256 channels for each SW master. If up to 256 tasks (threads) running on same processor use different channels, they can log data to CT-STM simultaneously without any mutex/semaphore protection.

CT-STM provides multiple registers for configuring which software masters are enabled for generating trace messages. The cTools libraries then provide various APIs for generating the software messages.

12.6.1.5 Hardware Messages

Hardware messages are generated by the Tracer modules. For more information, see Section 12.6.2.

12.6.1.6 Message Interleaving

The CT-STM architecture allows interleaving of software and hardware messages. These flows are typically asynchronous and could result in concurrent or very close accesses arbitrated by the DEBUGSS instrumentation interconnect and signaled by a MASTER message. The CT-STM buffer allows absorbing write accesses peaks resulting from interleaving.

12.6.1.7 Master ID Encoding

The CT-STM uses a Master ID field to differentiate between hardware and software messages. It uses the following field encodings:

- MReqMstID[7]: Differentiates software masters versus hardware masters (0 signifies a software master; 1 signifies a hardware master)
- MReqMstID[6-2]: Signifies the ID that has been assigned to the master by the TeraNet configuration
- MReqMstID[1-0]: Additional qualifier for multicore masters. Unused on this device

For master ID values of all HW and SW masters, see Chapter 3, *Interconnect*.

12.6.1.8 Timestamping

The CT-STM supports automatic timestamping upon receiving system trace messages. The timestamp is exported along with the message.

Table 12-7 shows the structure of the CT-STM memory space for multiple channels.

Table 12-7. CT-STM Channel Memory Space

Byte Address		Channel	Size	Description
0x00000	0x007FF	0	4-KB	Data message (no timestamp)
0x00800	0x00FFF			Data message with timestamp
0x01000	0x017FF	1	4-KB	Data message (no timestamp)
0x01800	0x01FFF			Data message with timestamp
0x02000	0x027FF	2	4-KB	Data message (no timestamp)
0x02800	0x02FFF			Data message with timestamp
0x03000	0x037FF	3	4-KB	Data message (no timestamp)
0x03800	0x03FFF			Data message with timestamp
...	...	...	...	...
0xFF000	0xFF7FF	255	4-KB	Data message (no timestamp)
0xFF800	0xFFFFF			Data message with timestamp

The application software shall write to:

- The lower 2-KB window, to trigger a data message without timestamping
- The upper 2-KB window, to trigger a data message with timestamping

1-KB address spaces can also be allocated for each channel, in which case writing to the upper 512 bytes would trigger a data message with timestamping. The length configuration is not exclusive – one master may instrument its application code through 4-KB channels and another may do so through 1-KB channels simultaneously. Both hardware and software masters may operate this way.

The CT-STM also supports local timestamping where the timestamped messages are stored to the CT-TBR. The user should enable local time-stamping mode to use this feature. In this case, the CT-STM can receive one hardware message (32 bits) per cycle. Considering the hardware message only case, the CT-STM adds 4 bits additional information to each hardware message or 12 bits additional information to each hardware message if the CT-STM is configured to add a timestamp. The CT-STM can send processed hardware messages to the CT-TBR at 32 bits per cycle. Therefore, the sustainable rate is 8/9 hardware messages per cycle, assuming the timestamp is not added. The CT-STM can buffer up to 128 full messages at a time. We can assume the CT-STM consumes 8/9 messages per cycle and buffers 1/9 messages per cycle. Consequently, it can process up to 128*9 = 1152 back-to-back hardware messages. When the timestamp is added to the hardware message, the sustainable rate is 8/11 hardware messages per cycle and the CT-STM should be able to process up to 128 / (1-8/11) = 469 hardware messages back-to-back.

When the CT-STM runs out of buffer space, it will hold the reception of messages from the Tracer modules. This may cause overflow of hardware messages within the Tracer modules. The hardware messages do carry an overflow indication.

12.6.1.9 CT-STM Message Format

Table 12-8 shows the message format for packets processed by the CT-STM.

Table 12-8. CT-STM Message Format

ID	Data	Mnemonic	Description
0001	M[7-0]	MASTER	Master ID
0010	O[7-0]	OVRF	Number of overflows
0011	C[7-0]	C8	Channel
0110	D[31-0]	D32	Data
1010	D[31-0] T[7-0]	D32TS	Data and timestamp

This message format is used by both software and hardware masters. Data words are stored internally in 44-bit packets within an internal FIFO before being exported through the ATB (for CT-TBR) or PTI (for external trace receiver) interfaces.

12.6.1.10 Overflow

When there is overflow of input data to the CT-STM from hardware messages, the CT-STM will stall input from any source to avoid overflowing its buffers.

12.6.1.11 Reset

The CT-STM is reset with a hardware power-on-reset (POR). All CT-STM registers are asynchronously reset. The CT-STM shall remain operational and with its setup preserved when a warm reset is triggered in order to export the trace history to the trace controller. This trace history may contain valuable information from a debug perspective allowing the user to understand the root cause of the warm reset (e.g. security violation). Making the CT-STM unsensitive to warm reset will also allow tracing the transactions processed right after the reset without reconfiguration of the System Trace.

The application or debugger software can reset the CT-STM at any time by writing a 1 to the SOFTRESET bit of the CT-STM System Configuration Register. This will put the CT-STM in the same reset state as a hardware reset.

12.6.2 Tracer Architecture and Operation

This section provides an overview of the Tracer architecture and describes the operational concepts of the Tracer module, including event generation, message formatting/exporting, and integration with both the DEBUGSS System Trace Module (CT-STM) and DEBUGSS Trace Buffer and Router (CT-TBR).

NOTE: The Tracer modules are also referred to as CP_Tracers (Common Platform Tracers).

12.6.2.1 Purpose of the Tracer

Tracer modules are used in conjunction with their supported SoC masters to generate system trace events. Corresponding messages are then generated based on these events and can be read to obtain information about the status and performance of the system.

12.6.2.2 Tracer Features

The following features are available for the Tracer modules:

- One hardware Tracer module for each supported slave interface, each containing the following:
  - Support for events A through G as described in Table 12-9
  - Unique event inputs A, F, and G for each supported master
  - Event inputs B, C, and E shared among supported masters
  - Dedicated FIFOs for events A, B, C, and E
  - Message filters
  - Two throughput counters
  - Accumulated Wait Time counter
  - Num Grant counter
  - Message Formatter
  - VBUSP slave port for CPU access to Tracer MMRs
  - VBUSP master port for access to STM port of Debug Subsystem
- Multiple Tracer modules on the device. Assignment is as follows:
  - TRACER_0 for MSMC SRAM bank 0
  - TRACER_1 for DDR_EMIF
  - TRACER_2 for CORE
  - TRACER_3 for PCIE
  - TRACER_4 is reserved
  - TRACER_5 for MCASP/MCBSP
  - TRACER_6 for GPMC/MMC/QSPI
  - TRACER_7 for EDMACC_0/EDMACC_1
  - TRACER_8 for CFG SCR
  - TRACER_9 for ALWAYSON_CFG
  - TRACER_10 for ARM_GIC
  - TRACER_11 for CIC
  - TRACER_12 for ROM/SPI
  - TRACER_13 for ALWAYSON_MAIN
  - TRACER_14 for PRU-ICSS
- Common Bus Architecture (CBA) event generators embedded in the SCR that generate master events for CorePacs, IPs, SRAMs, and the CFG SCR
- CBA slave event interfaces for each Tracer
- Capability to export three different types of messages from captured event data (event, statistics, and

status messages)

- Tracers run on either CHIP_CLK1/1 (TRACER_0, TRACER_1) or CHIP_CLK1/3 (TRACER_2...14) clock
- Multiple master connections to each slave Tracer module

12.6.2.3 Tracer Architecture

The Tracer module generates and exports system trace messages based on events triggered by the SoC masters that support it. The Tracer module is used to collect real-time data from a system by setting up various counters and filter modes provided by the module. It uses synchronous CBA event generators through four different event interfaces connected to the SCR to format and export trace messages. It also has the capability to provide various filter modes to the events and export the produced messages to the CT-TBR inside the Debug Subsystem.

The Tracer also has statistics counters to monitor the real-time performance of the CBA infrastructure. These counters are also triggered by the CBA event generators built into the SCR. All event messages are sent through a VBUSP interface. Each Tracer connection is specified by a unique SID.

Figure 12-2 shows how the Tracer functional block is connected to other components of the SoC. The Tracer module supports connections from multiple supported masters to its CBA event inputs. Various masters trigger CBA event generators on their own respective modules according to which events are occurring. In addition to the original transaction requests sent from the masters to the targeted slave, these events are sent to the CBA slave interfaces on the Tracer module, where the Tracer forms messages to send to the STM for reformatting. Messages originating from the Tracers are 32-bits wide and are reformatted into larger packets within the STM. An additional 4 bits are used by the STM to signify that the message is a hardware message. The STM then sends messages to the CT-TBR or to an external trace receiver through the Debug Port.

Figure 12-2. Tracer Connection

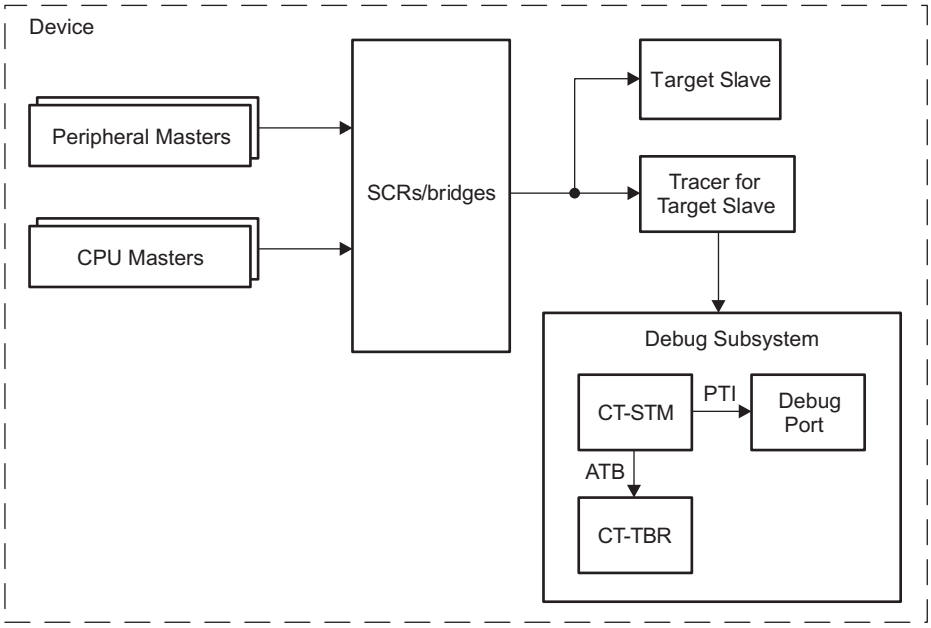
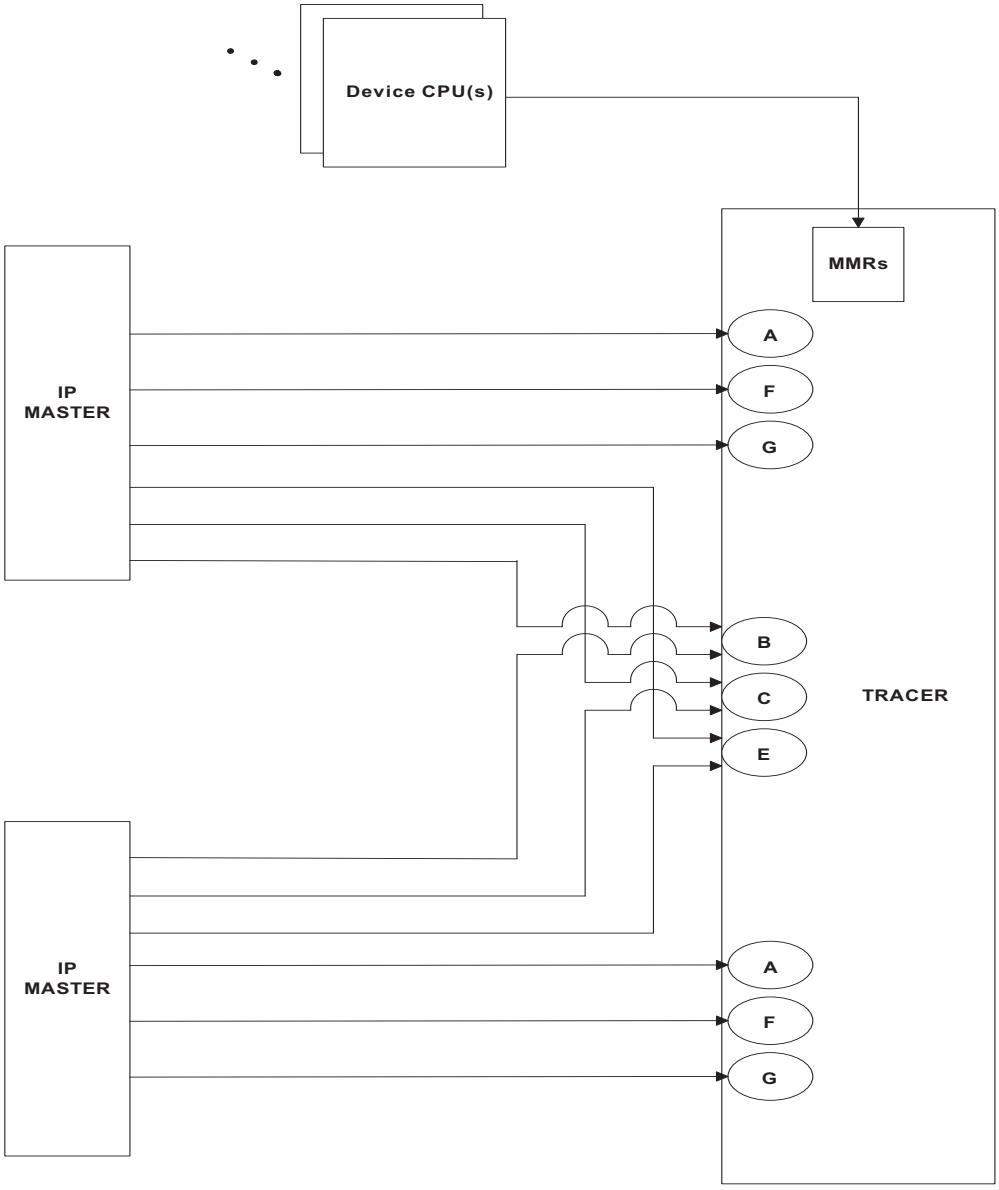


Figure 12-3 shows the individual CBA event connections from multiple IP masters to the Tracer for a particular slave. Each Tracer module has unique event A, F, and G inputs for each IP master it supports. For events B, C, and E, there is only one input for each of these events per Tracer module that is shared among supported IPs.

Figure 12-3. CBA Event Connections



12.6.2.3.1 Hardware Instrumentation through Tracers

There is a large number of masters in the SoC, all of which drive data transactions through the switch fabric (system interconnect), a certain minimum amount of visibility is required to balance priority and scheduling for successful system integration and debug. The following are requirements both in a debugger environment (host control through JTAG) and in the embedded system (host control through application software)

- Resource (memory) usage – Ability to track the bandwidth consumed by the system masters.
  - Bandwidth (#bytes/time)
  - Provide two configurable system master groups, and bandwidth tracking for each group
- Transaction trace – Ability to monitor the data transactions occurring by each master, to selected slave interfaces through tracing of key transaction points:
  - Arbitration won
  - Last data phase (transaction complete)
  - Two filtering functions for transaction traces to bring out the specific transactions:
    - Transaction-qualifier-based filtering: read/write
    - Address-range-based filtering
- Cross-triggering to turn on/off exporting capability from DSP/ARMSS.

To support this visibility, the Tracer module is integrated into the chip to monitor and trace the transactions to each of the high-speed memory slave ports of the main SCR(s). The Tracer is also used for tracing and monitoring the transactions sent to the configuration SCR peripherals.

NOTE: There is a bottleneck in the STM, such that it is not possible to activate all Tracer instances in the device at the same time.

The sliding time window specifies the measurement interval for all the CBA statistic counters implemented in the Tracer module. The sliding time window is specified in terms of the Tracer clock cycles. All the counters that are enabled start counting at the first transaction after the sliding window begins. When the sliding window timer expires, the counter values are loaded into the respective registers and the count starts again. If enabled, an interrupt is also generated when the sliding time window expires. The host CPU and/or EDMA can read the statistics counters upon assertion of the interrupt. If enabled, the counter value can also be exported to STM automatically after the sliding time window is expired.

12.6.2.3.2 Event Interface

The CBA event generation logic embedded in an SCR can generate events based on the transaction occurring at a master interface. The events are generated for every transaction targeted towards a particular slave interface. The master interfaces and the target slave interface for which the events are generated can be configured when the SCR is generated. Care should be taken such that the same transaction does not generate multiple events as it traverses through different SCRs in a system.

Tracer has interfaces to monitor and log the following events:

- New request event from master (Event A)
- New request event to slave (Event B)
- Last write data event from master (Event C)
- Last read data event to master (Event E)
- Write Merged (Event F)
- Command Discarded (Event G)

Tracer module provides the following filtering features:

- Filtering based on mstid on events B and E
- Filtering based on read/write on event B
- Filtering based on dtype on event B
- Filtering based on address range (inclusive of addresses within the range and exclusive outside the

range) on event B

- Filtering based on EMU0/1 control inputs on all events B, C and E

12.6.2.4 CBA Event Generation

For any transaction that targets a supported slave interface, an event can be generated to monitor the transaction from the master to the slave through the CBA event generators of Tracer. The user can determine how these events are generated and how they are exported.

When transaction requests are generated from an SoC master to a particular slave, that master can send event signals to the Tracer module for the targeted slave. The Tracer module in turn will route information to the CT-TBR within the Debug Subsystem or to an external trace receiver through the Debug Port. It is important to make sure the same transaction does not generate multiple events as it traverses the different SCRs in a system. Table 12-9 shows the different event types that can be triggered by Tracer.

Table 12-9. Tracer Events

Event	Action	Function
A	Master requesting to slave	This event triggers when there is a new request from the master decoded to the slave.
B	New request to slave	This event triggers when a transaction is sent to the slave.
C	Last write data to slave	This event triggers when the last write data is sent to the slave, thus completing the write burst.
E	Last read data from slave	This event triggers when the last read data arrives at the slave interface, thus completing the read burst.
F	Merge	Indicates that a write request from <mst> to <slv> has been merged with another request
G	Discard	Indicates that a read request from <mst> to <slv> has been discarded.

12.6.2.4.1 Event A - Master Request Event

Event A represents a master request transaction and is used for generating status and statistics messages from the Tracer. For example, consider a master issuing a write request to a target slave. The master could generate a transaction write request and at the same time generate an Event A on its CBA master interface which would then be sent to the CBA slave interface of the Tracer module for the targeted slave.

12.6.2.4.2 Event B - Arbitration Event

Event B represents a standard request to a slave that has been granted arbitration. With this event we can monitor many different transactions, so the user has the ability to establish filter modes to enable monitoring of the following:

- Trigger event when a transaction is sent to a slave. This signal interface does not track which master initiated the transaction. The Tracer uses dedicated event A CBA interfaces for each supported master for tracking the frequency of transactions from specific masters.
- Trigger event when the last transaction is arbitrated for a given command.
- Send the master ID associated with the given transaction.
- Send the arbitration direction associated with the given transaction.
- Send the dtype (0 for CPU Data Access, 1 for CPU Instruction Access, and 2 for DMA Access) associated with the given transaction.
- Send the transaction ID associated with the given transaction.
- Send the destination address associated with the given transaction.
- Send the byte count associated with the given transaction. This byte count is added to one of the throughput counters based upon how we configure the Transaction Qualifier Register. The throughput can be monitored based on a configurable timing window. The throughput gathered during the last timing window can then be read from one of two throughput registers, Throughput0 or Throughput1.
- The Transaction Qualifier Register and the Master ID Select Register N can be used to set any filtering

modes for generating Event B messages.

12.6.2.4.3 Event C - Write Completion Event

Event C is triggered when the last set of write data is sent to the targeted slave. This event also signifies the end of the write burst.

12.6.2.4.4 Event E - Read Completion Event

Event E is triggered when the last set of read data is sent to the targeted slave, completing the read burst. The associated master ID and transaction ID is also sent with this event. The MasterID Select Register N can be used to set any filtering modes required for this event.

12.6.2.4.5 Event F - Write Merge Event

Event F indicates that a write request from <mst> to <slv> has been merged with another request.

12.6.2.4.6 Event G - Read Discard Event

Event G indicates that a read request from <mst> to <slv> has been discarded.

12.6.2.5 Transaction Flow

Figure 12-4 shows the overall transaction flow of a Tracer event. The event is generated from the master interface with the transaction request. The Tracer creates the hardware message from the event that is routed to the STM to be stored on-chip or exported externally.

Figure 12-4. Tracer Generation

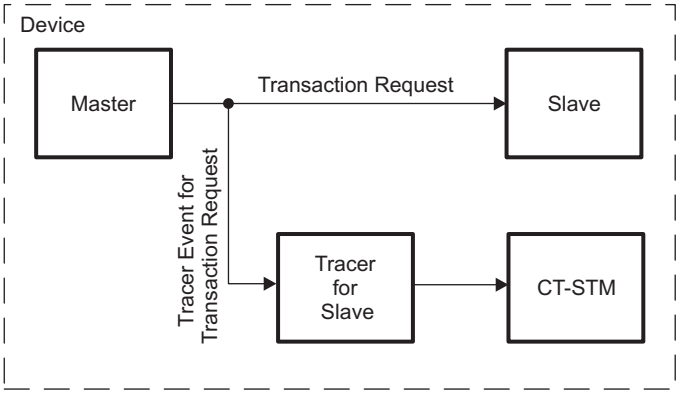


Figure 12-5 illustrates how an Event A can be generated and propagated to a Tracer module through a simple write to L2 memory.

Figure 12-5. Example—Event A to Tracer

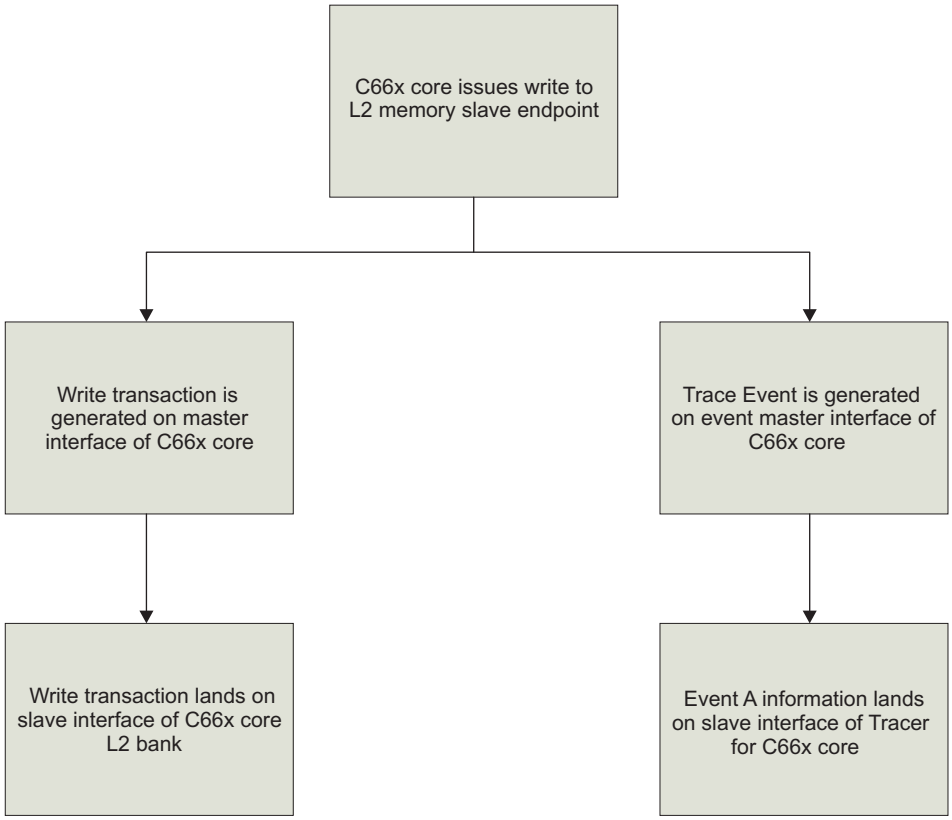
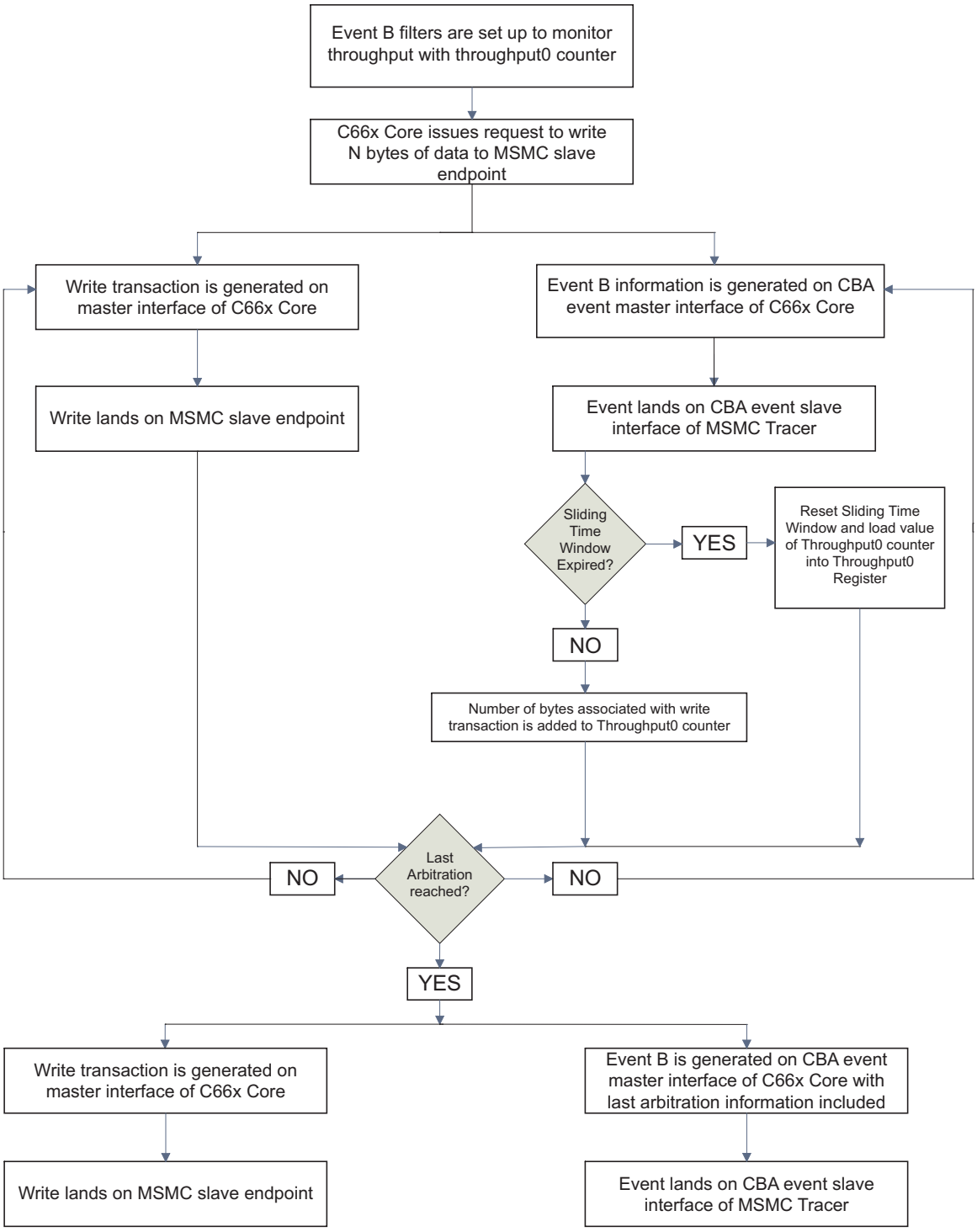


Figure 12-6 illustrates how an Event B is be generated and propagated through the Tracer module through a write to MSMC RAM.

Figure 12-6. Example—Event B to Tracer



NOTE: There is only one Event B interface for the Tracer module for the MSMC target that is shared amongst masters, whereas there are unique Event A interfaces on this module for each supported master.

12.6.2.6 FIFOs

The Tracer modules implement the following FIFOs:

Event FIFO: This FIFO queues the event signals arriving at the CBA event slave interface of the Tracer.

Message FIFO: This FIFO queues the messages that are constructed by the Tracer logic after an event has been received. As long as there is at least one message pending in this FIFO, the VBUSP trace transfer controller will issue a 32-bit transfer to the STM.

12.6.2.7 Trace Message Formats

The Tracer modules generate different message types according to the events they receive at their CBA event interfaces. These messages contain all the information that the corresponding events were generated and filtered on. Event A is recorded through status messages, whereas events B, C, and E are recorded through event messages. The following sections describe the formats for the different messages generated by the Tracers, STM, and TETB.

12.6.2.7.1 Status Messages

Status messages (Figure 12-7) are generated for Event A and provide information on request interaction between masters and slaves.

Figure 12-7. Status Message Format

31 29	28 24	23	22 0
Type=000	SID	G	Access Status

The SID specifies which unique Tracer this message is generated from.

The Access Status field is used to associate the transaction with a particular master event I/F. Each bit of the Access Status field signifies whether a request has happened on that particular Event A interface since the last status message was exported. A '0' means that no request event happened on that Event A interface since the last status message was exported. A '1' means that one or more request event happened on an event A interface since the last status message was exported.

As we can have up to 32 interfaces from which status messages are generated from and a limited amount of bits for representation, the status message can appear in two formats, shown in Figure 12-8.

Figure 12-8. Status Message Format

31	30 29	28 24	23	22 0
0	00	SID	0	Access Status (I/F 22-0)

31	30 29	28 24	23	22 9	8 0
0	00	SID	1	All 0	Access Status (I/F 31-23)

In the first configuration, bits 0-22 represent master interfaces 0-22. In the second, bits 0-8 represent the master interfaces 23-31. The two types are differentiated by the G field, bit 23.

12.6.2.7.2 Event Message

Event messages are generated for events B, C, and E. Figure 12-9 shows the message format for event messages.

Figure 12-9. Event Message Format

31 29	28 24	23 16	15 12	11	10	9 0
Type	SID	MID	XID (3-0)	0	R/W	Address (9-0)

The type field is encoded as following:

- 0b001: event B
- 0b010: event C
- 0b011: event E

The other fields are as follows:

- SID specifies which unique Tracer this message is generated from
- MID specifies the unique master ID from which the transaction originated to this Tracer slave endpoint.
- XID specifies the transaction ID.
- O indicates overflow status.
- R/W can be used to determine whether the transaction was read or write.
- The Address field can be used to determine the 10 bits of the associated 48-bit address that are exported with the event. The Address Mask Register is used to determine which 10 bits are exported.

Not all fields are valid for all event messages. For each event message the following fields are valid. All other fields are read as 0:

- Event B: Type, SID, MID, XID, O, RW, Address
- Event C: Type, SID, O
- Event E: Type, SID, MID, XID, O

12.6.2.7.3 Statistics Message

Each statistics message contains 4*32 bits as shown in Figure 12-10:

Figure 12-10. Statistic Message Format

31 29	28 24	23 0
Type=100	SID	Throughput_count0 (23-0)
Type=101	SID	Throughput_count1 (23-0)
Type=110	SID	Total_wait_time (23-0)
Type=111	SID	Total_granted (23-0)

Throughput_count0—This field equals the throughput from register Throughput0.

Throughput_count1—This field equals the throughput from register Throughput1.

Total_wait_time—This field is read from the Accumulated Wait time counter.

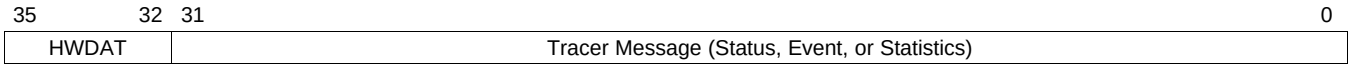
Total_granted—This field gives the reading of the grant counter which counts the number of times arbitration has been granted. The value is essentially equal to the number of B events that have occurred with arb_last asserted and is read from the Num Grant Counter.

Each of these fields is implemented by a dedicated counter in each Tracer and each start counting at the first transaction after the sliding time window begins. The sliding time window is explained in Section 12.6.2.10.

12.6.2.7.4 STM Message

The various messages from the Tracer modules are routed to the STM in the Debug Subsystem. Figure 12-11 shows how the messages originating from the Tracer modules will be encapsulated within STM.

Figure 12-11. STM Message Format



STM will pack incoming Tracer module data into 36-bit messages, 32 of which are data from the Tracer module and the most significant four bits are always 0b0110 to signify a HWDAT (Hardware Data) message.

12.6.2.8 CBA Counters and Statistics

Each Tracer's statistics counters allow the following statistics to be calculated:

- Bus bandwidth to slave used by one or more selected bus masters (bytes/sec.) = throughput for bus master X / sliding time window duration
- Average access size = throughput byte count / num accesses granted
- Bus utilization (transactions per second) = Num accesses granted / sliding time window duration
- Percentage of time there was contention for the bus = (accumulated wait time / sliding window length in cycles) * 100
- Minimum Average Latency = Accumulated Wait Time / number of accesses
- Percentage of bus throughput used by bus master X = (throughput for bus master X / throughput for all bus masters) * 100

, where sliding time window duration = sliding time window period in cycles / # cycles per second.

NOTE: The Tracers cannot measure average latency and average wait time. The Minimum Average Latency is not a true average arbitration latency, since it ignores the cycle counts where multiple bus masters are waiting at the same time. It will typically be lower than the true average latency.

12.6.2.9 Throughput Counters

Each Tracer module contains two throughput counters that can be enabled and filtered individually through specific registers. These counters are used in correspondence with Event B and exported as part of the statistics message. The throughput counters accumulate and record the total number of bytes forwarded to the target slave during the specified time duration. Filtering can be performed by address range, emu type, dtype, read/write transaction type, and master ID.

12.6.2.10 Sliding Time Window

Each Tracer module implements a sliding time window, measured in Tracer clock cycles, which specifies the total interval in which all CBA statistics counters are recorded. The Tracer clock cycles are tied to DMA cycles. In consequence, excluding CPU instruction or Data cycles in calculation will have no affect on the Tracer clock cycles computed. Excluding DMA cycles will exclude all Tracer cycles. When the sliding time window begins, all counters which are configured and enabled begin incrementing. When the sliding time window expires, these counter values are loaded into their respective registers and the counts repeat again based on the sliding time window interval. When a periodic stream is enabled (statistic counters or access status) messages are generated on expiration of the sliding time window even if there was no bus activity.

The interrupt raw bit of the Interrupt Raw Status Register is set when the window expires.

12.6.2.11 Cross-Triggering

By using EMU0 and EMU1 signals from a device CPU, there is the ability to start and stop trace recording at specific times with the STM instead of around transaction events. Both EMU0/1 assert respective signals in the Debug Subsystem that control trace start and stop. The Transaction Qualifier Register is used to determine whether to trace transactions between the assertion of EMU0 and EMU1 with the QUALIF_EMU field.

12.6.2.12 Filtering

For generating statistics and event messages, there are several filtering modes which can be applied before exporting messages. Filtering can be applied based on:

- mstid on events B and E
- read/write on event B
- dtype on event B
- address range (inclusive of addresses within the range and exclusive outside the range) on event B. If any bytes of a transaction fall within an address range then the transaction will pass the inclusive filter or fail the exclusive filter.
- EMU0/1 control inputs on all events B, C and E

12.6.2.13 Accumulated Wait Time Counter

The accumulated-wait-time counter is a 24-bit counter used to accumulate the number of cycles spent waiting for a request to be serviced. It is exported as part of the statistics trace message. When a new request is issued from a master device to a slave, this event will be captured on the Event A interface of the Tracer for the corresponding slave and the number of pending requests will be incremented. When a transaction is sent to the slave for this request, this event will be captured on the Event B interface of the Tracer and the number of pending requests will be decremented. As long as there are pending requests, the accumulation cycle count will be incremented. Events E and F will also decrement the pending count.

12.6.2.14 Num Grant Counter

This is a 24-bit counter that is used to count the number of times arbitration has been granted (the number of event B that occurred with arb_last asserted). This counter is reset when the sliding timer window expires. This counter is enabled by a software register bit. This counter value is exported as part of the statistics trace message.

12.6.2.15 Overflow Status

Due to limits in the Tracer event FIFO size and the STM bandwidth, overflow may occur and event traces could be generated faster than they are drained. To deal with overflow, Tracer modules implement an overflow bit for events B, C, and E. When set to 1 in the corresponding trace message, the bit indicates that there has been some event loss between the current event and the previous event of the same type. A value of 0 indicates that no event loss has occurred.

12.6.3 SoC Cross-Triggering

The device supports a cross-triggering feature, which provides a way to propagate debug (trigger) events from one processor/subsystem/module to another. For example, a given subsystem A can be programmed to generate a debug event, which can then be exported as a global trigger across the device. Another subsystem B can be programmed to be sensitive to the trigger line input and to generate an action upon trigger detection.

Examples of debug events are: processor entering debug state, CS_PTM trigger, TBR full and acquisition complete, etc.

Examples of debug actions are: debug request generation, restart (Cortex-A15 synchronous run), interrupt request generation, start/stop trace, etc.

Subsystems cross-triggering is consolidated at device level by the XTRIG module, which is embedded in the DEBUGSS. XTRIG also supports the off-chip trigger channels (EMU0 and EMU1).

NOTE: XTRIG is not programmatically visible from the JTAG interface or any device processor. Thus, cross-triggering is programmed at subsystem level.

12.6.3.1 SoC Cross-Triggering Connection

The cross-trigger lines are shared by all the subsystems implementing cross-triggering. An Arm subsystem trigger event can therefore be propagated to any application subsystem or system trace component. The remote subsystem or system trace component can be programmed to be sensitive to the global SOC trigger lines to either:

- Generate a processor debug request
- Generate an interrupt request
- Start/Stop processor trace
- Start/Stop CBA transaction tracing through CP_Tracers
- Start external logic analyzer trace
- Stop external logic analyzer trace

Table 12-10 summarizes the SoC cross-triggering connections.

Table 12-10. Cross-Triggering Connections

Name	Source Triggers	Sink Triggers
Inside DEBUGSS		
Device-to-device trigger via EMU0/1 pins	Yes	Yes
MIPI-STM	No	Yes
DEBUGSS CT-TBR	Yes	Yes
DEBUGSS CS-TPIU	No	Yes
Outside DEBUGSS		
DSPSS	Yes	Yes
CP_Tracers	Yes	Yes
ARMSS/Cortex-A15	Yes	Yes
PMMC/Cortex-M3	Not supported	Not supported
PRU-ICSS	No	No
CT-TBR in PMMC TraceCell	Yes	Yes

Table 12-11 describes the cross-trigger connection between various cross-trigger sources and TI XTRIG module.

Table 12-11. TI XTRIG Assignment

Name	Assigned XTRIG Channel Number
C66x DSP	XTRIG 0
CP_Tracer 0..14	XTRIG 1.. 15

Table 12-12 describes the CoreSight cross-trigger connection between various cross-trigger sources and the DEBUGSS.

Table 12-12. CoreSight Triggering Assignment

Name	Assigned CoreSight Trigger Channel Number
ARMSS Cortex-A15	0
PMMC TraceCell	1
Reserved	2
Reserved	3

12.6.3.2 DEBUGSS Cross-Triggering Schemes and Components

The DEBUGSS implements two cross-triggering schemes:

- TI cross-triggering scheme
- CoreSight cross-triggering scheme

The DEBUGSS has multiple modules to manage triggering across these cross-triggering schemes:

- XTRIG module (TI cross-trigger module) to support merge and distribution of TI asynchronous triggers (2 channels per interface).
- CS_CTM (CoreSight Cross-Trigger Matrix) to support merge and distribution of CoreSight triggers (up to 4 interfaces and 4 channels per interface) to CoreSight components inside and outside of the DEBUGSS. The following subsystems are connected to the CoreSight Triggering Interface (CTI):
  - ARMSS (Cortex-A15)
  - TBR at the PMMC TraceCell
- CT_CTA module (CTools Cross-Trigger Adapter) to support conversion between TI asynchronous trigger format (2 channels) and CoreSight handshake-based asynchronous trigger format

This cross-triggering architecture of the debug subsystem allows it to provide the following features:

- Migration of a trigger originating from a CoreSight trigger source and ending at a TI trigger sink
- Migration of a trigger originating from a TI trigger source and ending at a CoreSight trigger sink
- Migration of a trigger originating from a CoreSight trigger source and ending at a CoreSight trigger sink
- Migration of a trigger originating from a TI trigger source in to a TI trigger sink

12.6.3.2.1 System Instrumentation Cross-Triggering

The CT_STM module interfaces with the SOC cross-triggering logic and implements two trigger input lines. These triggers can then be programmed by the user to start and stop collecting software instrumentation data:

- The EMU0 trigger line is coupled to software masters trace start
- The EMU1 trigger line is coupled to software masters trace stop

NOTE: The input triggers do not affect hardware masters trace. STM does not generate trigger outputs.

12.6.3.2.2 Cross-Triggering with External Devices

Based on DRM settings, the EMU0 and EMU1 lines may also be used as external triggers. Trigger0 is connected to the EMU0 device pin, and Trigger1 is connected to the EMU1 device pin.

12.6.3.2.3 CT-TBR

CT-TBR, used for hardware and software instrumentation, can generate and receive triggers. It uses the Arm CoreSight cross-triggering scheme.

12.6.3.2.4 TPIU

Trace Port Interface Unit (TPIU) is another Arm CoreSight component which receives trigger using the Arm CoreSight cross-triggering scheme.

12.6.3.3 ARMSS Cross-Triggering Schemes and Components

ARMSS uses the standard Arm embedded cross-trigger network for cross-triggering with other device subsystems/modules. The primary components making up the trigger network are CTIs and CTMs. There are multiple instances of CTI and CTM modules inside ARMSS wrapper. Only CTIs are programmable by software to generate/receive trigger events. They are listed as follows:

- The Cortex-A15 MPCore has one CTI
- At ARMSS level, there is a CTI used as a trigger interface for the CS-STM module
- In the ARMSS TraceCell, there is a CTI used as a trigger interface for the CT-TBR module

12.6.4 Emulation Aware Peripherals

Normal debug actions should not corrupt the correctness of the application and should keep the intrusion to the application to a minimum. This requires that all peripherals that are sensitive to debug events should be emulation aware. Specifically:

- A peripheral that is sensitive to a debug event (CPU halt) should behave properly during processor suspension. This is communicated to the peripheral from the CPU through an Emulation Suspend Event.
- A peripheral that is sensitive to debug access should distinguish accesses initiated from an emulator versus from an application. This is communicated from the debugging master to the peripheral as Emulation Requests
- Errors triggered by debug access should not cause any undesired interrupt or exception to the normal execution of a processor

12.6.4.1 Emulation Suspend Event

The device support a suspend mechanism, which provides a means to stop a closely-coupled hardware process running on a device peripheral when the host processor enters debug state. The suspend mechanism is important for debug to ensure that peripherals operate in a lock-step manner with a host processor.

To facilitate the operation of such peripherals, each of the C66x DSP, Arm Cortex-A15, and PMMC Cortex-M3 outputs a group of signals indicating their execution state. Accordingly, an entry is provided for each peripheral that needs to take into account the suspend signals from these processors.

Typically, two bits (SOFT and FREE) are implemented in the corresponding peripheral register dedicated to emulation control. The SOFT bit should be programmed based on whether or not an immediate pause of the peripheral function is required or if the peripheral suspend should occur only after a particular completion point is reached in the normal peripheral operation. The FREE bit should be programmed to enable or disable the emulation suspend functionality.

The DEBUGSS supports up to 32 emulation suspend sources (processor cores) and 64 emulation suspend sinks (peripherals). The assignment of processor cores and peripherals is shown in Table 12-13 and Table 12-14, respectively.

By default, the logical AND of all the processor cores' debug suspend signals is routed to all peripherals except for the tightly-coupled timers dedicated to each core. For these dedicated timers, default routing is such that each timer can be suspended by its associated DSP or Arm core. It is possible to select an individual core to be routed to the peripheral (e.g., used in tightly-coupled peripherals like timers), a logical AND of all cores (global peripherals) or a logical OR of all cores by programming the DRM module residing in the DEBUGSS. The DRM module implements a Suspend Control logic for routing purpose.

Table 12-13 summarizes the suspend sources (device cores) assignment.

Table 12-13. Suspend Sources Assignment

DRM Suspend Control Input	Device Core
0	DSP C66x
1-6	Unused (tied to 0)
7	PMMC Cortex-M3
8	Arm Cortex-A15
9-29	Unused (tied to 0)
30	Logical OR of core # 0,7,8
31	Logical AND of core # 0,7,8

Note that the emulation suspend signals of all the cores are synchronized to CPU/6, which is the frequency of the slowest peripheral that uses these signals.

Table 12-14 summarizes the suspend sinks (device peripherals) assignment.

Table 12-14. Suspend Sinks Assignment

DRM Suspend Control Output	Device Peripheral
0	TIMER_0
1	TIMER_1
2	TIMER_2
3	TIMER_3
4	TIMER_4
5	TIMER_5
6	TIMER_6
7	I2C_0
8	I2C_1
9	I2C_2
10	UART_0
11	UART_1
12	UART_2
13	PRU-ICSS
14	NSS_L
15	DCAN_0
16	DCAN_1
17	ePWM_0
18	ePWM_1
19	ePWM_2
20	ePWM_3
21	ePWM_4
22	ePWM_5
23	SRSS
24	MCBSP
25	eCAP_0
26	eCAP_1
27	eQEP
28-63	Unused

12.6.4.2 Emulation Requests

To make an emulation access to peripheral non-intrusive, some of peripherals need to distinguish accesses from CPU (application code) versus from emulator (debugger). In general, peripheral with following register/state behaviors should distinguish the access from CPU versus from emulator to preserve the peripheral's state:

- An access initiating a flag clear
- An access initiating a pointer update
- An access initiating a peripheral state-machine update
- An access error initiating a bit being set in error register

Otherwise, when a developer opens a memory window in CCS to view peripheral's registers, the application software execution will potentially be corrupted due to unexpected change of the peripheral state.

To facilitate such operation, the protocol of VBUSM and VBUSP buses, which interconnect different units in the system, provides signals *cemudbg* or *emudbg* to differentiate a bus transaction initiated from the CPU or from an external emulator.

All peripherals that need to be aware of the emulation access must make use of the *cemudbg* or *emudbg* signals correctly to prevent side effects from occurring.

12.6.4.3 Peripherals Emulation Support Summary

Table 12-15 lists all the peripherals on this device, and the status of whether or not a peripheral supports emulation suspend or emulation request events.

Table 12-15. Peripherals Emulation Support Summary

Peripheral	Emulation Suspend Support				Emulation Request Support (cemudbg/emudbg)
	Stop Mode	Real-Time Mode	FREE Bit	SOFT Bit	
EDMA_x (x=0,1)	N	N	N	N	Y
TRACER_x (x=0..14)	N	N	N	N	N
MPU	N	N	N	N	Y
CIC	N	N	N	N	Y
BOOT_CFG	N	N	N	N	Y
SEC_MGR	N	N	N	N	Y
PSC	N	N	N	N	N
PLL	N	N	N	N	N
Semaphore	N	N	N	N	Y
GPIO	N	N	N	N	N
McASP_x (x=0..2)	N	N	N	N	N
DDR3	N	N	N	N	Y
MSMC	N	N	N	N	Y
GPMC	N	N	N	N	N
QSPI	N	N	N	N	N
MMC	N	N	N	N	Y
SPI_x (x=0..3)	N	N	N	N	Y
PCIe	N	N	N	N	N
USB	N	N	N	N	N
ASRC	N	N	N	N	N
MLB	N	N	N	N	N
DSS	N	N	N	N	N
TIMER_x (x=0..6)	Y	N	Y	Y	N
I2C_x (x=0..2)	Y	N	Y	N	Y
UART_x (x=0..2)	Y	N	Y	Y	Y
PRU-ICSS (includes 2 eCAP)	Y	Y	Y	Y	N
NSS_L	Y	Y	Y	Y	N
DCAN_x (x=0,1)	Y	Y	N	Y	Y
ePWM_x (x=0..5)	Y	N	Y	Y	N
SRSS (includes I2C)	Y	N	Y	N	Y
McBSP	Y	N	Y	Y	Y
eCAP_x (x=0,1)	Y	Y	Y	Y	N
eQEP	Y	Y	Y	Y	N
PMMC	N	N	N	N	Y
MSGMNGR	N	N	N	N	Y

Refer to the corresponding peripheral chapters for more details.

12.6.5 DMA Tooling

The EDMA is a powerful tool that can provide detailed visibility to the data flow through the device with minimal intrusiveness on real-time operation. Through the EDMACC, any event can be used to trigger a DMA transfer, or sequence of transfers. Additionally, any DMA transfer can generate an event back to the EDMA to chain to a "debug" transaction.

The EDMA has full visibility to the chip memory map and can access any location with an address, except the internal DSP configuration registers. The DMA has ability to initiate transactions at any priority level, so that DMA "debug" transactions can be dedicated to priority 7 (low), and completed on a dedicated transfer controller channel (physical channel that performs reads/writes).

Debug transactions triggered by system events, or DMA transfers (chaining) of interest can be used to capture:

- System time
- Transfer information (address, ID, data, etc)
- Statistics transactions can be kicked off periodically

All EDMACC events can be traced through DSP, when configured in the system event router, which allow the application or host tooling to leverage the DSP AET logic. This facilitates:

- Matching DMA activity to CPU activity
- Counting time between events of interest, such as transfer time, time between transfers, etc

12.6.6 Power/Clock/Reset Emulation Support

12.6.6.1 Power/Clock Emulation Support

Debug should not hang when:

- It accesses a target which is under clock-gate or power-down since the access path is not functioning
- During an access, the target gets clock-gated or gets powered-down

To meet the above requirement, the DEBUGSS provides interfaces for a debugger to interact with the system power/clock control module (PSC module). The interface provides the power/clock/reset status of the subsystem under the Debug TAP or Debug Cores, and the controls to force the target active (clock and power back on) or prohibit it from being clock-gated or powered-down.

If an emulation access is made to a peripheral whose clock is gated, the emulation access shall still complete as this should be guaranteed by SCR and PSC infrastructure.

If an emulation access is made to a memory region whose clock is gated, the emulation access shall still complete as this should be guaranteed by SCR and PSC infrastructure.

12.6.6.1.1 Emulation Power Domain

The emulation power domain (PD1) can be switched off in normal operating mode to reduce active power consumption. Emulation power domain includes most of debug modules that are not mandatory in customer applications.

By default, emulation power domain is active after POR. Application software must switch off the domain when no debug is required. The application software can also wake up the emulation domain at any time, even if no debugger is attached to the device. This is typically required to support trace or monitor-based debugging.

12.6.6.1.2 ICEMelter Module

ICEMelter is a small module residing in the ALWAYS_ON power domain (PD0) that controls the wake-up and power-down of the emulation power domain. ICEMelter monitors several system signals and device pins to determine if the emulation logic should be active or in a dormant state.

ICEMelter includes:

- Monitoring of the EMU0 and EMU1 device pins
- Monitoring of the JTAG device pins
- Interaction with the ICEPick module
- Interaction with the device PSC module

ICEMelter is designed to support the following activities:

- Wake-up of the emulation power domain when an external debugger is first attached
- Power-down of the emulation power domain after an external debugger is disconnected

When a debugger is attached after the emulation power domain is switched off by the application software, based on the activity of the debug interface pins, ICEMelter can send a request to the PSC module to ramp-up the emulation power domain.

ICEMelter is not programmatically visible from the JTAG interface or any device processor.

12.6.6.2 Reset Emulation Support

Software developers often would like to debug the behavior of their program from a reset condition, or even through reset. In both cases, the emulator must continue to be able to connect to and communicate with on-chip emulation logic while the chip is held in reset. The following features are supported through the ICEPick interface of the DEBUGSS:

- Support for debugger-generated global reset (CHIP_0_RST) via ICEPick and a way to block this reset generated by applications, such as the watchdog timer reset
- Support for debugger-generated local reset via either ICEPick or individual processor's local resets, as

well as the way of blocking such subsystem reset

- TRSTz/TLR reset applies only to test logic and some security related logic. TRSTz/TLR reset does not affect any functional operation of the device

NOTE: Any warm reset that leads to a CHIP_0_RST causes power cycling of DSP subsystem including the debug logic. The workaround to this issue is to require the debug driver to assert the "Force Active" command during a debug session to keep the PSC from power cycling the DSP subsystem.

For more information about the reset emulation support in the device, see Chapter 5.3, *Reset Management*.

12.6.7 Debug Boot Modes

The DEBUGSS supports debug boot modes determined from its input signals EMU[1:0] upon a rising edge of POR reset. Table 12-16 shows the details:

Table 12-16. Debug Boot Modes

EMU[1:0] Pins	TAPs in TDI->TDO path	Comment
0xb	N/A	Reserved
10b	ICEPick	Hardware Wait-In-Reset (WIR)
11b	ICEPick	Default condition

This feature is used to debug run-away boot code on the device.

12.6.7.1 Default Condition

None of the secondary TAPs are selected. The ICEPick TAP is the only TAP between device-level TDI and TDO pins. This is the recommended boot mode.

12.6.7.2 Hardware Wait-In-Reset

The device can boot to invoke WIR mode. If the device is booted in this mode, all processors within the device that support a TAP through ICEPick are held in reset until released. Individual processors may be released from reset (local), or all processors held in the reset state may be released at the same time (global).

12.7 Programming Guidelines

12.7.1 Application Support

TI includes a set of system level debug facilities in the Debug Subsystem of devices known as Chip Tools (CTools). For easy integration of DSP Core Trace and System Trace into applications, a set of libraries commonly referred to as CToolsLib are provided. CToolsLib is a collection of embedded target APIs/libraries focused on enabling easy access to the CTools. CToolsLib encompasses the following libraries to assist in trace integration:

- DSP Trace Library
- AET Library
- STM Library
- CP Tracer Library
- ETB/TBR Library
- PTM/ETM Library
- CTools Use-Case Library

For more information about these libraries, downloadable files, and other useful links, please visit the CToolsLib Wiki site:

http://processors.wiki.ti.com/index.php/CToolsLib

The previous link connects to TI community resources. Linked contents are provided “AS IS” by the respective contributors. They do not constitute TI specifications and do not necessarily reflect TI's views; see TI's Terms of Use.

12.7.2 Programming Overview

The following sections demonstrate the capabilities of the cTools libraries for various debug tools. Care should be taken to make sure the power and sleep controller for the Debug Subsystem is enabled before proceeding with further debug configuration.

12.7.2.1 DSP Core Trace Specific Configuration

DSPTraceLib enables the user to do the following:

- Claim ownership of the Trace Export Block
- Configure Pin Manager
- Claim and enable trace streams
- Configure type of trace capture
- Start/stop trace recording

12.7.2.2 AET Specific Configuration

CToolsLib for AET provides APIs for configuring the AET logic for various trace jobs. The AET library enables the user to do the following:

- Initialize AET
- Claim AET
- Set up AET to trigger on a certain set of conditions
- Start/Stop triggering with AET

12.7.2.3 STM Specific Configuration

CToolsLib provides various APIs for STM configuration as follows:

- Sending various types of software messages.
- Configuring the Tracers and generating various types of hardware messages

In addition, the STM can be configured for the following features:

1. Setting STM for optimized or unoptimized operation. In optimized operation only a pointer to the message string is transported, unlike in unoptimized operation where the entire null terminated string is transported.
2. Configuring STM channel resolution to either 1024 or 4096 bytes for specified channels

12.7.2.4 Debug Pin Manager Configuration

To export trace data off chip to an external trace receiver, the Debug Pin Manager of the DRM must be programmed to specify which pins will be used for export and for what purpose. With TI's CToolsLib, this configuration is performed automatically.

Glossary

Glossary

TI Glossary **—** This glossary lists and explains terms, acronyms, and definitions.

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265